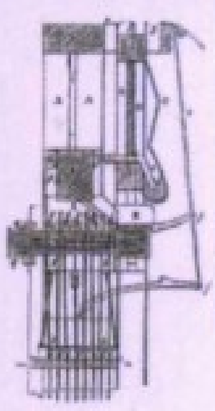
国外计算机科学教材系列

自然语言处理综论

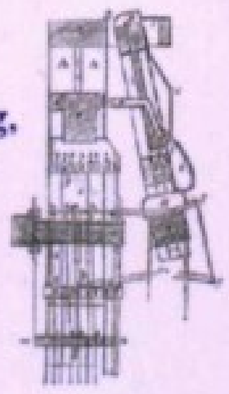
Speech and Language Processing

An Introduction to Natural Language Processing,
Computational Linguistics, and Speech Recognition



SPEECH and
LANGUAGE
PROCESSING

An Introduction to
Natural Language Processing,
Computational Linguistics,
and Speech Recognition



DANIEL JURAFSKY & JAMES H. MARTIN

[美] Daniel Jurafsky
James H. Martin 著

冯志伟 孙 乐 译

PEARSON Prentice Hall

PHEI

電子工業出版社
Publishing House of Electronics Industry
http://www.phei.com.cn

经典教材

自然语言处理综论

Speech and Language Processing

An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition

高采用率的自然语言处理经典教材！

本书英文版出版之后好评如潮，国外许多著名大学纷纷把本书选为自然语言处理和计算语言学课程的主要教材，该书被誉为该领域教材的“黄金标准”。

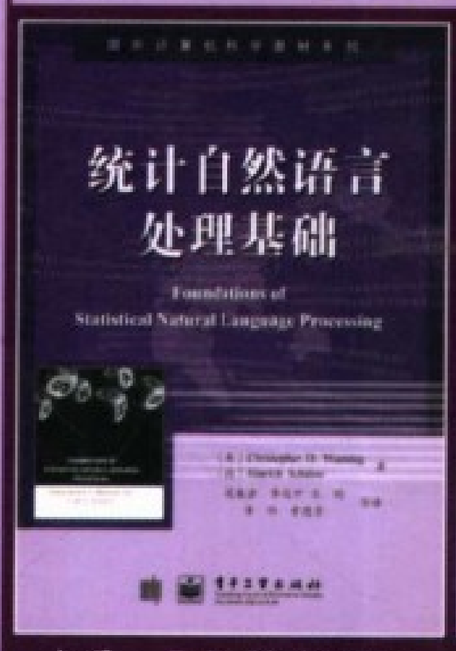
本书综合了自然语言处理、计算语言学和语音识别的内容，全面论述计算机自然语言处理，深入探讨计算机处理自然语言的词汇、句法、语义、语用等各个方面的问题，介绍了自然语言处理的各种现代技术。

本书四大特色：

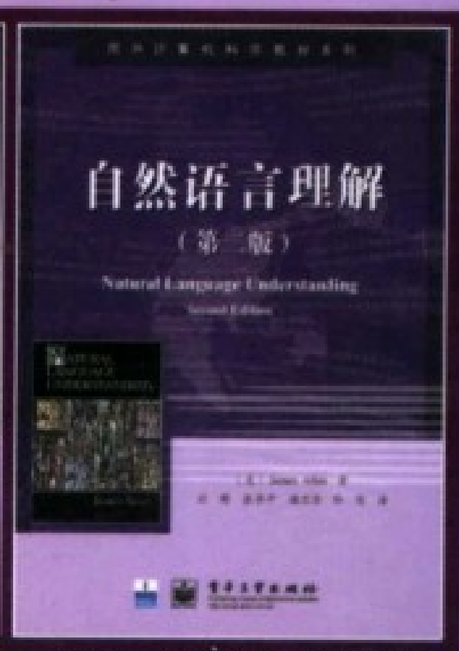
■ 覆盖全面 ■ 强调实用 ■ 注重评测 ■ 语料为本

本书配套站点：http://www.prenhall.com/jurafsky

相关图书



书号：7-5053-9921-7



书号：7-121-00755-X

Daniel Jurafsky

在美国加利福尼亚大学获计算机科学博士学位，现于美国科罗拉多大学语言学系和计算机科学系任教，并在认知科学研究所工作，主要研究方向为语言的概率模型和语音信息处理。由于他在语音和语言处理方面的成就，于1997年获美国NSF职业奖。

James H. Martin

在美国加利福尼亚大学获计算机科学博士学位，现任职于美国科罗拉多大学计算机科学系和认知科学研究所，主要研究方向为计算语义学、机器学习和信息检索。

冯志伟

国家教育部语言文字应用研究所研究员、博士生导师。先后在北京大学和中国科学技术大学获双硕士学位，在语音和语言的计算机处理领域具有多年的研究经验，曾在多个国家参与研究和教学工作，主要研究方向为自然语言处理、计算语言学和机器翻译，主要著作有《自然语言的计算机处理》和《数理语言学》等18部。

ISBN 7-121-00776-2

9 787121 007767 >

Pearson Education

传播教育信息 共享教育资源

华信教育资源网 www.hxedu.com.cn

欢迎登录 获取优质教学资源

PEARSON Prentice Hall

PHEI

责任编辑：马 岚
特约编辑：马爱文
责任美编：毛惠庚

本书贴有激光防伪标志，凡没有防伪标志者，属盗版图书

ISBN 7-121-00776-2 定价：78.00元

国外计算机科学教材系列

自然语言处理综论

Speech and Language Processing

An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition

[美] Daniel Jurafsky
James H. Martin 著

冯志伟 孙 乐 译

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书是一本全面系统地讲述计算机自然语言处理的优秀教材。本书英文版出版之后好评如潮，国外许多著名大学纷纷把本书选为自然语言处理和计算语言学课程的主要教材，该书被誉为该领域教材的“黄金标准”。本书包含的内容十分丰富，分为四个部分，共21章，深入细致地探讨了计算机处理自然语言的词汇、句法、语义、语用等各个方面的问题，介绍了自然语言处理的各种现代技术。从层次的角度看，本书的论述是按照自然语言的不同层面逐步展开的，首先论述单词的自动形态分析，接着论述自动句法分析，然后论述各种语言单位的自动语义分析，最后论述连贯文本的自动分析、对话与会话的智能代理以及自然语言生成。从技术的角度看，本书介绍了正则表达式、有限状态自动机、文本-语音转换、发音与拼写的概率模型、词类自动标注、*N*元语法、隐马尔可夫模型、上下文无关语法、特征与合一、词汇化剖析与概率剖析、一阶谓词演算、词义排歧、修辞结构理论、机器翻译等非常广泛的内容。本书具有“覆盖全面、注重实用、强调评测、语料为本”四大特色。在本书的配套网站上，还提供了相关的资源和工具，便于读者在实践中进一步提高。

本书不仅可以作为高等学校自然语言处理和计算语言学等课程的本科生和研究生教材，而且也是从事自然语言处理相关领域的研究人员和技术人员的必备参考。

Simplified Chinese edition Copyright © 2005 by PEARSON EDUCATION ASIA LIMITED and Publishing House of Electronics Industry.
Speech and Language Processing: Aa Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, ISBN: 0130950696 by Daniel Jurafsky, James H. Martin. Copyright © 2000.
All Rights Reserved.
Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Prentice Hall.
This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macau).

本书中文简体字翻译版由电子工业出版社和Pearson Education培生教育出版亚洲有限公司合作出版。未经出版者预先书面许可，不得以任何方式复制或抄袭本书的任何部分。
本书封面贴有 Pearson Education 培生教育出版集团激光防伪标签，无标签者不得销售。

版权贸易合同登记号　图字：01-2003-0357

图书在版编目（CIP）数据

自然语言处理综论 /（美）朱夫斯凯（Jurafsky, D.）等著；冯志伟，孙乐译.
北京：电子工业出版社，2005.6
（国外计算机科学教材系列）
书名原文：Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition
ISBN 7-121-00776-2

Ⅰ. 自...　Ⅱ. ①朱... ②冯... ③孙...　Ⅲ. 自然语言处理-高等学校-教材　Ⅳ. TP391

中国版本图书馆CIP数据核字（2005）第047102号

责任编辑：马　岚　　特约编辑：马爱文
印　　刷：北京市天竺颖华印刷厂
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编：100036
经　　销：各地新华书店
开　　本：787 × 1092　1/16　　印张：38.25　字数：1079千字
印　　次：2005年6月第1次印刷
定　　价：78.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换；若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

出 版 说 明

21世纪初的5至10年是我国国民经济和社会发展的重要时期，也是信息产业快速发展的关键时期。在我国加入WTO后的今天，培养一支适应国际化竞争的一流IT人才队伍是我国高等教育的重要任务之一。信息科学和技术方面人才的优劣与多寡，是我国面对国际竞争时成败的关键因素。

当前，正值我国高等教育特别是信息科学领域的教育调整、变革的重大时期，为使我国教育体制与国际化接轨，有条件的高等院校正在为某些信息学科和技术课程使用国外优秀教材和优秀原版教材，以使我国在计算机教学上尽快赶上国际先进水平。

电子工业出版社秉承多年来引进国外优秀图书的经验，翻译出版了“国外计算机科学教材系列”丛书，这套教材覆盖学科范围广、领域宽、层次多，既有本科专业课程教材，也有研究生课程教材，以适应不同院系、不同专业、不同层次的师生对教材的需求，广大师生可自由选择和自由组合使用。这些教材涉及的学科方向包括网络与通信、操作系统、计算机组织与结构、算法与数据结构、数据库与信息处理、编程语言、图形图像与多媒体、软件工程等。同时，我们也适当引进了一些优秀英文原版教材，本着翻译版本和英文原版并重的原则，对重点图书既提供英文原版又提供相应的翻译版本。

在图书选题上，我们大都选择国外著名出版公司出版的高校教材，如Pearson Education培生教育出版集团、麦格劳－希尔教育出版集团、麻省理工学院出版社、剑桥大学出版社等。撰写教材的许多作者都是蜚声世界的教授、学者，如道格拉斯·科默(Douglas E. Comer)、威廉·斯托林斯(William Stallings)、哈维·戴特尔（ Harvey M. Deitel)、尤利斯·布莱克（ Uyless Black ）等。

为确保教材的选题质量和翻译质量，我们约请了清华大学、北京大学、北京航空航天大学、复旦大学、上海交通大学、南京大学、浙江大学、哈尔滨工业大学、华中科技大学、西安交通大学、国防科学技术大学、解放军理工大学等著名高校的教授和骨干教师参与了本系列教材的选题、翻译和审校工作。他们中既有讲授同类教材的骨干教师、博士，也有积累了几十年教学经验的老教授和博士生导师。

在该系列教材的选题、翻译和编辑加工过程中，为提高教材质量，我们做了大量细致的工作，包括对所选教材进行全面论证；选择编辑时力求达到专业对口；对排版、印制质量进行严格把关。对于英文教材中出现的错误，我们通过与作者联络和网上下载勘误表等方式，逐一进行了修订。

此外，我们还将与国外著名出版公司合作，提供一些教材的教学支持资料，希望能为授课老师提供帮助。今后，我们将继续加强与各高校教师的密切联系，为广大师生引进更多的国外优秀教材和参考书，为我国计算机科学教学体系与国际教学体系的接轨做出努力。

电子工业出版社

教材出版委员会

主　任　　杨芙清　　北京大学教授
中国科学院院士
北京大学信息与工程学部主任
北京大学软件工程研究所所长

委　员　　王　珊　　中国人民大学信息学院院长、教授

胡道元　　清华大学计算机科学与技术系教授
国际信息处理联合会通信系统中国代表

钟玉琢　　清华大学计算机科学与技术系教授
中国计算机学会多媒体专业委员会主任

谢希仁　　中国人民解放军理工大学教授
全军网络技术研究中心主任、博士生导师

尤晋元　　上海交通大学计算机科学与工程系教授
上海分布计算技术中心主任

施伯乐　　上海国际数据库研究中心主任、复旦大学教授
中国计算机学会常务理事、上海市计算机学会理事长

邹　鹏　　国防科学技术大学计算机学院教授、博士生导师
教育部计算机基础课程教学指导委员会副主任委员

张昆藏　　青岛大学信息工程学院教授

中文版序言

The goal of a textbook author is the same as the goal of any teacher: passing on our love for our field to a new generation of students, encouraging them to do innovative and creative new work, and helping them to advance the state of human knowledge. For a textbook in the interdisciplinary area of speech and language processing, there are the additional goals of enabling students from differing backgrounds (computer science, linguistics, electrical engineering) to acquire the knowledge and tools of the new interdisciplinary field, and to develop an appreciation for the beauty and complexity and variety of human language. We therefore feel extremely lucky that Feng Zhiwei Laoshi, aided by Dr. Sun Le, undertook the arduous job of translating this book. Feng Laoshi is the perfect scholar for the job of translating such a book, because of his long experience in our field, his wide breadth of research interests throughout computational linguistics in general and Chinese computational linguistics specifically, his remarkable familiarity with the state of our field across the world, from China to France, from Korea to Germany, and of course his expertise on translation as a research area! We are also very excited that this translation into Chinese is the first translation of our book out of English. China's long history of the study of language is of course well known, and in this new century the young scientists of China are already playing a key role in the important scientific advances of our field. We look forward to even more amazing contributions from China and hope that our small book, now with the help of Feng Laoshi and Dr. Sun, can provide a small aide in the great role that Chinese scientists are playing on the world scientific stage!

Daniel Jurafsky and James H. Martin

Palo Alto, California, and Boulder, Colorado

——译文——

教材的作者与所有教师有着相同的目标，即把我们对于本专业的热爱传达给新一代的学生，鼓励他们进行创新性的研究和探索，帮助他们把人类知识进一步向前推进。由于语音和语言的计算机处理属于交叉学科的领域，所以这本关于该交叉学科领域的教材还有其特定的目标。这些特定的目标就是使来自不同知识背景（计算机科学、语言学和电子工程）的学生掌握这门新的交叉学科的基本知识和工具，并在学习过程中循序渐进地感受人类语言的美妙性、复杂性和多样性。因此，当了解到冯志伟老师在孙乐博士的协助下承担了把这本教材翻译成中文的艰辛工作时，我们感到无比荣幸。我们认为，冯志伟老师是翻译这本教材的最理想的学者，因为他在这个专业领域具有多年的经验；他的研究兴趣涉及面广，既包括普通的计算语言学研究，也包括具体的汉语计算语言学的研

究；他对于这个学科在全世界的情况了如指掌，从中国到法国，从韩国到德国，他都亲身参与了这些国家的计算语言学研究工作；并且，翻译一直是冯老师长期从事的一个研究领域，他当然也是精研通达的翻译内行！这个中译本是英文原著的第一个外文译本，它的出版使我们非常激动和振奋。众所周知，中国在语言研究方面有着悠久的历史，在21世纪，中国年轻一代的科学工作者在这个领域的一些重要科学进展方面已经起着关键性的作用。我们期待着中国在这个领域里进一步做出更加出色的贡献。我们也希望，在中国科学工作者为全世界的科学进步事业所发挥的巨大作用中，由于冯老师和孙乐博士的帮助，拙著也能够为此尽我们的绵薄之力！

译 者 序

采用计算机技术来研究和处理自然语言是20世纪40年代末期和20世纪50年代才开始的，50多年来，这项研究取得了长足的进展，成为了计算机科学中一门重要的新兴学科——自然语言处理（Natural Language Processing，简称NLP）。

我们认为，计算机对自然语言的研究和处理，一般应经过以下4方面的过程：

1. 把需要研究的问题在语言学上加以形式化，使之能以一定的数学形式，严密而规整地表示出来；
2. 把这种严密而规整的数学形式表示为算法，使之在计算上形式化；
3. 根据算法编写计算机程序，使之在计算机上加以实现；
4. 对于所建立的自然语言处理系统进行评测，使之不断地改进质量和性能，以满足用户的要求。

美国计算机科学家Bill Manaris在*Advanced in Computers*（《计算机进展》）第47卷的Natural language processing: A human-computer interaction perspective(《从人机交互的角度看自然语言处理》）一文中曾经给自然语言处理提出了如下的定义：

自然语言处理可以定义为研究在人与人交际中以及在人与计算机交际中的语言问题的一门学科。自然语言处理要研制表示语言能力（linguistic competence）和语言应用（linguistic performance）的模型，建立计算框架来实现这样的语言模型，提出相应的方法来不断地完善这样的语言模型，根据这样的语言模型设计各种实用系统，并探讨这些实用系统的评测技术。

Bill Manaris关于自然语言处理的这个定义，比较全面地表达了计算机对自然语言的研究和处理的上述四个方面的过程。我们认同这样的定义。

根据这样的定义，我们认为，建立自然语言处理模型需要如下不同平面的知识：

1. 声学和韵律学的知识：描述语言的节奏、语调和声调的规律，说明语音怎样形成音位。
2. 音位学的知识：描述音位的结合规律，说明音位怎样形成语素。
3. 形态学的知识：描述语素的结合规律，说明语素怎样形成单词。
4. 词汇学的知识：描述词汇系统的规律，说明单词本身固有的语义特性和语法特性。
5. 句法学的知识：描述单词（或词组）之间的结构规则，说明单词（或词组）怎样形成句子。
6. 语义学的知识：描述句子中各个成分之间的语义关系，这样的语义关系是与情景无关的，说明怎样从构成句子的各个成分推导出整个句子的语义。
7. 话语分析的知识：描述句子与句子之间的结构规律，说明怎样由句子形成话语或对话。
8. 语用学的知识：描述与情景有关的情景语义，说明怎样推导出句子具有的与周围话语有关的各种涵义。
9. 外界世界的常识性知识：描述关于语言使用者和语言使用环境的一般性常识，例如，语言使用者的信念和目的，说明怎样推导出这样的信念和目的内在的结构。

当然，关于自然语言处理所涉及的知识平面还有不同的看法，不过，一般而言，大多数的自然语言处理研究人员都认为，这些语言学知识至少可以分为词汇学知识、句法学知识、语义学知识和

语用学知识等平面。每一个平面传达信息的方式各不相同。例如，词汇学平面可能涉及具体的单词的构成成分（如语素）以及它们的屈折变化形式的知识；句法学平面可能涉及在具体的语言中单词或词组怎样结合成句子的知识；语义学平面可能涉及怎样给具体的单词或句子指派意义的知识；语用学平面可能涉及在对话中话语焦点的转移以及在给定的上下文中怎样解释句子的涵义的知识。

下面我们具体说明在自然语言处理中这些知识平面的一般情况。如果对计算机发一个口头的指令：Delete file x（删除文件 X），为了通过自然语言处理系统让计算机理解这个指令的涵义，并且执行这个指令，一般来说需要经过如图 0.1 所示的处理过程。

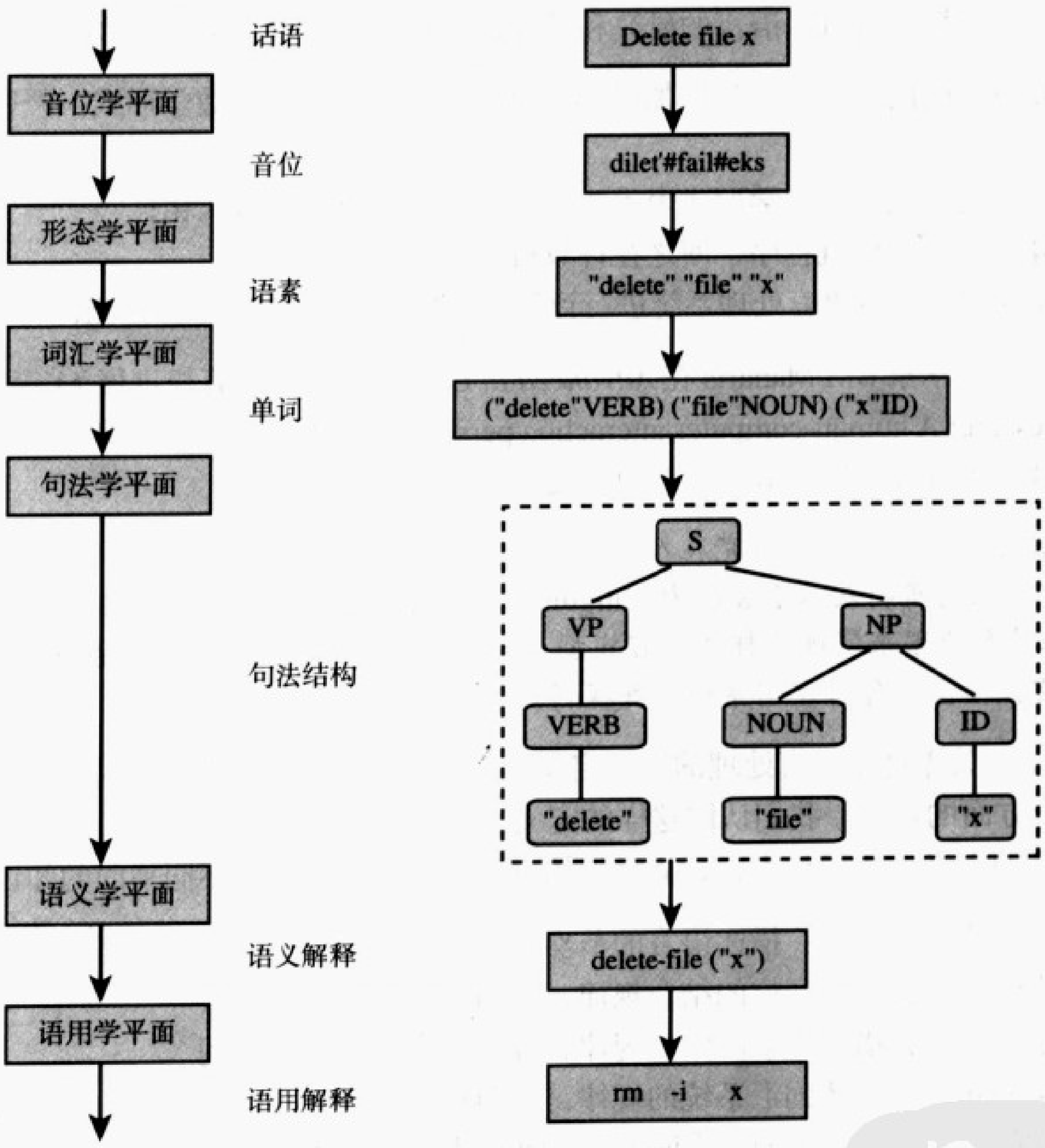


图 0.1　自然语言处理系统中的知识平面

从图中可以看出，自然语言处理系统首先把指令 Delete file x 在音位学平面转化成音位系列 dilet'#fail#eks，然后在形态学平面把这个音位系列转化为语素系列 delete，file 和 x，接着在词汇学平面把这个语素系列转化为单词系列并标注相应的词性：("delete"VERB) ("file"NOUN) ("x"ID)，在句法学平面进行句法分析，得到这个单词系列的句法结构，用树形图表示，在语义学平面得到这个句法结构的语义解释 delete-file ("x")，在语用学平面得到这个指令的语用解释 rm -i x，最后让计算机执行这个指令。

这个例子来自美国自然语言处理学者 Wilensky 为 UNIX 设计的一个语音理解界面，叫做 UNIX Consultant。这个语音理解界面使用了上述的第 1 个至第 6 个平面的知识，得到口头指令 Delete file x

的语义解释：delete-file ("x")。然后，使用第 8 个平面的语用学知识把这个语义解释转化为计算机的指令语言 rm -i x，让计算机执行这个指令，这样便可以使用口头指令来指挥计算机的运行了。

不同的自然语言处理系统需要的知识平面可能与UNIX Consultant不一样，根据实际应用的不同要求，很多自然语言处理系统只需要使用上述9个平面中的部分平面的知识就行了。例如，书面语言的机器翻译系统只需要第3个至第7个平面的知识，个别的机器翻译系统还需要第8个方面的知识；语音识别系统只需要第 1 个至第 5 个平面的知识。

上述9个平面的知识主要涉及的是语言学知识，由于自然语言处理是一个多边缘的交叉学科，除了语言学之外，它还涉及如下的知识领域：

- **计算机科学**：给自然语言处理提供模型表示、算法设计和计算机实现的技术。
- **数学**：给自然语言处理提供形式化的数学模型和形式化的数学方法。
- **心理学**：给自然语言处理提供人类言语行为的心理模型和理论。
- **哲学**：给自然语言处理提供关于人类的思维和语言的更深层次的理论。
- **统计学**：给自然语言处理提供基于样本数据来预测统计事件的技术。
- **电子工程**：给自然语言处理提供信息论的理论基础和语言信号处理技术。
- **生物学**：给自然语言处理提供大脑中人类语言行为机制的理论。

自然语言处理需要的知识如此之丰富，涉及的领域如此之广泛，而我们翻译的美国科罗拉多大学 Daniel Jurafsky 和 James Martin 的这本著作正好满足了这样的要求。

几年前我从韩国到新加坡参加国际会议时，在书店发现此书，马上就被它丰富的内容和流畅的表达吸引住了。会议结束回到韩国之后，我就开始认真阅读此书，我发现此书覆盖面非常广泛，理论分析十分深入，而且强调实用性和注重评测技术，几乎所有的例子都来自真实的语料库，此书的内容不仅覆盖了我们在上面所述的9个平面的语言学知识和外在世界的常识性知识，而且还涉及到计算机科学、数学、心理学、哲学、统计学、电子工程和生物学等领域的知识，我怀着极大的兴趣前后通读了两遍。当时我在韩国科学技术院电子工程与计算机科学系担任访问教授，在我给该系博士研究生开的“自然语言处理-II”（NLP-II）的课程中，使用了该书的部分内容，效果良好。我觉得这确实是一本很优秀的自然语言处理的教材。我常常想，如果我们能够把这本优秀的教材翻译成中文，让国内的年轻学子们也能学习本书，那该是多么好的事情！

后来，在一次机器翻译研讨会上，电子工业出版社的编辑找到我，告诉我说他们打算翻译出版此书。当时电子工业出版社已经进行过调查，目前国外绝大多数大学的计算机科学系都采用此书作为“自然语言处理”课程的研究生教材，他们希望我来翻译这本书，与电子工业出版社配合，推出高质量的中文译本。我们双方的想法不谋而合，于是，我欣然接受了本书的翻译任务，开始进行本书的翻译。

我虽然已经通读过本书两遍，对于本书应该说是有一定的理解了，但是，亲自动手翻译起来，却不像原来想像的那样容易，要把英文的意思表达为确切的中文，下起笔来，总有汲深绠短之感，大量的新术语如何用中文来表达，也是颇费周折令人踌躇的难题。我利用了全部的业余时间来进行翻译，连续工作了 11 个月，当翻译完 14 章（全书的三分之二）的时候，我患了眼病，视力出现障碍，难于继续翻译工作，还剩下 7 章（全书的三分之一）没有翻译，“行百里者半九十”，这 7 章的翻译工作究竟如何来完成呢？正当我束手无策一筹莫展的时候，中国科学院软件研究所副研究员孙乐博士表示愿意继续我的工作，与我协作共同完成本书的翻译。孙乐博士有很好的自然语言处理的基础，我们又是忘年之交的好朋友，由他来继续我的翻译工作是最理想不过的了，电子工业出版社也同意孙乐参与本书的翻译。孙乐博士的翻译工作十分认真，他每翻译一章，就交给我审校，遇到

疑难问题时我们共同切磋，反复推敲，他顺利地完成了第15章到第21章的翻译，现在，在我们两人的通力合作下，全书的翻译总算大功告成了。原书每章都有习题，由于这些习题涉及的语言背景不适合中国读者，因此在中文版中未将习题包括在内。

正如本书作者指出的，本书具有“覆盖全面，强调实用，注重评测，语料为本”的特点，我们希望，这个中文译本能够在我国的自然语言处理的教学和科学研究中，产生积极的作用，我们还希望，读者能够喜欢这个译本，并给我们提出批评和指正。

本书译者的部分工作得到国家自然科学基金（编号：60203007）和北京市科技新星计划（编号：H020820790130）的资助，特此致谢。

冯志伟

前　言

现在，语音和语言的计算机处理进入了一个令人振奋的时期。在这个时期，历史上彼此不同的研究部门（自然语言处理、语音识别、计算语言学、计算心理语言学）开始融合在一起。语音识别研究的商品化以及对于基于互联网的语言处理技术的需求，有力地推动了各种实用的自然语言处理系统的开发。由于使用大规模的联机语料库，使得在从语音到话语的各个不同的层面都可以使用统计方法。我们在设计这本既可作为教学之用又可作为参考书之用的专著时，试图描绘出各个不同学科开始融合在一起的这种情景。本书具有如下的特点：

1. **覆盖全面**　为了统一地描述语音处理和语言处理，本书涵盖了传统上分别在不同的系和不同课程中讲授的内容。例如，在电子工程系的语音识别课程中的内容；在计算机科学系的自然语言处理课程中的自动句法分析、语义解释、语用学等内容；在语言学系的计算语言学课程中的计算形态学和计算音系学等内容。本书介绍了这些领域中的基本算法，无论这些算法原来是在语音处理还是在语言处理中提出的，无论它们原来是从逻辑的角度还是从统计的角度提出的，我们力求将来自不同领域的算法合在一起统一描述。我们也试图把一些诸如拼写检查、信息检索和信息抽取这样的应用领域的内容包括在本书中，使其覆盖得更全面。这种广为覆盖的方法的一个潜在问题是，我们只好把每个领域中的一些概论性的材料也包括到本书中。因此，在阅读本书时，语言学家可以跳过有关发音语音学方面的章节，计算机科学家可以跳过有关正则表达式的章节，电子工程师可以跳过有关信号处理的章节。当然，尽管这本书写得这样长，我们也不可能做到包罗万象。正因为如此，本书不能替代语言学、自动机和形式语言理论，特别是关于统计学和信息论的各种专门著作，这些著作显然是非常重要的。
2. **注重实用**　理论联系实际是非常重要的。在本书中，我们始终注意把自然语言处理的算法和技术（从隐马尔可夫模型到合一算法，从λ运算到基于转换的学习）应用于解决现实世界中遇到的各种重要问题，例如拼写检查、文本文献检索、语音识别、网页信息处理、词类标注、机器翻译、口语对话代理等。为了达到这样的目的，我们在每一章中都要讲授一些关于自然语言处理的应用问题。这种方法的好处是，当介绍有关自然语言处理的知识时，可以给学生们提供一个背景来理解和模拟特定领域中的应用问题。
3. **强调评测**　近年来，在自然语言处理中统计算法越来越受到重视，语音处理和语言处理的有组织的评测系统越来越多，这些都使得评测得到了越来越多的强调和重视。因此，我们在本书中许多领域设立了“方法论探讨”（Methodology Box），具体讲述怎样评测一个系统。例如，我们介绍了训练集和测试集的概念，交叉确认（cross-validation）以及诸如困惑度（perplexity）这样的信息论评测指标。
4. **语料为本**　现代的语音处理和语言处理很多是建立在公共资源的基础上的。这些资源包括：语音生语料库和文本生语料库，标注语料库和树库，用于语音标记、词类标记、自动句法分析、词义以及对话层面的现象的标准标注集等。我们力图在全书中介绍很多这样的重要语言资源（例如，Brown, Switchboard, callhome, ATIS, TREC, MUC, BNC 等语料库），并且提供

很多有用的标记集的完整清单以及编码技巧（例如 Penn Treebank, CLAWS C5 和 C7，以及 ARPAbet），不过难免会有遗漏。此外，除了在本书中直接包括了很多资源的URL（Uniform Resource Locator）之外，我们还把这些资源放在本书的网站上，这样即可使这些得到及时的更新。

本书首先可以用做研究生或高年级本科生的教材或系列教材。由于本书的覆盖面广，并且有大量的算法，所以，本书也可以用做语音处理和语言处理的各个领域中的大学生和专业人员的参考书。

本书概览

除了前言和书后面的附录之外，本书共分为四个部分。第一部分是“词汇的计算机处理”，讲述与词汇的计算机处理有关的语音学、音系学、形态学的基本概念，介绍语音和词汇计算机处理中的各种算法，如有限自动机、有限转录机、加权转录机、*N*元模型、隐马尔可夫模型等。第二部分是“句法的计算机处理”，介绍英语的词类和短语的结构语法，讲述用于词类处理和结构处理的一些主要的算法，如基于HMM的词类标注、基于转换的学习、CYK分析算法、Earley分析算法、合一与类型特征结构、词汇化剖析和概率剖析以及诸如Chomsky层级分类和抽吸引理（pumping lemma）等分析工具。第三部分是“语义的计算机处理”，介绍一阶谓词演算以及语义的各种表示方法，组合语义分析的各种方法、信息抽取、言语理解和机器翻译。第四部分是“语用的计算机处理”，讲述所指判定（reference resolution）、话语的结构和连贯性、口语对话的现象、对话和言语行为模式、对话管理以及机器翻译和自然语言生成中的各种处理方法。

本书使用方法

本书材料丰富，可作为两学期的语音处理和语言处理系列教材。本书也可以作为各种不同用途的一个学期的教材使用。

自然语言处理 一个季度	自然语言处理 一个学期	语音与语言处理 一个学期	计算语言学 一个季度
1. 导论	1. 导论	1. 导论	1. 导论
2. 正则表达式，FSA	2. 正则表达式，FSA	2. 正则表达式，FSA	2. 正则表式，FSA
8. 词性标注	3. 形态学，FST	3. 形态学，FST	3. 形态学，FST
9. 上下文无关文法	6. *N*元语法	4. 计算音系学	4. 计算音系学
10. 剖析	8. 词性标注	5. 发音的概率模型	10. 剖析
11. 合一	9. 上下文无关文法	6. *N*元语法	11. 合一
14. 语义学	10. 剖析	7. HMM与语音识别	13. 复杂性
15. 语义分析	11. 合一	8. 词性标注	16. 词汇语义学
18. 话语	12. 概率剖析	9. 上下文无关文法	18. 话语
20. 生成	14. 语义学	10. 剖析	19. 对话
	15. 语义分析	12. 概率剖析	
	16. 词汇语义学	14. 语义学	
	17. WSD与信息检索	15. 语义分析	
	18. 话语	19. 对话	
	20. 生成	21. 机器翻译	
	21. 机器翻译		

本书的某些章节也可以选用于人工智能、认知科学或者信息检索等课程。

致谢

有三位作者对本书做出了贡献，他们协助我们写了有关章节。他们是Andy Kehler, Keith Vander Linden 和 Nigel Ward。Andy Kehler 写了第 18 章（话语分析），Keith Vander Linden 写了第 20 章（生成），Nigel Ward 写了第 21 章（机器翻译）。Andy Kehler 还写了 19.4 节。此外，Paul Taylor 帮助我们写了 4.7 节和 7.8 节。

Daniel Jurafsky 在此要感谢他的父母，是他们鼓励Daniel把每件事都做得尽善尽美，按时完成，并且抽时间到体育馆去锻炼身体。感谢 Nelson Morgan，因为 Morgan 引导他从事语音识别的研究，并且教导他对任何事情都要问一个“这样行吗？”感谢 Jerry Feldman，因为 Jerry 经常帮助他寻找问题的正确答案，教导他对于任何事情都要问一问：“这确实是重要的吗？”感谢 Chuck Fillmore，因为Chuck 是他的第一个咨询人，和他分享对于语言的爱好，特别是对于研究论元结构的爱好，教导他要始终重视数据，并且要用实例来说明问题，只有当数据真正有价值时，才值得花时间和精力去研究。感谢 Robert Wilensky，Robert 是他的博士论文的指导教师，Robert 教导他懂得了合作共事以及团队精神的重要性。感谢Doris 和Cary, Elaine 和Eric, Irene 和Sam, Susan 和Richard, Lisa 和Mike, Mike 和 Fia, Erin 和 Chris, Eric 和 Beth, Pearl 和 Tristan, Bruce 和 Peggy, Ramon 和 Rebecca, Adele 和 Ali, Terry, Kevin, Becky, Temmy, Lil, Lin, Ron 和 David, Mike, Jessica 和 Bill，以及他们的家庭，因为他们给了他热情的支持，并且给他提供停留的地方以便他进行写作。

James H. Martin 在此也要感谢他的父母，是他们给了James 鼓励，并且允许他走上自然语言处理这条在当时看来似乎很古怪的学术道路。感谢他的博士论文指导老师Robert Wilensky，是Robert 给他机会在 Berkeley 开始学习自然语言处理。感谢 Peter Norvig，是 Peter 给他提供了许多正面的例子，并且指引他找到正确的途径。感谢 Rick Alterman，是 Rick 在关键和困难的时刻，给了他鼓励和勇气。感谢 Chuck Fillmore, George Lakoff, Paul Kay 和 Susanna Cumming，因为他们教过 James，使他懂得了语言学。感谢 Michael Main，因为当他不能担负系里的许多工作时，是 Michael 替他做了弥补。最后，James 还要感谢他的妻子 Linda，正是由于她多年的支持和耐心，James 才能够完成本书的写作。

科罗拉多大学的所在地 Boulder 是从事语音处理和语言处理的好地方。这里，我们要感谢我们在科罗拉多大学的同事们，他们的合作共事对我们的研究和教学有极大的影响。他们是：语言学系的 Alan Bell, Barbara Fox, Laura Michaelis 和 Lise Menn；计算机科学系的 Clayton Lewis, Gerhard Fischer, Mike Eisenberg, Mike Mozer, Liz Jessup 和 Andrzej Ehrenfeucht；心理学系的 Walter Kintsch, Tom Landauer 和 Alice Healy；口语理解中心的 Ron Cole, John Hansen 和 Wayne Ward；我们还要感谢我们在计算机科学系和语言学系的现在的和过去的学生 Marion Bond, Noah Coccaro, Michelle Gregory, Keith Herold, Michael Jones, Patrick Juola, Keith Vander Linden, Laura Mather, Taimi Metzler, Douglas Roland 和 Patrick Schone。

许多朋友仔细地阅读了本书并且进行了试教，他们提出了很多有益的建议。我们的同事也在百忙中抽出时间来阅读本书，提出了宝贵的意见和建议。这些意见和建议使得本书的很多部分都得到了改进，增色不少。我们应该深深地感激他们。他们是：Alan Bell, Bob Carpenter, Jan Daciuk, Graeme Hirst, Andy Kehler, Kemal Oflazer, Andreas Stolcke 和 Nigel Ward。我们的编辑 Alan Apt，系列丛书编辑 Peter Norvig 和 Stuart Russell，以及制作编辑 Scott DiSanno，都对于本书的设计和内容提出了很多有益的建议。我们还要感激许多朋友和同事们，他们阅读了本书的个别章节。对于他们的意见和建议中我们不明白的地方，他们还回答了我们很多的问题。我们还要感激科罗拉多大学Boulder校

部选这门课的学生们，加利福尼亚大学 Berkeley 分校 Daniel Jurafsky 班上的同学们，以及伊利诺伊大学 Urbana-Champaign 分校 LSA 暑期学院的学生们。此外还有：

Yoshi Asano, Todd M. Bailey, John Bateman, Giulia Bencini, Lois Boggess, Michael Braverman, Nancy Chang, Jennifer Chu-Carroll, Noah Coccaro, Gary Cottrell, Gary Dell, Jeff Elman, Robert Dale, Dan Fass, Bill Fisher, Eric Fosler-Lussier, James Garnett, Susan Garnsey, Dale Gerdemann, Dan Gildea, Michelle Gregory, Nizar Habash, Jeffrey Haemer, Jorge Hankamer, Keith Herold, Beth Heywood, Derrick Higgins, Erhard Hinrichs, Julia Hieschberg, Jerry Hobbs, Fred Jelinek, Liz Jessup, Aravind Joshi, Terry Kleeman, Jean-Pierre Koenig, Kevin Knight, Shalom Lappin, Julie Larson, Stephen Levinson, Jim Magnuson, Jim Mayfield, Lise Menn, Laura Michaelis, Corey Miller, Nelson Morgan, Christine Nakatani, Mike Neufeld, Peter Norvig, Mike O'Connell, Mick O'Donnell, Rob Oberbreckling, Martha Palmer, Dragomir Radev, Terry Regier, Ehud Reiter, Phil Resnik, Klaus Ries, Ellen Riloff, Mike Rosner, Dan Roth, Patrick Schone, Liz Shriberg, Richard Sproat, Subhashini Srinivasin, Paul Taylor, Wayne Ward, Pauline Welby, Dekai Wu 和 Victor Zue。

我们要感谢认知科学研究所以及计算机科学系和语言学系，谢谢他们对于我们多年的支持。我们还要感谢国家科学基金会，Daniel Jurafsky在写作本书期间还部分地得到了NSF CAREER IIS-9733067 的资助，Andy Kehler 也部分地得到了 NSF IIS-9619126 的资助。

目　　录

第1章　导论 1
1.1　语音与语言处理中的知识 1
1.2　歧义 3
1.3　模型和算法 4
1.4　语言、思维和理解 4
1.5　学科现状与近期发展 6
1.6　语音和语言处理简史 7
1.6.1　基础研究：20世纪40年代和20世纪50年代 7
1.6.2　两个阵营：1957年至1970年 8
1.6.3　四个范型：1970年至1983年 8
1.6.4　经验主义和有限状态模型的复苏：1983年至1993年 9
1.6.5　不同领域的合流：1994年至1999年 9
1.6.6　多重发现 10
1.6.7　心理学的简要注记 10
1.7　小结 11
1.8　文献和历史说明 11

第一部分　词汇的计算机处理

第2章　正则表达式与自动机 14
2.1　正则表达式 14
2.1.1　基本正则表达式模式 15
2.1.2　析取、组合与优先关系 18
2.1.3　一个简单的例子 18
2.1.4　一个比较复杂的例子 19
2.1.5　高级算符 20
2.1.6　正则表达式中的替换、存储器与ELIZA 21
2.2　有限状态自动机 22
2.2.1　用FSA来识别羊的语言 22
2.2.2　形式语言 25
2.2.3　另外的例子 26
2.2.4　非确定FSA 27
2.2.5　使用NFSA接收符号串 28
2.2.6　识别就是搜索 31
2.2.7　确定自动机与非确定自动机的关系 32

2.3 正则语言与 FSA 33
2.4 小结 34
2.5 文献和历史说明 35

第 3 章 形态学与有限状态转录机 36
3.1 英语形态学概观 37
3.1.1 屈折形态学 38
3.1.2 派生形态学 40
3.2 有限状态形态剖析 41
3.2.1 词表和形态顺序规则 41
3.2.2 用有限状态转录机进行形态剖析 44
3.2.3 正词法规则和有限状态转录机 48
3.3 把 FST 词表与规则相结合 50
3.4 与词表无关的 FST：PORTER 词干处理器 52
3.5 人是怎样进行形态处理的 53
3.6 小结 54
3.7 文献和历史说明 54

第 4 章 计算音系学与文本 – 语音转换 56
4.1 言语语音与语音标音法 57
4.1.1 发音器官 58
4.1.2 辅音：发音部位 60
4.1.3 辅音：发音方法 61
4.1.4 元音 62
4.1.5 音节 63
4.2 音位和音位规则 64
4.3 音位规则和转录机 65
4.4 计算音系学中的一些高级问题 68
4.4.1 元音和谐 68
4.4.2 模板式形态学 70
4.4.3 优选理论 70
4.5 音位规则的机器学习 74
4.6 TTS 中从文本映射到语音 75
4.6.1 发音词典 75
4.6.2 词典之外的查找：文本分析 77
4.6.3 基于有限状态转录机（FST）的发音词典 79
4.7 文本 – 语音转换中的韵律 82
4.7.1 韵律的音系学性质 82
4.7.2 韵律的语音和声学性质 83
4.7.3 语音合成中的韵律 83
4.8 人处理音位和形态的过程 84

4.9 小结 85
4.10 文献和历史说明 85

第5章 发音与拼写的概率模型 87
5.1 关于拼写错误 88
5.2 拼写错误模式 89
5.3 非词错误的检查 90
5.4 概率模型 90
5.5 把贝叶斯方法应用于拼写 92
5.6 最小编辑距离 95
5.7 英语的发音变异 97
5.8 发音问题研究中的贝叶斯方法 101
5.8.1 发音变异的决策树模型 104
5.9 加权自动机 105
5.9.1 从加权自动机计算似然度：向前算法 106
5.9.2 解码：Viterbi 算法 109
5.9.3 加权自动机和切分 112
5.9.4 用切分来进行词表的自动归纳 113
5.10 人类发音研究 114
5.11 小结 116
5.12 文献和历史说明 116

第6章 *N*元语法 118
6.1 语料库中单词数目的计算 119
6.2 简单的（非平滑的）*N*元语法 121
6.2.1 *N*元语法及其对训练语料库的敏感性 126
6.3 平滑 128
6.3.1 加 1 平滑 129
6.3.2 Witten-Bell 打折法 131
6.3.3 Good-Turing 打折法 134
6.4 回退 135
6.4.1 回退与打折相结合 136
6.5 删除插值法 137
6.6 拼写和发音的*N*元语法 138
6.6.1 上下文有关的错拼更正 138
6.6.2 发音模型的*N*元语法 139
6.7 熵 140
6.7.1 用于比较模型的交叉熵 142
6.7.2 英语的熵 143
6.8 小结 144
6.9 文献和历史说明 144

第7章 HMM与语音识别 146
7.1 语音识别的总体结构 147
7.2 隐马尔可夫模型概述 150
7.3 再谈Viterbi算法 152
7.4 先进的解码方法 157
7.4.1 A*解码算法 158
7.5 语音的声学处理 162
7.5.1 声波 162
7.5.2 怎样解释波形 163
7.5.3 声谱 163
7.5.4 特征抽取 166
7.6 声学概率的计算 166
7.7 语音识别系统的训练 169
7.8 用于语音合成的波形生成 171
7.8.1 音高和音延的修正 171
7.8.2 单元选择 172
7.9 人的语音识别 173
7.10 小结 174
7.11 文献和历史说明 175

第二部分 句法的计算机处理

第8章 词的分类与词类标注 178
8.1 大多数英语词的分类 179
8.2 英语的标记集 185
8.3 词类标注 186
8.4 基于规则的词类标注 187
8.5 随机词类标注 189
8.5.1 说明问题的一个例子 190
8.5.2 实际的HMM标注算法 191
8.6 基于转换的标注 193
8.6.1 怎样应用TBL规则 193
8.6.2 怎样学习TBL规则 194
8.7 其他问题 196
8.7.1 多重标记和多项词 196
8.7.2 未知词 197
8.7.3 基于类的*N*元语法 197
8.8 小结 198
8.9 文献和历史说明 199

第9章 英语的上下文无关语法 201
9.1 组成性 202
9.2 上下文无关规则和树 203
9.3 句子级的结构 207
9.4 名词短语 209
9.4.1 在中心名词前的成分 209
9.4.2 名词后的成分 210
9.5 并列关系 212
9.6 一致关系 212
9.7 动词短语和次范畴化 214
9.8 助动词 216
9.9 口语的句法 216
9.9.1 不流畅现象 217
9.10 语法等价与范式 218
9.11 有限状态语法和上下文无关语法 218
9.12 语法和人的语言处理 219
9.13 小结 220
9.14 文献和历史说明 221
第10章 基于上下文无关语法的剖析 223
10.1 剖析就是搜索 224
10.1.1 自顶向下剖析 225
10.1.2 自底向上剖析 225
10.1.3 自顶向下剖析与自底向上剖析的对比 227
10.2 基本的自顶向下剖析 227
10.2.1 增加自底向上过滤 230
10.3 基本的自顶向下剖析的问题 232
10.3.1 左递归 232
10.3.2 歧义 233
10.3.3 子树的重复剖析 236
10.4 Earley 算法 238
10.4.1 预测 240
10.4.2 扫描 240
10.4.3 完成 240
10.4.4 示例 240
10.4.5 从线图中检索剖析树 242
10.5 有限状态剖析方法 243
10.6 小结 246
10.7 文献和历史说明 246

第 11 章　特征与合一 248
11.1　特征结构 249
11.2　特征结构的合一 251
11.3　语法中的特征结构 255
11.3.1　一致关系 256
11.3.2　中心语特征 258
11.3.3　次范畴化 259
11.3.4　其他词类的次范畴化 262
11.3.5　长距离依存关系 263
11.4　合一的实现 264
11.4.1　合一的数据结构 264
11.4.2　合一算法 266
11.5　带有合一约束的剖析 270
11.5.1　把合一结合到 Earley 剖析器中 270
11.5.2　复制的必要性 274
11.5.3　合一剖析 275
11.6　类型与继承 277
11.6.1　类型的扩充 279
11.6.2　合一的其他扩充 280
11.7　小结 280
11.8　文献和历史说明 280

第 12 章　词汇化剖析与概率剖析 282
12.1　概率上下文无关语法 283
12.1.1　PCFG 的概率 CYK 剖析 286
12.1.2　PCFG 概率的学习 287
12.2　PCFG 的问题 288
12.3　概率词汇化的 CFG 289
12.4　依存语法 293
12.4.1　范畴语法 295
12.5　人的剖析 295
12.6　小结 300
12.7　文献和历史说明 300

第 13 章　语言的复杂性 302
13.1　Chomsky 层级 303
13.2　怎么判断一种语言不是正则的 304
13.2.1　抽吸引理 305
13.2.2　英语和其他自然语言是正则语言吗 307
13.3　自然语言是上下文无关的吗 309
13.4　计算复杂性和人的语言处理 311

13.5 小结 314
13.6 文献和历史说明 315

第三部分 语义的计算机处理

第 14 章 意义的表示法 318
14.1 意义表示的计算要求 319
14.1.1 可能性验证 320
14.1.2 无歧义表示 320
14.1.3 规范形式 321
14.1.4 推论与变元 322
14.1.5 表达能力 323
14.2 语言的意义结构 323
14.2.1 谓词论元结构 323
14.3 一阶谓词演算 324
14.3.1 FOPC 基础 325
14.3.2 FOPC 的语义 326
14.3.3 变量和逻辑量词 327
14.3.4 推论 329
14.4 某些与语言学相关的概念 330
14.4.1 范畴 330
14.4.2 事件 331
14.4.3 时间表示 333
14.4.4 体 336
14.4.5 信念表示 338
14.4.6 缺陷 340
14.5 有关的表示方法 341
14.6 意义的其他表示方法 341
14.6.1 作为行动的意义 341
14.6.2 作为真值的意义 342
14.7 小结 342
14.8 文献和历史说明 343

第 15 章 语义分析 344
15.1 句法驱动的语义分析 344
15.1.1 给上下文无关语法规则扩充语义 346
15.1.2 量词辖域和复杂项的转译 351
15.2 给英语片断附加语义分析 352
15.2.1 句子 352
15.2.2 名词短语 353
15.2.3 动词短语 355
15.2.4 介词短语 357

15.3 把语义分析结合到 Earley 剖析中 359
15.4 惯用语和组成性 360
15.5 鲁棒的语义分析 361
15.5.1 语义语法 361
15.5.2 信息抽取 363
15.6 小结 367
15.7 文献和历史说明 368
第 16 章 词汇语义学 370
16.1 词位及其涵义之间的关系 371
16.1.1 同形关系 371
16.1.2 多义关系 373
16.1.3 同义关系 375
16.1.4 上下位关系 376
16.2 WORDNET：词汇关系信息库 377
16.3 词的内在结构 380
16.3.1 题元角色 380
16.3.2 选择限制 385
16.3.3 基元分解 388
16.3.4 语义场 390
16.4 语言的创造性与词典 391
16.4.1 隐喻 391
16.4.2 换喻 392
16.4.3 隐喻和换喻的计算方法 392
16.5 小结 392
16.6 文献和历史说明 393
第 17 章 词义排歧与信息检索 394
17.1 基于选择限制的排歧 394
17.1.1 选择限制的局限性 395
17.2 鲁棒的词义排歧 396
17.2.1 机器学习方法 397
17.2.2 基于词典的方法 402
17.3 信息检索 403
17.3.1 向量空间模型 403
17.3.2 检索词加权 405
17.3.3 检索词的选择和创造 407
17.3.4 同形关系、多义关系和同义关系 408
17.3.5 改进用户的查询条件 408
17.4 信息检索的其他任务 409
17.5 小结 410
17.6 文献和历史说明 411

第四部分　语用的计算机处理

第 18 章　话语 414
18.1　所指判定 415
18.1.1　所指现象 417
18.1.2　同指的句法和语义约束 419
18.1.3　代词解释中的优先关系 422
18.1.4　代词判定算法 423
18.2　文本的连贯 430
18.2.1　现象 430
18.2.2　基于推理的判定算法 431
18.3　话语结构 436
18.4　所指和连贯的心理语言学研究 438
18.5　小结 441
18.6　文献和历史说明 441

第 19 章　对话与会话智能代理 443
19.1　什么使对话出现差别 444
19.1.1　话轮和话段 444
19.1.2　对话的共同基础 446
19.1.3　会话隐涵 447
19.2　对话行为 448
19.3　对话行为的自动解释 451
19.3.1　对话行为的计划推理解释 452
19.3.2　对话行为的基于提示的解释 456
19.3.3　要点 460
19.4　对话结构与连贯性 460
19.5　会话智能代理中的对话管理 464
19.6　小结 468
19.7　文献和历史说明 469

第 20 章　自然语言生成 471
20.1　语言生成导引 472
20.2　生成的体系结构 473
20.3　表层实现 474
20.3.1　系统语法 474
20.3.2　功能合一语法 477
20.3.3　要点 481
20.4　话语规划 481
20.4.1　文本说明图 481
20.4.2　修辞关系 483
20.4.3　小结 487

20.5 其他问题 487
20.5.1 微规划 487
20.5.2 词汇选择 488
20.5.3 生成系统评价 488
20.5.4 语音生成 488
20.6 小结 489
20.7 文献和历史说明 489
第 21 章 机器翻译 492
21.1 语言的相似性和差异性 494
21.2 转换模型 497
21.2.1 句法转换 498
21.2.2 词汇转换 499
21.3 中间语的思想：使用意义 500
21.4 直接转换 502
21.5 使用统计技术 504
21.5.1 流畅性的量化 505
21.5.2 忠实性的量化 505
21.5.3 输出的搜索 506
21.6 可用性与系统开发 506
21.7 小结 508
21.8 文献和历史说明 508
附录 A 正则表达式的算符 510
附录 B PORTER STEMMING 算法 511
附录 C 标记集 C5 和 C7 514
附录 D HMM 模型的训练：向前 – 向后算法 519
参考文献 525
术语表 577

第1章 导 论

Dave Bowman: Open the pod bay doors, HAL.
HAL: I'm sorry Dave, I'm afraid I can't do that.
Stanley Kubrick and Arthur C. Clarke,
screenplay of *2001: A Space Odyssey*[①]

在 Stanley Kubrick 的 2001 年的电影《太空奥德赛》中，有一台称为 HAL 的 9000 计算机，这台计算机具有20世纪最受人们认可的一些特征。影片中的HAL是一个具有高级语言处理能力并且能够说英语和理解英语的智能机器人（artificial agent），在影片情节的关键时刻，HAL 甚至能够进行唇读（reading lip），上面就是电影中的角色 Dave 先生请求智能机器人 HAL 打开太空舱的分离舱门（pod bay door），与 HAL 之间进行的一段对话。HAL 的作者 Arthur C. Clarke 曾经乐观地预言，到了一定的时候，我们就可以制造出像 HAL 这样的智能机器人。但是，现在我们离这样的预言还有多远呢？为了让HAL具有与语言相关的能力，我们还应该做些什么呢？我们认为，像HAL这样的机器人至少应该通过语言与人类进行交流。其中包括通过语音识别（speech recognition）和自然语言理解（natural language processing，当然包括唇读）来与人类沟通，通过自然语言生成（natural language generation）和语音合成（speech synthesis）来与人类交际。HAL 也应该能够进行信息检索（information retrieval，发现它所需要的文本资源在哪里）和信息抽取（information extraction，从文本资源中抽取它所需要的信息），并且能够进行推理（inference，根据已知的事实推出结论）。

尽管这些问题现在还没有完全解决，但是HAL需要的一些与语言相关的技术现在已经研制出来了，并且有一部分技术已经商品化。解决这样的问题以及其他类似的问题，是自然语言处理、计算语言学、语音识别与语音合成的主要研究内容。我们把它们统称为语音与语言的计算机处理（speech and language processing），或者简单地称为自然语言处理（natural language processing）。因此，自然语言处理也同时包括了语音处理的内容。本书的书名也相应地简称为《自然语言处理综论》，“综论”的含义，既包括语言处理，也包括语音处理。本书的目的在于向读者清楚地描述在21世纪开始之际，自然语言处理这门技术的现状。本书中将讨论的自然语言处理的应用范围，包括了智能机器人 HAL 所要具备的上述内容，此外还包括诸如拼写校正（spelling correction）、语法检查（grammar checking）、信息检索（information retrieval）和机器翻译（machine translation）等自然语言处理的其他有价值的研究领域。

1.1 语音与语言处理中的知识

我们把处理口语和书面语（统称为“语言”）的计算技术称为语音和语言处理，简称自然语言

---

① 这是本章的开场白，为了便于读者理解，给出译文如下：
Dave Bownman：HAL，请你打开太空舱的分离舱门。
HAL：对不起，Dave，我不能这样做。
Stanley Kubrick 和 Arthur C. Charke
2001 年电影剧本：《太空奥德赛》

处理。我们将看到，这是一个范围很广泛的定义，这个定义包括了从人人皆知的诸如词数计算、自动换行等简单技术，直到诸如在 Web 上的自动问答、实时的口语自动翻译等高级技术。

自然语言处理的这些应用与其他应用系统的区别是，自然语言处理要使用语言知识。例如，UNIX 的 wc 程序可以用来计算文本文件中的字节数、词数或行数。当我们用它来计算字节数和行数时，wc 只用于进行一般的数据处理。但是，当我们用它来计算一个文件中词的数目时，就需要关于“什么是一个词”的语言知识，这样，这个 wc 也就成为了一个自然语言处理系统。

当然，wc 只是一个非常简单的系统，它只具有极为有限的语言知识。像 HAL 这样有更复杂的语言能力的智能机器人将要求更广泛和更深刻的语言知识。我们只要读一读本章开头 HAL 和 Dave 进行的对话，就可以了解到此类更复杂的应用所需的语言知识的范围和种类。

为了确定 Dave 讲什么，HAL 必须能够分析它所接收的声音信号，并把 Dave 的这些信号复原成词的系列。与此相似，为了生成回答，HAL 必须把它的回答组织成词的系列，并且生成 Dave 能够识别的声音信号。要完成这两方面的任务，需要语音学（phonetics）和音系学（phonology）的知识，这样的知识可以帮助我们建立词如何在话语中发音的模型（见第 4 章和第 5 章）。

值得注意的是，与 Star Trek 的指令数据不同，HAL 还能够说出类似 *I'm* 和 *can't* 这样的缩约形式。产生并识别单词的这样或那样的变体（例如，识别 *doors* 是复数）要求形态学方面的知识，这些知识能够反映关于上下文中词的形态和行为的有关信息（见第 2 章和第 3 章）。

除了处理一个个的单词之外，HAL 还应该知道怎样分析 Dave 所提出的请求结构。这样的分析能够使 HAL 确定，Dave 说的话是关于要 HAL 采取某种行动的一个请求，这样的请求不同于下面关于陈述客观世界的简单命题，也不同于下面关于 *door* 的问话，它们是 Dave 请求的不同变体：

HAL, the pod bay door is open.（HAL，分离舱门是开着的。）

HAL, is the pod bay door open?（HAL，分离舱门是开着的吗？）

此外，HAL 还必须使用类似的结构知识把一个个的单词组织成符号串，构成它的回答。例如，HAL 必须知道，下面的单词序列对于 Dave 是没有意义的，尽管这个单词系列所包含的单词与它原来的回答中所包含的单词完全一样：

I'm I do, sorry that afraid Dave I'm can't.

这里所说的关于组词成句的知识，称为句法（syntax）。

显而易见，如果只知道 Dave 所说的话语的各个单词以及句法结构，并不能使 HAL 了解 Dave 提出的请求的实质。为了理解 Dave 的请求事实上是关于要求关闭分离舱门的一个命令，而不是讲关于当天中饭的菜单的事情，就要有复合词的语义的知识，有词汇语义学（lexical semantics）的知识以及如何把这样的复合词组成更大的语言意义实体的知识，即关于组合语义学（compositional semantics）的知识。

另外，尽管智能机器人 HAL 的行为还不十分熟练，但它也应该充分地懂得如何对 Dave 表示礼貌。例如，它不要简单地回答 *No* 或者 *No, I won't open the door*。HAL 首先用表示客气的话回答 *I'm sorry* 和 *I'm afraid*，然后委婉地说 *I can't*，而不是直截了当地（并且老老实实地）说 *I won't*[①]。这种礼貌和委婉语言的用法属于语用学（pragmatics）的研究领域。

最后，HAL 不是简单地无视 Dave 的请求，让门继续关着，而是对 Dave 最初的请求选择结构会话的方式来对待。HAL 在给 Dave 的回答中，正确地使用单词 that 来简单地表示会话中话段之间的共同部分。正确地把这样的会话组织成结构，需要话语规约（discourse convention）的知识。

① “我不愿关门。”这样的回答显得非常生硬、呆板和偏执。

总之，在复杂的语言行为中需要的语言知识可以分为6个方面：

- 语音学与音系学，研究语言的语音
- 形态学，研究词的有意义的组合
- 句法学，研究词与词之间的结构关系
- 语义学，研究意义
- 语用学，研究如何用语言来达成一定的目的
- 话语学，研究大于话段的语言单位

1.2 歧义

上述6个方面的语言知识存在着一个令人吃惊的事实：语音和语言计算机处理的绝大多数或者全部研究都可以看成是在其中某个层面上的消解歧义。如果我们想把某个意思输入计算机，但是存在着若干个不同的结构来表示这个意思，那么就说这样的输入是有歧义的。我们来考虑口语中的一个句子 *I made her duck*。这个句子可能有5个不同的意思（也可能还会更多），以下是歧义的若干实例：

I cooked waterfowl for her.（我给她烹饪鸭子） （1.1）

I cooked waterfowl belonging to her.（我烹饪属于她的鸭子） （1.2）

I created the (plaster?) duck she owns.（我把她的石膏(?)鸭子做了创新） （1.3）

I caused her to quickly lower her head or body.（我使她很快地把她的头或身体放低一些） （1.4）

I waved my magic wand and turned her into undifferentiated waterfowl. （1.5）

（我挥动魔杖把她变成了一只人们一点儿也看不出破绽的鸭子）

这些不同的意思都是由于歧义引起的。首先，*duck* 和 *her* 的词类在形态或句法上是有歧义的。*duck* 可以是动词或名词，而 *her* 可以是表示给予格的代词或表示所属格的代词。其次，*make* 在语义上是有歧义的，它的意思可以是 *create*（创造），也可以是 *cook*（烹饪）。最后，动词 *make* 还可以有不同的句法歧义。*make* 可以做及物动词，带直接宾语，如例句（1.2）；*make* 也可以做双及物动词，带两个宾语，如例句（1.5），表示把第一宾语（*her*）变成了第二个宾语（*duck*）；*make* 还可以带一个直接宾语和一个动词，如例句（1.4），表示使直接宾语（*her*）进行某个动作（*duck*）。此外，在口语的句子中，还可以有一种更为深刻的歧义，第一个词可以被理解为 *eye*，第二个词可以被理解为 *maid*。

这样，歧义就更加复杂了。在本书中，我们会经常介绍消解（resolve）这些歧义（或称排歧（disambiguate）的模型和算法。例如，使用词类标注（part-of-speech tagging）方法确定 *duck* 是名词还是动词。使用词义排歧（word sense disambiguation）方法确定 *make* 的意思是 *create*（创造）还是 *cook*（烹饪）。词类排歧和词义排歧是词汇排歧（lexical disambiguation）的两个主要内容。很多研究都可以纳入到词汇排歧的框架之内。例如，在文本-语音合成系统中，当读到单词 *lead* 的时候，必须判断这个 *lead* 是应该按照 *lead pipe* 中的 *lead* 读音，还是应该按照 *lead me on* 中的 *lead* 读音。此外还有句法排歧（syntactic disambiguation）。例如，当我们判断 *her* 和 *duck* 是属于不同的实体，如例句（1.1）或例句（1.4），还是属于同一个实体，如例句（1.2），这样的问题就属于句法排歧问题，可以通过概率剖析（probabilistic parsing）方法来解决。在上述例子中没有出现的一些歧义（例如，判断一个句子是陈述句还是疑问句），可以通过言语行为解释（speech act interpretation）方法来解决。

1.3 模型和算法

50年来的自然语言处理研究说明，前一节中所描述的那些知识可以使用数量有限的形式模型或理论来获得。值得庆幸的是，这些模型和理论都来自计算机科学、数学和语言学的工具，在这些领域受过训练的人对这样的工具一般都不会感到生疏。其中最重要的部分是状态机（state machine）、形式规则系统（formal rule system）、逻辑（logic）以及概率论（probability theory）和其他的机器学习工具。从熟知的计算范型出发，这样的模型本身就可以给出为数不多的算法。其中最重要的算法是状态空间搜索（state space search）算法和动态规划（dynamic programming）算法。

简单地说，状态机就是形式模型，形式模型应该包括状态、状态之间的转移以及输入表示等。这种基本模型的变体有确定的有限状态自动机（deterministic finite-state automata）、非确定的有限状态自动机（non-deterministic finite-state automata）和有限状态转录机（finite-state transducer），它们可以写到一个输出器中；另外还有加权自动机（weighted automata）、马尔可夫模型（Markov model）和隐马尔可夫模型（hidden Markov model），它们都包含一个概率组成成分。

与这些过程性模型紧密联系的模型是陈述性模型。在这些陈述性模型中，最重要的有正则语法（regular grammar）、正则关系（regular relation）、上下文无关语法（context-free grammar）、特征增益语法（feature-augmented grammar）以及这些文法的相应概率文法变体。状态机和形式规则系统是用于处理音系学、形态学和句法学的主要工具。

与状态机和形式规则系统相关联的最典型的算法就是搜索代表有关输入的假设的状态空间。例如，在语音识别中，对于输入的词搜索其音位系列的空间；在剖析中，对于输入的句子的正确句法分析搜索其树的空间，等等。在自然语言处理中经常使用的算法都是一些众所周知的图算法，例如，深度优先算法（depth-first search）以及最佳优先搜索算法（best-first）和A* 搜索算法（A* search）等试探性算法的变体。动态规划范型对于很多这样的方法的计算可循性是至关重要的，因为只有这样才能确保避免冗余的计算。

对于获取语言知识起着关键性作用的第三种模型是逻辑。我们将讨论一阶逻辑（first order logic），即谓词演算（predicate calculus），以及诸如特征结构、语义网络、概念依存等有关的形式化方法。在传统上，这些逻辑表达方法是处理语义学、语用学和话语分析等方面知识的选择工具（正如我们将会看到的，尽管在这些领域的应用越来越依赖于在音系学、形态学和句法学中所使用的简单机制，但逻辑表达方法也非常重要）。

概率论是获取语言知识的技术中的最后一个部分。其他的各种模型（状态机、形式系统和逻辑）都可以使用概率得到进一步提高。概率论的一个重要应用是解决前面我们讨论过的歧义问题：几乎所有语音处理和语言处理问题都可以这样来表述："对于某个歧义的输入给出N个可能性，选择其中概率最高的一个。"

概率论的另一个优点是它同时也是一种机器学习（machine learning）的模型。机器学习的研究目前主要集中在探索自动学习前面描述过的自动机、规则系统、搜索试探、分类等的各种表达。这样的系统可以使用大规模的语料库来训练，特别是当我们还没有很好的因果关系模型时，机器学习系统可以用来作为一种有效的模型技术。贯穿全书，我们都要描述机器学习的算法。

1.4 语言、思维和理解

如果计算机能够像我们这样熟练地处理语言，那么，这就意味着计算机已经达到了真正的智能机器的水平。这种信念基于这样的事实：语言总是与我们的认知能力纠缠在一起。Alan Turing(1950)

是第一个认识到计算机与认知能力之间有着如此密切关系的科学家。在他的一篇著名的论文中，Turing提出了图灵测试（Turing Test）的想法。Turing在他的论文的开头就指出，关于什么是机器思维的问题无法回答，因为“机器”（*machine*）与“思维”（*think*）这两个术语本身就是含糊不清的。因此，他建议做一个游戏来进行测试。在游戏中，计算机对于语言的使用情况可以用来作为判断计算机是否能进行思维的依据。如果计算机在游戏中获胜，那么就可以判断计算机具有智能。

在Turing的游戏中有三个参加者：两个人和一台计算机。其中的一个人充当提问者的角色，他要使用电传打字机向另外两个参加者提出一系列问题，根据这两个参加者的回答判断哪一个回答是计算机做出的。计算机的任务是尽量设法来愚弄提问者，对于提问者的问题，尽量做出像人一样的回答，设法使提问者相信它真的是一个人。第二个参加游戏的人则尽量设法使提问者相信第三个参加者是计算机，只有他和提问者才是人。

下面是Turing在他的论文中所描述的一个交互过程。显而易见，计算机要模拟人，并不能要求它成为一个无所不知的专家。

Q：Please write me a sonnet on the topic of the Forth Bridge.

A：Count me out on this one. I never could write poetry.

Q：Add 34957 to 70764.

A：（30 s之后给出回答）105621.

第一个提问要求写一首商籁体的诗，这显然是高智能的工作，因此，回答是拒绝写这样的诗歌。第二个提问要求做5位数的加法，需要30 s的思索之后才能够给出正确的回答。这样的问题不论对于人来说，还是对于计算机来说，都并非易事。Turing预言，大约到20世纪末，具有10 GB存储量的计算机经过5 min的提问之后，将会有30%的可能性愚弄提问者，使提问者分辨不清它究竟是人还是计算机。如果你也想像计算机那样愚弄别人，也可能说不清楚究竟采用什么样的严格标准去愚弄，才可以使别人感到困惑。然而，对于Turing来说，关键的问题是，只要使用人的语言就足以作为测定智能的可操作标准了。

令人惊讶的是，1966年就有人设计了一个称为ELIZA的程序来验证Turing的思想（Weizenbaum，1966）。ELIZA是一个早期的自然语言处理系统，这个系统能与用户进行有一定限制的对话。下面是ELIZA系统模拟心理医生的一段对话：

用户$_1$： You are like my father in some ways.

ELIZA$_1$： WHAT RESEMBLANCE DO YOU SEE

用户$_2$： You are not very aggressive but I think you don't want me to notice that.

ELIZ$_2$： WHAT MAKES YOU THINK I AM NOT AGGRESSIVE

用户$_3$： You don't argue with me.

ELIZA$_3$： WHY DO YOU THINK I DON'T ARGUE WITH YOU

用户$_4$： You are afraid of me.

ELIZA$_4$： DOES IT PLEASE YOU TO BELIEVE I AM AFRAID OF YOU

在第2章中将会详细讨论到，ELIZA只是使用模式匹配的方法进行输入并把它转换成适当的输出，这是一个非常简单的系统。事实上ELIZA并没有必要懂得如何模拟心理医生，它只是使用简单的模式匹配就取得了成功。正如系统的设计人Weizenbaum所说的，在ELIZA系统中，听话者的所作所为就好像他们对于周围的世界一无所知。

ELIZA与Turing思想的深刻联系在于，很多与ELIZA进行过交互的人都相信，ELIZA确实理解了他们所说的话以及他们所提出的问题。Weizenbaum(1976)指出，甚至在把程序的操作过程向人们做了解释之后，仍然有不少人继续相信ELIZA的能力。近年来，人们又以不同的形式重复着Weizenbaum的工作。自1991年以来，在Loebner奖的比赛中，人们试图设计各种计算机程序来做Turing测试。尽管这些比赛的科学意义不是很大，但这些比赛的成绩说明，哪怕是很粗糙的程序有时也会愚弄人们的判断力(Shieber, 1994)。哲学家和人工智能研究者对于Turing测试究竟是否适合用来测试智能的争论已经持续很多年了，但上述比赛的结果并没有平息这样的争论(Searle, 1980)。

就本书的目的而言，这样的比赛结果与计算机究竟能否思维，或者计算机究竟能否理解自然语言的问题是风马牛不相及的。更为重要的是，在社会科学中的有关研究证实了Turing在同一篇文章中的预见：

> 然而，我相信，在本世纪末，词语的使用和教育的舆论将大大地改变，使我们有可能谈论机器思维而不致遭到别人的反驳。

现在已经清楚，无论人们相信什么，无论人们是否知道了计算机的内部工作情况，他们都在谈论计算机，并且都在与计算机进行着交互，把计算机当成一个社会实体。人们把计算机当成人一样对待，他们要对它讲礼貌，把它当成团队中的成员，并且期望计算机能够理解人们的需求，能够非常自然地与人们进行交互。例如，Reeves and Nass(1996)发现，当计算机要求人们来评价计算机的所作所为好不好时，人们要针对不同计算机提出的同样的问题做出更多的正面回答。人们似乎担心他们给计算机的回答不够礼貌。Reeves和Nass在另外的实验中还发现，如果计算机对人们说一些奉承的话，人们给计算机的评价也就会高一些。给出这样的一些预设，使用语音和语言的系统就能够给众多的用户在很多应用方面提供更自然的交互界面。这些情况导致了一个称为会话代理（conversational agent）的研究焦点，所谓会话代理就是通过会话进行交际的计算机人造实体。

1.5 学科现状与近期发展

> We can only see a short distance ahead, but we can see plenty there that needs to be done.①
>
> Alan Turing

现在语音和语言处理正处于激动人心的时刻。目前已经商品化的机器人语音识别系统，互联网的兴起，都使得语音和语言处理的应用成为了当前的热门话题，有时甚至过分地提出了一些令人激动的可能的应用领域。我们想在这里列举出该学科当前的一些应用项目，并提出该学科近期发展的一些可能的方面。

加拿大的计算机程序能够接受每天的天气预报数据，然后自动生成天气预报报告，不必经过进一步编辑就可以用英语和法语公布(Chandioux, 1976)。

Systran的Babel Fish机器翻译系统每天可以处理Alta Vista搜索引擎的100万个翻译请求。

到马萨诸塞州坎布里奇市访问的一个访问者用口语问计算机在什么地方可以吃饭，系统查询了一个关于当地饭店的数据库之后，给出了相关信息作为回答(Zue et al., 1991)。

这些仅仅是当前语音和语言处理技术的几个为数不多的事实。下面我们再举出更多的实例来说明当前世界上在语音和语言处理的研究和开发实验室中进行的一些探索性应用研究。

让计算机阅读数百篇典型的大学生论文并给这些论文打分，而计算机的打分结果与人的打分结果几乎毫无区别，难以分辨(Landauer et al., 1997)。

---

① 译文：尽管我们只能向前看很短的距离，但是我们能看清楚需要做什么事情。

让计算机充当自动阅读家庭教师，帮助改善阅读能力。它能教小孩阅读故事，当阅读人要求阅读或者出现阅读错误时，计算机能使用语音识别器来进行干预(Mostow and Aist, 1999)。

给计算机装上图像识别系统，它就可以观看一段足球比赛的录像，并且用自然语言报告比赛的情况(Wahlster, 1989)。

对于有言语或交际障碍的残疾人，计算机能预见下面将要出现的词语，给他们做出提示，或者当他们说话时帮助在词语方面进行扩充，使残疾人能完整地说出简洁的话语(Newell et al., 1998; McCoy et al., 1998)。

1.6 语音和语言处理简史

在历史上，语音和语言处理曾经在计算机科学、电子工程、语言学和心理认知语言学等不同的领域分别进行研究。之所以出现这种情况，是由于语音和语言处理包括了一系列性质不同而又彼此交叉的学科，它们是：语言学中的计算语言学（computational linguistics）、计算机科学中的自然语言处理（natural language processing）、电子工程中的语音识别（speech recognition）、心理学中的计算心理语言学（computational psycholinguistics）。在本节中，我们将把在语音和语言处理中的这些不同历史线索进行总结性的说明。但是，本节只是提供一个梗概，更详细的介绍可以参阅有关章节，以便进一步了解相应的领域及其术语。

1.6.1 基础研究：20世纪40年代和20世纪50年代

这个领域的研究最早可以追溯到第二次世界大战结束时那个充满理智的时代，那个时代刚发明了计算机。从20世纪40年代到20世纪50年代末期，有两项基础性的研究值得注意：一项是自动机的研究，另一项是概率或信息论模型的研究。

20世纪50年代提出的自动机理论来源于Turing(1936)的算法计算模型，这种模型被认为是现代计算机科学的基础。Turing的工作首先导致了McCulloch-Pitts的神经元（neuron）理论(McCulloch and Pitts, 1943)。一个简单的神经元模型就是一个计算的单元，它可以用命题逻辑来描述。接着，Turing的工作导致了Kleene(1951, 1956)关于有限自动机和正则表达式的研究。Shannon(1948)把离散马尔可夫过程的概率模型应用于描述语言的自动机。Chomsky(1956)从Shannon的工作中吸取了有限状态马尔可夫过程的思想，首先把有限状态自动机作为一种工具来刻画语言的语法，并且把有限状态语言定义为由有限状态语法生成的语言。这些早期的研究工作产生了形式语言理论（formal language theory）这样的研究领域，采用代数和集合论把形式语言定义为符号的序列。Chomsky(1956)在研究自然语言时首先提出了上下文无关语法，但是Backus(1959)和Naur et al.(1960)在描述ALGOL程序语言的工作中也独立地发现了这种上下文无关语法。

这个时期的另一项基础研究工作是用于语音和语言处理的概率算法的研制，这是Shannon的另一个贡献。Shannon把通过诸如通信信道或声学语音这样的媒介传输语言的行为比喻为噪声信道（noisy channel）或解码（decoding）。Shannon还借用热力学的术语“熵”（entropy）来作为测量信道的信息能力或语言的信息量的一种方法，并且他用概率技术首次测定了英语的熵。

在这个时期还研究了声谱(Koenig et al., 1946)。声谱和实验语音学的基础研究为而后语音识别的研究奠定了基础。这导致了20世纪50年代第一个机器语音识别器的研制成功。1952年，Bell实验室的研究人员建立了一个统计系统来识别由一个单独的说话人说出的10个任意数字(Davis et al.,1952)。该系统存储了10个依赖于说话人的模型，它们粗略地代表了数字的前两个元音的共振峰。Bell实验室的研究人员采用选择与输入具有最高相关系数模式的方法，达到了97%~99%的准确率。

1.6.2 两个阵营：1957年至1970年

在20世纪50年代末期到20世纪60年代初期，语音和语言处理明显地分成两个阵营：一个是符号派（symbolic），一个是随机派（stochastic）。

符号派的工作可以分为两个方面。一方面是20世纪50年代后期以及20世纪60年代初期和中期Chomsky等的形式语言理论和生成句法研究，很多语言学家和计算机科学家的剖析算法研究，早期的自顶向下和自底向上算法研究以及后期的动态规划研究。最早的完整剖析系统是Zelig Harris的“转换与话语分析课题”（Transformation and Discourse Analysis Project，简称TDAP）。这个剖析系统于1958年6月至1959年7月在宾夕法尼亚大学研制成功(Harris, 1962)[①]。另一方面是人工智能的研究。在1956年夏天，John McCarthy，Marvin Minsky，Claude Shannon和Nathaniel Rochester等学者聚集在一起组成了一个为期两个月的研究组，讨论关于他们称为“人工智能”（Artificial Intelligence，简称AI）的问题。尽管有少数AI研究者着重于研究随机算法和统计算法（包括概率模型和神经网络），但是大多数的AI研究者着重研究推理和逻辑问题。典型的例子是Newell和Simon关于“逻辑理论家”（Logic Theorist）和“通用问题解答器”（General Problem Solver）的研究工作。早期的自然语言理解系统都是按照这样的观点建立起来的。这些简单的系统把模式匹配和关键词搜索与简单试探的方法结合起来，进行推理和自动问答，它们都只能在某一个领域内使用。在20世纪60年代末期，学者们又研制了更多的形式逻辑系统。

随机派主要是一些来自统计学专业和电子学专业的研究人员。在20世纪50年代后期，贝叶斯方法开始用于解决最优字符识别问题。Bledsoe and Browning(1959)建立了用于文本识别的贝叶斯系统，该系统使用了一部大词典，计算词典的单词中所观察到的字母系列的似然度，把单词中每个字母的似然度相乘，就可以求出字母系列的似然度来。Mosteller and Wallace(1964)用贝叶斯方法来解决在The Federalist（联邦主义者）文章中的原作者的分布问题。

20世纪60年代出现了基于转换语法的第一个人类语言计算机处理的可严格检测的心理模型，并且还出现了第一个联机语料库：Brown美国英语语料库，该语料库包含100万单词的语料，样本来自不同文体的500多篇书面文本，涉及的文体有新闻、中篇小说、写实小说、科技文章等。这些语料是布朗大学在1963年至1964年收集的(Kučera and Francis, 1967; Francis, 1979; Francis and Kučera, 1982)。王士元（William S. Y. Wang）在1967年建立了DOC（Dictionary on Computer），这是一部联机的汉语方言词典。

1.6.3 四个范型：1970年至1983年

下一个时期在语音和语言处理研究中取得了令人瞩目的进展，一系列的研究堪称这一研究领域的典范，我们称之为“范型”（paradigm）。这些范型的研究在语音和语言处理领域的研究中仍然处于支配地位。

在这个时期，随机范型（stochastic paradigm）在语音识别算法的研制中起着重要的作用，其中特别重要的是隐马尔可夫模型和比喻为噪声信道与解码的模型。这些模型是分别独立由两支队伍研制的。一支是Jelinek，Bahl，Mercer以及IBM的Thomas J. Watson研究中心的研究人员，另一支是卡内基梅隆大学的Baker等人，Baker受到了普林斯顿防护分析研究所的Baum和他的同事们的工作的影响。AT&T的贝尔实验室也是语音识别和语音合成的中心之一，详情可参阅Rabiner and Juang (1993)对这方面的工作的全面描述。

---

① 这个系统最近又重新建立起来，Joshi and Hopely(1999)以及Karttunen(1999)对这个系统进行了描述，他们指出，该系统的剖析本质上是用层叠式的有限状态转录机实现的。

基于逻辑的范型(logic-based paradigm)起始于Colmerauer和他的同事们关于Q系统(Q-system)和变形文法（ metamorphosis grammar ）的工作(Colmerauer, 1970, 1975)。Colmerauer是Prolog语言的先驱者。定子句文法（ Definite Clause Grammar ）(Pereira and Warren, 1980)也是在基于逻辑的范型方面的早期工作之一。Kay(1979)对于功能语法的研究，稍后Bresnan and Kaplan(1982)在词汇功能语法（ Lexical Function Grammar，简称LFG ）方面的工作，都是特征结构合一（ feature structure unification ）研究方面的重要成果。

这个时期的自然语言理解起始于Terry Winograd的SHRDLU系统，这个系统能够模拟一个嵌入玩具积木世界的机器人的行为(Winograd, 1972a)。该系统的程序能够接受自然语言的书面指令，例如“*Move the red block on top of the smaller green one*”（请把绿色的小积木块移动到红色积木块的上端），从而指挥机器人摆弄玩具积木块。迄今为止，我们还没有看到如此复杂和精妙的系统。这个系统还首次尝试建立基于Halliday系统语法的全面的英语语法（在当时看来是全面的）。Winograd的模型还清楚地说明，句法剖析也应该重视语义和话语的模型。Roger Schank和他在耶鲁大学的同事和学生们（经常被称为耶鲁学派）建立了一些语言理解程序。这些程序构成了一个系列，他们重点研究诸如脚本、计划和目的这样的人类概念知识以及人类记忆机制(Schank and Albelson, 1977; Schank and Riesbeck, 1981; Cullingford, 1981; Wilensky, 1983; Lehnert, 1977)。他们的工作经常使用基于网络的语义学理论(Quillian, 1968; Norman and Rumelhart, 1975; Schank, 1972; Wilks, 1975c, 1975b; Kintsch, 1974)，并且在他们的表达方式中(Simmons, 1973)开始引进Fillmore关于格角色的概念(Fillmore, 1968)。

基于逻辑的范型和基于自然语言理解的范型还可以在系统中融合起来。例如，LUNAR问答系统(Woods, 1967, 1973)是一个自然语言理解系统，在该系统中就使用了谓词逻辑来进行语义解释。

话语模型范型（ discourse model paradigm ）集中探讨了话语研究中的4个关键领域。Grosz和她的同事们研究了话语中的子结构（ substructure ）和话语焦点(Grosz, 1977a; Sidner, 1983)；一些研究者开始研究自动参照消解(automatic reference resolution) (Hobbs, 1978)。在基于逻辑的言语行为研究中，建立了“信念-愿望-意图”的框架，即BDI（ Belief-Desire-Intention ）的框架(Perrault and Allen, 1980; Cohen and Perrault, 1979)。

1.6.4 经验主义和有限状态模型的复苏：1983年至1993年

在1983年至1993年期间，语音和语言处理的研究又回到了20世纪50年代末到20世纪60年代初几乎被否定的有限状态和经验主义这两种模型上。这两种模型之所以出现这种复苏，其部分原因在于过去Chomsky对于Skinner的“言语行为”（ Verbal Behavior ）的很有影响的评论(Chomsky, 1959b)在这时遭到了理论上的反对。第一种模型是有限状态模型，由于Kaplan and Kay(1981)在有限状态音系学和形态学方面的工作，以及Church(1980)在句法的有限状态模型方面的工作，这种模型又重新得到注意。本书自始至终都会讨论与有限状态模型有关的工作。

在这个时期的第二个倾向是所谓的“重新回到经验主义”；这里特别值得注意的是，语音和语言处理的概率模型的提出，这样的模型受到IBM的Thomas J. Watson研究中心的语音识别概率模型的强烈影响。这些概率模型和其他数据驱动的方法还传播到了词类标注、句法剖析、附着歧义的判定以及从语音识别到语义学的联接主义方法的研究中去。

在这个时期，自然语言生成的研究也取得了引人瞩目的成绩。

1.6.5 不同领域的合流：1994年至1999年

在20世纪的最后5年中，语音和语言处理这个领域发生了很大的变化。这主要表现在三个方

面。首先，概率和数据驱动的方法几乎成为了自然语言处理的标准方法。句法剖析、词类标注、参照消解和话语处理的算法全都开始引入概率，并且采用从语音识别和信息检索中借用的评测方法。其次，由于计算机的速度和存储量的增加，使得在语音和语言处理的一些子领域，特别是在语音识别、拼写检查、语法检查这些子领域，有可能进行商品化的开发。语音和语言处理的算法开始被应用于增强交替通信（Augmentative and Alternative Communication，简称 AAC）中。最后，Web 的发展使得进一步加强基于语言的信息检索和信息抽取的需求变得更加突出。

1.6.6 多重发现

尽管我们在这里只是简单地回顾了语音和语言处理的发展历史，但已经可以看出，在许多场合下同样的思想可能会多次在不同的地方独立地被发现。在本书中我们将讨论几个这种“多重发现”。例如，动态规划在序列比较中的应用被 Viterbi，Vintsyuk，Needleman 与 Wunsch，Sakoe 与 Chiba，Sankoff 与 Reichert 等，以及 Wagner 与 Fischer 分别独立地提出过（见第 5 章和第 7 章）。语音识别中的 HMM 模型和噪声信道模型被 Baker，Jelinek，Bahl 和 Mercer 分别独立地提出过（见第 7 章）。上下文无关语法被 Chomsky，Backus 与 Naur 分别独立地提出并研究过（见第 9 章）。瑞士德语中存在非上下文无关句法的证明被 Huybregts 和 Shieber 分别独立地研究过（见第 13 章）。把合一运算应用于语言处理被 Colmerauer 等人和 Kay 分别独立地提出过（见第 11 章）。

这些多重发现难道是令人惊讶的巧合吗？科学社会学者 Robert K. Merton(1961)反对巧合的说法，他指出：

> 一切科学发现，包括那些从表面上看来似乎独一无二的科学发现，原则上都是多重的。

显而易见，历史上确实存在着许多众所周知的多重的科学发现和科学发明的事例。Ogburn and Thomas(1922)曾经列出了一个多重发现表，其中列举了很多事例。例如，Leibnitz 和 Newton 分别发明了微积分；Wallace 和 Darwin 分别研究了自然选择的理论；Gray 和 Bell 分别发明了电话[①]。然而，Merton 根据进一步的事例提出的这样的假设：多重发现是一个规律，而不是偶然的例外。很多公认的独一无二的发现原来是过去没有公布过的工作或没有被接受的工作的再发现。他的人类学方法论的观点中还有一个更有力的论点，这种论点认为，科学家本身总是在把多重发现作为准则的假定下从事研究工作的。因此，科学生活的很多方面都是为了帮助科学家避免被别人“抢先得到”他的发现而设计的。例如，在给杂志提交论文时要注明日期；在科学研究的记录中要仔细地注明日期；及时周转预研报告或技术报告。

1.6.7 心理学的简要注记

本书的许多章节中都有关于人类对于语言处理的心理学研究的简要说明。显而易见，理解人类的语言处理，无论对于这种研究本身还是对于认知科学的整个领域的一部分来说，都是极为重要的科学探索目标。而且，理解人类对于自然语言的处理，经常能够帮助我们建立更好的语言处理的机器模型。但是，这样的看法似乎与人们通常对此的认识相矛盾，因为通常认为，对于自然的算法的直接模仿在工程应用中没有多大作用。其论据是，如果我们毫无差异地复制自然，并不能导致工程技术上的成功。例如，如果飞机也像鸟一样地摆动它的机翼，这样的设计并不适用于飞机制造工程，因为具有固定机翼的飞机在工程技术上是更为成功的解决方法。然而，语言并不是航空。模仿自然

① 一般情况下认为，Ogburn 和 Thomas 注意到了多重发现的普遍存在意味着文化的环境是科学发现的决定性因素，而个人的天分并不是科学发现的决定性因素。然而，Merton 半开玩笑地援引 19 世纪以及 19 世纪以前的材料说明，这样的说法本身也是一种多重发现。

只是有时对于航空有用（因此飞机才有了机翼），但如果我们试图解决以人类为中心的问题，模仿就会特别有用。飞机的飞行与鸟的飞行有着不同的目的，但语音识别系统的目的与法院的记录员每天的工作目的却非常一致，两者的目的都是把口语的对话转写下来。由于人类在这个方面已经做得非常成功，我们就可以从原先的解决方法中受到启示，来研究语音识别的方法。由于语音和语言处理系统的一个重要应用是人机交互，所以模拟人类习以为常的解决方法肯定非常有效。

1.7 小结

本章介绍语音和语言处理领域。下面是本章的要点：

- 理解语音和语言处理研究的一个好办法是考察怎样创造2001年电影剧本《太空奥德赛》中的HAL这样的智能代理。
- 语音和语言处理技术与音系学、语音学、形态学、句法学、语义学、语用学和话语分析等不同平面上的语言知识的形式模型和形式表示方法有着密切的依赖关系。使用包括状态机、形式规则系统、逻辑等在内的形式模型以及概率论可以获取这样的知识。
- 语音和语言处理的基础是计算机科学、语言学、数学、电子工程和心理学。在语音和语言处理中要使用这些学科的标准框架中的某些算法。
- 语言和思维之间的密切联系使语音和语言处理技术成为关于智能机器辩论的中心议题。关于人类怎样与复杂媒体交互的研究表明，语音和语言处理技术在今后技术的发展过程中将起着至关重要的关键性作用。
- 语音和语言处理的革命性的应用目前已经在现实世界的周围呈现出来了。语音识别的最新进展和万维网的建设将进一步引导出这种技术的更加丰富多彩的实际应用前景。

1.8 文献和历史说明

语音和语言处理的各个分支领域的研究成果发表在很多会议论文集和杂志上。这些会议和杂志的中心内容都集中在计算语言学和自然语言处理两个方面，主要与美国计算语言学会（Association for Computational Linguistics，简称ACL）和它的欧洲伙伴（即，欧洲计算语言学会（European Association for Computational Linguistics，简称EACL）、国际计算语言学会议（International Conference on Computational Linguistics，简称COLING）有关系。ACL和EACL的年会论文集和每两年一次的COLING会议是该领域研究工作的首要论坛。有关的会议还有每两年一次的应用自然语言处理会议（Applied Natural Language Processing，ANLP）和自然语言处理中的经验方法会议（Empirical Methods in Natural Language Processing，EMNLaP）。*Computational Linguistics*（计算语言学）杂志是这个领域的最早的出版物，尽管该杂志倾向于理论和语言学方面。*Natural Language Engineering*（自然语言工程）杂志则倾向于语音和语言研究的很多实际应用方面。

语音识别、理解和合成的研究成果发表于每两年一次的口语处理国际会议上（International Conference on Spoken Language Processing，简称ICSLP），这个会议与欧洲语音通信和技术会议（European Conference on Speech Communication and Technology，简称EUROSPEECH）交替召开。IEEE国际声学、言语和信号处理会议（IEEE International Conference on Acoustics, Speech and Signal Processing，IEEE ICASSP）每年召开一次，是美国声学学会的会议。语音处理方面的杂志主要有*Speech Communication*（语音通信）、*Computer Speech and Language*（计算机语音与语言）以及*IEEE Transactions on Pattern Analysis and Machine Intelligence*（IEEE模式分析与机器智能学报）。

从人工智能角度研究语言处理的成果发表于美国人工智能学会（American Association for Artificial Intelligence，简称AAAI）的年会以及两年一次的国际人工智能联合会议（International Joint Conference on Artificial Intelligence，简称IJCAI）。下面的人工智能出版物定期发表有关语音和语言处理的成果：*Artificial Intelligence*（人工智能），*Computational Intelligence*（计算机智能），*IEEE Transactions on Intelligent Systems*（IEEE智能系统学报）以及*Journal of Artificial Intelligence Research*（人工智能研究杂志）。语言认知模型的研究成果主要发表于认知科学学会年会及其杂志*Cognitive Science*（认知科学）上。ARPA还召开了一种很有影响的序列讨论会，讨论会的论文都是特别邀请的，讨论会的名称为*DARPA Speech and Natural Language Processing Workshop*（DARPA语音和自然语言处理讨论会）或*ARPA Workshop on Human Language Technology*（ARPA人类语言技术讨论会）。

有关语音和语言处理的各方面的教材很多。*Foundations of Statistical Language Processing*[①](Manning and Schütze, 1999)重点讲述标注、剖析、排歧、搭配等方面的统计模型。*Statistical Language Learning*(Charniak, 1993)介绍了相似的内容，尽管内容有些陈旧，篇幅也比较短，但是通俗易懂，是一本入门读物。*Natural Language Understanding*[②](Allen, 1995)从人工智能的角度讲述了语言处理的各方面的材料，覆盖面比较广。*Natural Language Processing in Lisp/Prolog*(Gazdar and Mellish, 1989)特别讲述了自动机、剖析、特征和合一等方面的内容。*Prolog and Natural-Language Analysis*(Pereira and Shieber, 1987)介绍了基于Prolog的剖析和解释技术。*Artificial Intelligence: A Modern Approach*(Russell and Norvig, 1995)是人工智能的导论性读物，其中包括了自然语言处理的章节。*Mathematical Methods in Linguistics*(Partee et al., 1990)全面地介绍了数理语言学。*Survey of the State of the Art in Human Language Technology*(Cole, 1997)概述了语音和语言处理的所有领域的情况。*Readings in Natural Language Processing*(Grosz et al., 1986)搜集了自然语言处理领域中的很多基础性研究论文，尽管内容有些陈旧，但所搜集的文章都是非常优秀的论文。

有关语音和语言处理资源的各种材料，现在都可以通过万维网查询。在本书的主页上可以找到这些资源。本书的主页是：http://www.cs.colorado.edu/~martin/slp.html。

① 本书中译本《统计自然语言处理基础》已由电子工业出版社引进出版。书号7-5053-9921-7。——编者注

② 本书中译本《自然语言理解》已由电子工业出版社引进出版。书号7-121-00755-X。——编者注

第一部分　词汇的计算机处理

词是语言的最基本的建筑材料。人类的所有语言，无论是口头语言、记号语言还是书面语言，都是由词构成的。语音和语言处理的每个领域，从语音识别到机器翻译，再到Web上的信息检索，都要求具有丰富的词的知识。人类语言处理的心理语言学模型和生成语言学模型也都是建立在词汇知识的基础上的。

这个部分包括6章，介绍了单词的拼写模型、发音模型和形态模型，以及与单词知识有关的3个重要的实际应用方面：自动语音识别（automatic speech recognition，简称ASR）、文本-语音合成（text-to-speech synthesis，简称TTS）和拼写错误校正（correction of spelling error）。最后，在这些章节中还讲述了一种对于语音和语言处理可能最重要的计算模型：自动机。介绍的自动机有4种类型：有限状态自动机（Finite State Automata，简称FSA）和正则表达式、有限状态转录机（Finite State Transducer，简称FST）、加权转录机（weighted transducer）、隐马尔可夫模型（Hidden Markov Model，简称HMM）以及单词序列的N元语法模型。

第2章　正则表达式与自动机

In the old days, if you wanted to impeach a witness you had to go back and fumble through endless transcripts. Now it's on a screen somewhere or on a disk and I can search for a particular word—say every time the witness used the word glove*—and then quickly ask a question about what he said years ago. Right away you see the witness get flustered.*

Johnnie L. Cochran Jr., attorney, *New York Times*, 9/28/97[①]

假定你是美洲旱獭（woodchuck）的爱好者，想了解关于这种林地动物的更多信息。你可以在你喜欢的Web浏览器中键入*woodchuck*这个词。浏览器会返回几个网站。你灵机一动，继续键入这个词的复数形式*woodchucks*，这时就会发现屏幕上显示出如下字样："*interesting links to woodchucks and lemurs*"（与美洲旱獭和狐猴相链接）和"*all about Vermont's unique, endangered species*"（佛蒙特州仅有的全部濒危物种）。如果你不愿重复做这样的搜索，而只想搜索一次，可以键入指令*woodchuck with an optional final s*（带有一个可选末尾字母*s*的*woodchuck*）。此外，你可能想找到一个网站，其中*woodchuck*的第一个字母为大写字母*W*（即*Woodchuck*）。你可能还想查询在某个文件中的所有物价；你可能想看到诸如$199，$25，$24.99的所有符号串。所有这些情况下，都要用到正则表达式的知识。在本章中，我们将介绍正则表达式，这是字符文本序列的标准记录方式。在上述Web搜索的例子中，在信息检索的其他应用中，都要使用正则表达式；在PC，Mac和UNIX的词处理中，在语料库的频率计算以及其他类似的应用中，正则表达式都起着重要的作用。

介绍了正则表达式之后，我们将通过有限状态自动机（finite-state automaton）来进一步说明如何实现这些正则表达式。有限状态自动机不仅是一种用来实现正则表达式的数学工具，而且也是计算语言学中最有用的工具。诸如有限状态转录机这样的自动机的变种、隐马尔可夫模型以及*N*元语法，都是语音识别、语音合成、拼写检查和信息抽取的重要组成部分，我们在下面的章节中将介绍这些内容。

2.1　正则表达式

SIR ANDREW: Her C's, her U's and her T's: why that?

Shakespeare, *Twelfth Night*

正则表达式（regular expression，简称RE）是计算机科学标准化中一项尚未被赞颂过的成就。它是一种用于描述文本搜索符号串的语言。在UNIX（vi, Perl, Emac和grep）和Microsoft Word（第六版和更高的版本）以及WordPerfect中，用于搜索文本的正则表达式语言几乎是完全一样的。

---

① 在过去的日子里，如果你对某个证据有所怀疑，必须回过头去检查无穷无尽的抄本。现在，我只需要在屏幕上或者在磁盘上查询某个特别的单词。例如，每次都使用单词Glove来查询有关的证据，就可以立刻查问到关于某人多年以前所说过的与这个单词有关的问题，马上你就可以激动地看到你所需的证据了。

在不同的Web搜索引擎中，存在着具有不同特征的正则表达式。除了这些实际的用处之外，正则表达式还是计算机科学和语言学的一种最重要的理论工具。

正则表达式最初是由Kleene(1956)开发的，更详细的情况可参阅“历史”这一节。正则表达式是专用语言中用于描述符号串的简单类别的公式。符号串是符号的序列；对于大多数基于文本的检索技术来说，符号串就是字母数字字符（字母、数字、空白、表、标点符号）的任意序列。在基于文本的检索技术中，一个空白相当于一个字符，与其他字符一样同等对待，我们用符号␣来表示空白。

从形式上说，正则表达式是用来刻画符号串集合的一个代数表述。因此，它可以用于描述搜索符号串，也可以用于以形式的方法定义一种语言。我们首先讲述如何用正则表达式描述文本搜索，然后逐渐讲解正则表达式的其他用途。在2.3节中，我们将说明只需要三个正则表达式的算符就足以刻画符号串，但是在本节中都使用的是Perl语言中便捷而通用的正则表达式句法。由于普通文本处理程序中采用的大多数正则表达式句法是一致的，所以这里所讲的都可以扩展应用于UNIX，Microsoft Word和WordPerfect的正则表达式。附录A给出了与Perl的句法不同的几个领域。

正则表达式的搜索要求有一个试图搜索的模式（pattern）和一个被搜索的文本语料库（corpus）。正则表达式的搜索函数将对整个语料库进行搜索，并返回包含该模式的所有文本。在诸如Web搜索引擎这样的信息检索系统（IR）中，文本将是整个文档或Web页。在一个词处理系统中，文本可以是独立的单词，或者是文档行。在本章的其他部分，我们将使用后一种范式。因此，如果我们给出一个搜索模式，将假定搜索引擎返回的是文档行。这正是UNIX grep命令所做的工作。下面我们将用下划线强调模式中与正则表达式匹配的部分。对于一个正则表达式来说，搜索可以返回所有的匹配，也可以只返回第一个匹配。在这里我们只显示第一个匹配。

2.1.1　基本正则表达式模式

最简单的正则表达式是由简单字符构成的一个序列。例如，要搜索*woodchuck*，可以键入/woodchuck/这个正则表达式。这样，正则表达式/Buttercup/就与包含子字符串*Buttercup*的任何字符串相匹配，例如I'm called little Buttercup（我们假定在这个搜索应用中返回的是整行）。从此处起，我们将在正则表达式的前后加斜线“/”，以便区分正则表达式和模式。之所以使用斜线，是因为这种表示法是Perl使用的，但在这种表示法中斜线并不是正则表达式的一部分。

搜索符号串可能只包含一个单独的字母（如/!/），或者包括字母序列（如/urgl/）。我们在与正则表达式相匹配的第一个例子下面加了下划线（实际上也可以选择返回比第一个例子更多的东西）。

正则表达式	匹配模式的实例
/woodchucks/	“interesting links to woodchucks and lemurs”
/a/	“Mary Ann stopped by Mona’s”
/Claire␣says,/	“Dagmar, my gift please,” Claire says,”
/song/	“all our pretty songs”
/!/	“You’ve left the burglar behind again!” said Nori

正则表达式是区分大小写的（case sensitive）；小写/s/区别于大写/S/；/s/与小写*s*匹配，/S/与大写*S*匹配。这意味着，/woodchuck/与字符串*Woodchuck*不匹配。我们使用方括号“[”和“]”来解决这个问题。内部有括号的字符串表示所匹配的字符是析取（disjunction）的。例如，图2.1表明，与/[*wW*]/匹配的模式中或者包含*w*，或者包含*W*。

正则表达式	匹配	模式的例子
/[wW]oodchuck/	Woodchuck 或 woodchuck	"Woodchuck"
/[abc]/	'a', 'b', 或 'c'	"In uomini, in soldati"
/[1234567890]/	任何数字	"plenty of 7 to 5"

图 2.1 用括号[ ]表示字符的析取

正则表达式/[1234567890]/可以表达任何简单数字。类似数字或字母的字符都是构成表达式的重要建筑材料，它们处理起来有时会变得很不方便。例如，当我们用"任意大写字母"正则表达式

```
/[ABCDEFGHIJKLMNOPQRSTUVWXYZ]/
```

来描述任何大写字母时，就显得很不方便。在这样的情况下，可以用连字符（-）来表示在某一范围（range）内的任何字符。正则表达式/[2-5]/表示字符 2,3,4 和 5 范围内的一个任意符号。正则表达式/[b-g]/表示字符 b,c,d,e,f 和 g 范围内的一个任意符号。下面是其他的例子：

正则表达式	匹配	匹配模式的例子
/[A-Z]/	一个大写字母	"we should call it 'Drenched Blossoms'"
/[a-z]/	一个小写字母	"my beans were impatient to be hoed!"
/[0-9]/	一个单独的数字	"Chapter 1: Down the Rabbit Hole"

图 2.2 使用括号[ ]和连字符 - 表示某个范围

通过使用脱字符(^)，方括号还可以用来表示不出现某个单独的字符。如果在开方括号之后有脱字符，那么相应的模式就是否定的。例如，正则表达式/[^a]/与任何不包含 a 的单个字符相匹配。但是，这种用法仅当脱字符处于开方括号之后的第一个位置时才有效。如果脱字符出现在其他位置，它只能表示脱字符本身。图 2.3 是一些具体的例子。

正则表达式	匹配（单字符）	匹配模式的例子
[^A-Z]	不是一个大写字母	"Oyfn pripetchik"
[^Ss]	既不是'S'也不是's'	"I have no exquisite reason for't"
[^\.]	不是点号	"our resident Djinn"
[e^]	'e'或'^'	"look up ^ now"
a^b	模式'a^b'	"look up a^b now"

图 2.3 使用脱字符 ^ 表示否定或仅仅表示它自身

通过使用方括号，解决了 *woodchuck* 的大小写问题，但还不能回答前面提出的那个问题：如何既表示 *woodchuck* 又表示 *woodchucks*？我们不能用方括号实现这样的表示，因为方括号允许说"s 或 S"，但不允许说"s 或无"。为此，我们使用问号/?/来表示"前一个字符或无"，如图 2.4 所示。

正则表达式	匹配	匹配模式的例子
woodchucks?	woodchuck 或 woodchucks	"woodchuck"
colou?r	color 或 colour	"colour"

图 2.4 问号表示它前面的字符是可选的

我们可以把问号的意义看成是"前一个字符的有或无"。这是一种表达我们想要多少东西的方法。迄今为止，我们还没有介绍过如何表示一个以上的事物，然而有时需要用正则表达式表示重复的事物。例如，羊的叫声可以看成语言，这种语言是如下包含重复符号的符号串：

baa!

baaa!

baaaa!

baaaaa!

baaaaaa!

...

这种语言的开头是一个 *b*，后面跟着至少两个 *a*，最后是一个惊叹号。有一种基于星号或“*”的算符可以允许我们表达“若干个 *a*”，这种算符称为“Kleene *”（英文读为“cleany star”，我们不妨将其读为“Kleene 星号”）。Kleene 星号意味着“前面紧邻的字符或正则表达式出现零次或连续出现若干次”。这样，/a*/ 就可以表示“由零或若干个 *a* 构成的符号串”，它可以与 *a* 或 *aaaaaa* 相匹配，也可以与 *Off Minor* 相匹配，因为 Off Minor 包含零个 *a*。所以，与包含一个或多个 *a* 的符号串相匹配的正则表达式是 /aa*/，它表示一个 *a* 后面跟着零个或多个 *a*。更复杂的模式也可以重复。所以，/[ab]*/ 表示“零个或多个 *a* 或 *b*”（不是表示“零个或多个右方括号”）。这个正则表达式可以与类似 *aaaa*，*ababab* 或 *bbbb* 的符号串相匹配。

现在我们已经完全知道了怎样用正则表达式来表示多位数的价钱。单位数的价钱的正则表达式是 /[0-9]/。因此，一个整数（数字串）的正则表达式就是 /[0-9][0-9]*/。为什么不是 /[0-9]*/ 呢?

有时，把数字的正则表达式写两次会令人感到厌烦，因此提出了一种表示数字“最少有一个”的简单方法。这种方法就是 Kleene 加号（Kleene+），Kleene 加号意味着“前面一个或多个字符”。因此，正则表达式 /[0-9]+/ 是“数字序列”的规范表达式。羊叫声的语言有两种表示方法：/baaa*!/ 和 /baa+!/。

还有一个重要的字符就是点号（/./），这是一个通配符，表示任何与单个字符（回车符除外）相匹配的字符。

正则表达式	匹配	模式的例子
/beg.n/	位于*beg*和*n*之间的任何字符	begin, beg'n, begun

图 2.5　用点号表示任意字符

通配符经常与 Kleene 星号结合起来使用，含义是“任何字符串”。例如，如果我们想找到文本中的某一行，其中 aardvark 一词出现了两次。我们可以用正则表达式表示为：/aardvark.*aardvark/。

锚号（anchor）是一种把正则表达式锚在符号串中的某个特定位置的特殊字符。最普通的锚号是脱字符（^）和美元符（$）。脱字符与行的开始相匹配。正则表达式 /^The/ 表示单词 The 只出现在一行的开始。这样，脱字符可以有三种用法：表示一行的开始；在方括号内表示否定；只表示脱字符本身。什么样的上下文可以让 Perl 知道某个特定脱字符的功能是什么呢？。美元符 $ 表示一行的结尾，所以模式“_$”是一个有用的模式，表示一行的结尾是一个空白。正则表达式 /^The dog\.$/ 表示仅包含短语 *The dog* 的一个行。这里必须使用反斜杠“\”，因为我们想让“.”表示点号，而不表示通配符。

此外还有两个其他的锚号：\b 表示词界，而 \B 表示非词界。因此，/\bthe\b/ 表示单词 *the*，而不表示单词 other。从技术上讲，Perl 把词定义为数字、下划线或字母的任何序列。这是根据像 Perl 和 C 这样的程序语言中关于词的定义而言的。例如，/\b99/ 表示在“*There are 99 bottles of beer on the wall*”中的符号串 99，因为 99 在一个空白后面。但是，这个正则表达式不表示“*There are 299 bottles of beer on the wall*”中的符号串 99，因为 99 在一个数字后面。然而，这个正则表达式可以表示 $99 中的 99，因为 99 跟在美元符号 $ 后面，$ 不是数字、下划线或字母。

2.1.2 析取、组合与优先关系

假定我们需要搜索关于宠物的文本，而且我们对 *cat* 或 *dog* 最感兴趣。这时，我们试图搜索符号串 *cat* 或 *dog*。因为我们不能使用方括号来搜索 "*cat* 或 *dog*"（为什么不能呢?），所以需要一个称为析取算符（disjunction operator）的新算符 "|"，这样的算符又称为析取符（pipe symbol）。正则表达式 /cat|dog/ 表示或者是符号串 *cat*，或者是符号串 *dog*。

有时我们需要在比较长的序列中间使用析取符。例如，假定我想为表弟 David 搜索关于他的宠物 *guppy*（虹）的信息，我要怎样才可以同时表达 *guppy* 和它的复数形式 *guppies* 呢? 我们不能简单地表示为 /guppy|ies/，因为这样的表达式只能与符号串 *guppy* 和 *ies* 相匹配。像 *guppy* 这样的符号序列优先于（precedence）析取符 "|"。为了使析取算符只能应用于特定的模式，我们需要使用圆括号算符 "(" 和 ")"。把一个模式括在圆括号中，它就能够像一个单独的字符来使用，而且在其中可以使用析取符 "|" 和 Kleene* 等算符。因此，表达式 /gupp(y|ies)/ 表示析取符仅仅应用于后缀 *y* 和 *ies*。

当我们使用如 Kleene* 这样的计数符时，圆括号算符 "(" 也是很有用的。与算符 "|" 不同，Kleene* 算符只能用来表示单个字符，不能用来表示整个序列。如果我们想匹配某一符号串的重复出现，可以使用行符号包含列标记 *Column 1 Column 2 Column 3*。表达式 /Column) [0-9] + */ 不能与任何列相匹配，但可以与一个后面有任意数目的空白的列相匹配！星号 "*" 在这里仅仅用于表示它前面的空白符号 "_"，而不表示整个序列。我们可以用圆括号写出正则表达式 /(Column [0-9] +_*)*/，这个表达式与单词 *Column* 后面跟着一个数字和任意数目的空白组成的符号串相匹配。整个模式可以重复任意次数。

可见，一个算符可能优先于其他算符，因此我们有必要使用括号来表示这种优先关系。在正则表达式中，这种优先关系是通过算符优先层级（operator precedence hierarchy）来形式地描述的。下面的表中给出了正则表达式算符优先的顺序，其优先级按从高到低的顺序排列：

圆括号	( )
计数符	* + ? { }
序列与锚	the ^my end$
析取符	|

由于计数符比序列具有更高的优先性，所以 /the*/ 与 *theeeee* 相匹配，而不与 *thethe* 相匹配。由于序列比析取符具有更高的优先性，所以 /thelany/ 与 *the* 或 *any* 相匹配，而不与 *theny* 相匹配。

模式有时可能具有歧义。当正则表达式 /[a-z]*/ 与 *once upon a time* 这个文本相匹配时，由于 /[a-z]*/ 可以与零个或更多个字母相匹配，所以这个正则表达式可以什么都不匹配，也可以与首字母 *o*，*on*，*onc* 或 *once* 相匹配。在这些场合，正则表达式应该总是尽其可能与其中最长的符号串相匹配，它应该匹配 *once*。我们可以说，这些模式总是贪心地（greedy）扩充，试图覆盖尽可能长的符号串。

2.1.3 一个简单的例子

假定我们想写一个正则表达式来查找英语的冠词 *the*，我们可以写出一个简单的表达式（但并不正确）：

```
/the/
```

这个表达式不能表示当*the*位于句子开头时的情况，因为这时*the*的第一个字母要大写，即写为The。这使我们想到使用表达式：

```
/[tT]he/
```

但是，当文本中*the*嵌入在其他单词中间的时候（例如，*other*或*theology*），这样的表达式就不正确了。这时，我们就需要在表达式中说明，一个单词的两端应该有边界，表达式应该是：

```
/\b[tT]he\b/
```

如果不用/\b/，我们是不是也可以达到这个目的呢？因为/\b/不能处理*the*后面带下划线或数字的情况，我们也不想把下划线或数字看成是词的界限。但是，我们试图在可能出现下划线或数字的某个上下文中找到*the*（例如，*the_*或*the25*）。我们需要说明，在*the*的两侧不能出现字母。这时，表达式为：

```
/[^a-zA-Z][tT]he[^a-zA-Z]/
```

但是，这个表达式仍然还有问题。当*the*出现在一行的开头时，我们就会找不到它。这是因为我们曾经用正则表达式[^a-zA-Z]来避免嵌入的*the*，这意味着，在文本中the的前面必定有某个单独的字符，哪怕这个字符是非字母字符。如果我们说明，在the的前面或者是一行的开头，或者是非字母字符，就可以避免这样的问题。这时的正则表达式如下：

```
/(^|[^a-zA-Z])[tT]he[^a-zA-Z]/
```

2.1.4　一个比较复杂的例子

让我们举个更有意义的例子来说明正则表达式的能力。假定我们想要用正则表达式帮助用户在Web上购买计算机。用户需要的是“500 MHz以上，32 Gb磁盘空间，价钱低于$1000的PC机”。为了进行这样的检索，我们首先需要能够查找诸如*500 MHz*, *32 Gb*, *Compaq*, *Mac*, *$999.99*这样的表达式。在本节的其他部分，我们将设计某些正则表达式来做这样的工作。

首先，我们来设计关于价格的正则表达式。下面是美元符号$后接一个数字符号串的表达式。注意，Perl善于表达这样的$，而不让它表示行尾。正则表达式如下（它能做到这一点吗?）：

```
/$[0-9]+/
```

现在我们需要处理美元中的小数部分，可以在上述表达式后加小数点和两个数字。正则表达式如下：

```
/$[0-9]+\.[0-9][0-9]/
```

这样的表达式只能表示*$199.99*，而不能表示*$199*。我们需要把小数部分设成可随意选择的，并且确定单词的边界。正则表达式如下：

```
/\b$[0-9]+(\.[0-9][0-9])?\b/
```

怎样来表达处理器的速度（兆赫，megahertz=MHz；千兆赫，gigahertz=GHz）呢？表达式如下：

```
/\b[0-9]+␣*(MHz|[Mm]egahertz|GHz|[Gg]igahertz)\b/
```

注意，我们用/␣*/表示“零或更多空间”，因为这里可能总会有一些多余的空间。在处理磁盘空间（千兆字节，Gb=gigabytes）或存储量（兆字节，Mb=megabytes；千兆字节，Gb=gigabytes）

时，我们还要允许千兆字节的小数是随选的（5.5 Gb）。注意，这里使用“?”来表示最后一个s是随选的。正则表达式如下：

```
/\b[0-9]+␣*(Mb|[Mm]egabytes?)\b/
/\b[0-9](\.[0-9]+)?␣*(Gb|[Gg]igabytes?)\b/
```

最后，我们还可以用简单的正则表达式来表示系统和商家的名称：

```
/\b(Win95|Win98|WinNT|Windows␣*(NT|95|98|2000)?)\b/
/\b(Mac|Macintosh|Apple)\b/
```

2.1.5 高级算符

还有一些有用的正则表达式高级算符（advanced operator）。图2.6列出了一些有用的通用字符的替换名，使用这些替换名可以节省打字的工作量。除了Kleene* 和Kleene+之外，我们还可以使用花括号括起来的数字作为计数符。例如，正则表达式/{3}/表示“前面的字符或表达式正好出现3个”。这样，/a\{24}z/就表示*a*后面跟随着24个点，再跟随着一个*z*（不是*a*后面跟随着23个或者25个点再跟随着一个*z*）。

正则表达式	扩充表达式	匹配	模式的例子
\d	[0-9]	任何数字字符	Party␣of␣5
\D	[^0-9]	任何非数字字符	Blue␣moon
\w	[a-zA-Z0-9␣]	任何字母数字字符或空白	Daiyu
\W	[^\w]	一个非字母数字字符	!!!!
\s	[␣\r\t\n\f]	空白区域（空白，表格）	
\S	[^\s]	非空白区域	in␣Concord

图2.6 通用字符集的替换名

数字的范围也可以用类似的办法来表示。/{n,m}/表示前面的字符或表达式出现*n*个到*m*个；/{n,}/表示前面的表达式至少出现*n*个。图2.7中的表总结了用于计数符的正则表达式。

正则表达式	匹配
*	前面的字符或表达式出现零个或多个
+	前面的字符或表达式出现一个或多个
?	前面的字符或表达式恰恰出现零个或一个
{n}	前面的字符或表达式出现*n*个
{n,m}	前面的字符或表达式出现*n*个到*m*个
{n,}	前面的字符或表达式至少出现*n*个

图2.7 用于计数符的正则表达式算符

最后，如图2.8所示，还可以用基于右斜杠（\）的记法来引用某些特殊字符。最普通的记法就是换行符（newline）“\n”和制表符（tab）“\t”。为了引用某个特殊的字符，例如.，*，[和\，可以在这个字符前面加右斜杠\（/\./，/*/，/\[/和/\\/）。

正则表达式	匹配	匹配模式的例子
*	星号"*"	“K*A*P*L*A*N”
\.	点号“.”	“Dr. Livingston, I presume”
\?	问号	“Would you light my candle?”
\n	换行符	
\t	表格符	

图2.8 某些加右斜杠的字符

关于正则表达式的更加详细的说明，以及关于 Perl，UNIX 和 Microsoft Word 中正则表达式的差别，读者可参见附录 A。

2.1.6　正则表达式中的替换、存储器与 ELIZA

正则表达式的一个重要用途是替换（substitution）。例如，Perl 的替换运算符 s/regexp1/regexp2/ 可把一个用正则表达式描述的符号串替换为另一个用正则表达式描述的符号串：

```
s/colour/color/
```

我们经常需要引用与某个模式相匹配的符号串中的特定部分。例如，假定要把文本中的所有整数的两侧加上尖括号，具体地说，要把 *the 35 boxes* 替换为 *the <35> boxes*。我们只要引用相应的整数，就可以很容易地在它的两侧加上尖括号。为此，在第一个模式（即整数）的两侧加上圆括号 "(" 和 ")"，然后在第二个模式中使用数字算符（number operator）"\1" 以便回过头去参照第一个模式：

```
s/([0-9]+)/<\1>/
```

圆括号和数字也可以用来说明某个符号串或表达式在文本中出现两次。例如，假定我们要查找模式 "the Xer they were, the Xer they will be",想让其中的 X 指同一个符号串。这时，我们在第一个 X 的两侧加圆括号，并且用数字算符 "\1" 替换第二个 X:

```
/the (.*)er they were, the \1er they will be/
```

这里的 "\1" 可以与用圆括号中的项相匹配的任何符号串来替换。因此，与这个正则表达式相匹配的符号串是 *The bigger they were, the bigger they will be*，而不是 *The bigger they were, the faster they will be*。这样的数字算符也可以与其他数字一起使用。如果要匹配的是两个不同的圆括号中的符号串集合，\2 就表示第二个圆括号中的符号串集合。例如

```
/the (.*)er they (.*), the \1er they \2/
```

就可以匹配 *The bigger they were, the bigger they were*，但是这个表达式不能匹配 *The bigger they were, the bigger they will be*。

这样的数字存储器称为寄存器（register）。例如，寄存器 1、寄存器 2、寄存器 3 等。这样的存储特性不是每种正则表达式语言都具备的，但这常常是正则表达式的 "扩充" 特性。

这种使用替换的存储器在开发诸如 ELIZA(Weizenbaum, 1966) 之类的简单的自然语言理解程序时是非常有用的。ELIZA 模拟一个心理咨询医生的行为，它可以同用户进行对话。例如：

用户$_1$:　Men are all alike.
ELIZA$_1$:　IN WHAT WAY.
用户$_2$:　They're always bugging us about something or other.
ELIZA$_2$:　CAN YOU THINK OF A SPECIFIC EXAMPLE.
用户$_3$:　Well, my boyfriend made me come here.
ELIZA$_3$:　YOUR BOYFRIEND MADE YOU COME HERE.
用户$_4$:　He says I'm depressed much of the time.
ELIZA$_4$:　I AM SORRY TO HEAR YOU ARE DEPRESSED.

ELIZA 进行一连串的正则表达式的替换，每个正则表达式与输入行中的某个部分相匹配，然后替换并改变它们。首先的替换是把所有的 *my* 替换为 *YOUR*，把所有的 *I'm* 替换为 *YOU ARE*，等等。然后的替换是查找输入中相关的模式，并产生出适合的输出。下面是一些例子：

```
s/.* YOU ARE (depressed|sad) .*/I AM SORRY TO HEAR YOU ARE \1/
s/.* YOU ARE (depressed|sad) .*/WHY DO YOU THINK YOU ARE \1/
s/.* all .*/IN WHAT WAY/
s/.* always .*/CAN YOU THINK OF A SPECIFIC EXAMPLE/
```

因为对于一个文本可能进行多重的替换，需要对替换编序号，以便按顺序使用它们。作为练习，请你创建这样的模式。

2.2 有限状态自动机

正则表达式充其量不过是一种用于文本搜索的方便的元语言。首先，正则表达式是描述有限状态自动机（finite-state automaton，简称FSA）的一种方法。有限状态自动机是本书中将要描述的计算工作的理论基础。任何正则表达式都可以用有限状态自动机来实现（除了使用存储特性的那些正则表达式之外，关于这个问题以后还会进一步详述）。任何有限状态自动机都可以用正则表达式来描述。有限状态自动机和正则表达式彼此对称。其次，正则表达式是用来刻画正则语言（regular language）的一种方法，正则语言是一种特别的形式语言。正则表达式、有限状态自动机和正则语言这三种理论结构的关系可用图2.9来说明。

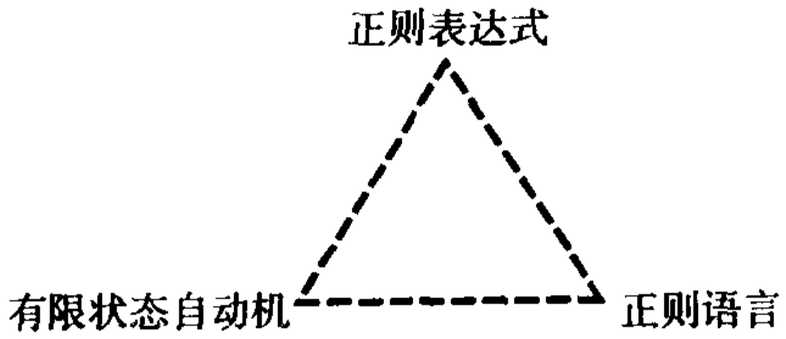


图2.9 有限状态自动机、正则表达式和正则语言之间的关系；此图是Martin Kay提出的

这一节首先介绍与上节的正则表达式有关的有限状态自动机，然后讲述从正则表达式到自动机的一般的映射方法。尽管我们是从如何实现正则表达式开始讲述有限自动机的，但FSA还有各种不同的用处，我们将在本章和下面的章节中讲述这些问题。

2.2.1 用FSA来识别羊的语言

After a while, with the parrot's help, the Doctor got to learn the language of the animals so well that he could talk to them himself and understand everything they said.

Hugh Lofting, *The Story of Doctor Dolittle*

让我们从前面讨论过的羊的语言开始。我们把羊的语言定义为由下面的（无限）集合构成的任何符号串：

baa!
baaa!
baaaa!
baaaaa!
baaaaaa!
...

① 过了一会儿，在鹦鹉的帮助下，博士先生把动物的语言学得非常好，以至于博士能够亲自和动物谈话，并且能够理解它们所说的一切。

Hugh Lofting《多立特博士的故事》

描述这种羊的语言的正则表达式是 /baa+!/。图 2.10 是模拟这种正则表达式的一个自动机（automaton）。自动机（也可以称为有限自动机或有限状态自动机，简称 FSA）能够识别符号串的集合，在我们的例子中，符号串就是羊的语言，这与正则表达式所做的方式相同。我们用有向图来表示自动机。有向图包括两部分：点（或结点）的有限集合和两个点之间的有向连接的弧的集合。我们用圆圈来表示点，用箭头来表示弧。一个自动机有五个状态，可以用图论中的结点来表示。状态 0 是初始状态（start state），用进入的箭头来表示；状态 4 是最后状态（final state）或接收状态（accepting state），用双圈来表示。在上面的有向图中还有 4 个转移（transition），用弧来表示。

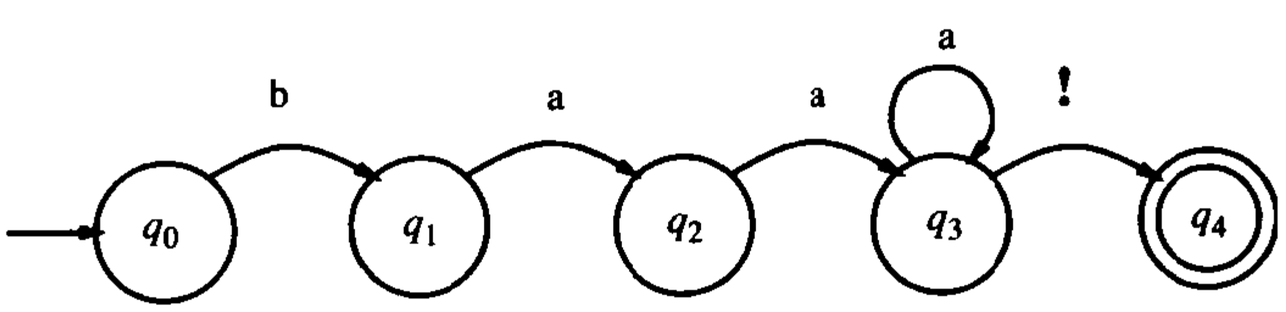


图 2.10　用于羊的语言的有限自动机

FSA 可以用来识别（或接收）符号串。接收方式如下：首先，把输入想像成写在一个长的带子上，带子可分为一些单元格（cell），带子的一个单元上可以写一个符号，如图 2.11 所示。

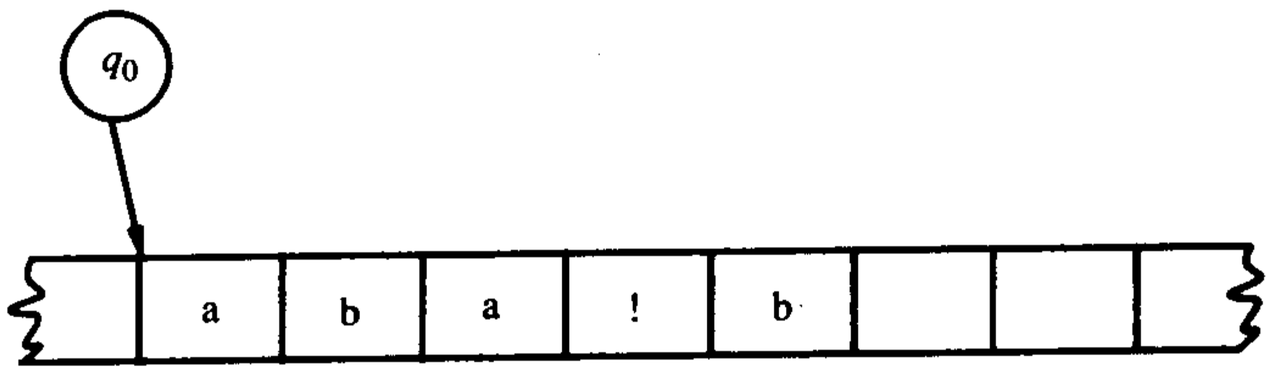


图 2.11　带有单元格的带子

自动机从状态（q_0）开始，反复进行如下的过程：查找输入带子上的下一个字母。如果带子上的字母与自动机中离开当前状态的弧相匹配，那么就穿过这个弧，移动到下一个状态，而在输入带子上也相应地向前移动一个符号。如果输入带子上的符号已经读完，则进入接收状态（q_4），那么，自动机就成功地识别了记录在输入带子上的羊的语言。如果自动机总不能够进入最后状态，这或者是因为输入带子上的符号已经读完，或者是因为某些输入与自动机的弧不匹配（如图 2.11 所示），或者是因为自动机在某一个非终极状态停住了，这时我们就说，自动机拒绝（reject）输入符号，或者说自动机不能接收输入。

我们也可以用状态转移表（state-transition table）来表示自动机。与图的表示方法一样，状态转移表可以表示初始状态、接收状态和符号在状态之间转移的情况。下面是图 2.10 中的 FSA 的状态转移表。

我们在状态 4 后面加了冒号，这表示 4 是终极状态（你想要多少个终极状态就可以有多少个终极状态），∅ 表示非法转移或不能转移。第一行可以读为：“如果在状态 0 并且看到 b，那么就应该转移到状态 1；如果在状态 0 并且看到输入符号 a 或!，那么就失败了”。

从形式上说，一个有限自动机可以用下面 5 个参数来定义：

- Q：N 中状态 $q_0, q_1, \cdots, q_N$ 的有限集合
- Σ：有限的输入符号字母表
- q_0：初始状态
- F：终极状态的集合，$F \subseteq Q$

- $\delta(q, i)$：状态之间的转移函数或转移矩阵。给定一个状态 $q \in Q$ 和一个输入符号 $i \in \Sigma$，$\delta(q, i)$ 返回一个新的状态 $q' \in Q$，因此，$\delta(q, i)$是从 $Q \times \Sigma$ 到 Q 的一个关系。

在图 2.10 关于羊的语言的自动机中，$Q = \{q_0, q_1, q_2, q_3, q_4\}$，$\Sigma = \{a, b, !\}$，$F = \{q_4\}$，而 $\delta(q, i)$ 由图 2.12 中的转移表来确定。

	输出		
状态	b	a	!
0	1	0	0
1	0	2	0
2	0	3	0
3	0	3	4
4:	0	0	0

图 2.12　图 2.10 中的 FSA 的状态转移表

图 2.13 介绍了使用状态转移表识别符号串的算法。这个算法是“确定性的识别器”，因此简称为 D-RECOGNIZE。确定性（deterministic）算法是一种没有选择点的算法，对于任何输入，算法总是知道怎样工作。下一节将介绍非确定的自动机，这种非确定的自动机必须决定下一步应该移动到哪个状态。

```
function D-RECOGNIZE(tape, machine) returns accept or reject

  index ← Beginning of tape
  current-state ← Initial state of machine
  loop
   if End of input has been reached then
    if current-state is an accept state then
      return accept
    else
      return reject
   elsif transition-table[current-state,tape[index]] is empty then
     return reject
   else
     current-state ← transition-table[current-state,tape[index]]
     index ← index + 1
  end
```

图 2.13　FSA 确定性识别的一个算法。如果整个符号串都在 FSA 所定义的语言中，则返回接收；如果符号串不在这个语言中，则拒绝

D-RECOGNIZE 使用一个输入带子和一个自动机。如果在带子上的符号串被自动机接收了，就返回 accept；否则返回 reject。注意，因为 D-RECOGNIZE 假定它已经指向了被查找的符号串，所以它的任务只是通常使用的正则表达式的一般问题的一个部分。例如，我们通常使用正则表达式来发现语料库中的某一个符号串。

D-RECOGNIZE 开始工作时要初始化变量 *index*，使之成为带子的开头，把 *current-state* 作为机器的初始状态。然后，D-RECOGNIZE 进入一个 loop，以驱动算法的其他部分。它要检查是否已经到达了输入的终点。这样，它便可以接收输入（如果当前状态是接收状态），或者拒绝输入（如果不是）。

如果还有输入留在带子上，D-RECOGNIZE 就要查看状态转移表，以决定下一步要移动到哪个状态。变量 *current-state* 指出它要查找转移表中的哪一列，而带子上的当前符号则指出它要查找转

移表中的哪一行。这样做出的转移表中的项用于更新变量*current-state*和*index*的值，并且逐渐在带子上向前推进。如果转移表的项为空，机器就不知道往哪里走，所以必须拒绝输入。

图 2.14 说明了用这个算法来处理表示羊的语言的 FSA（输入符号串是 baaa!）的追踪过程。

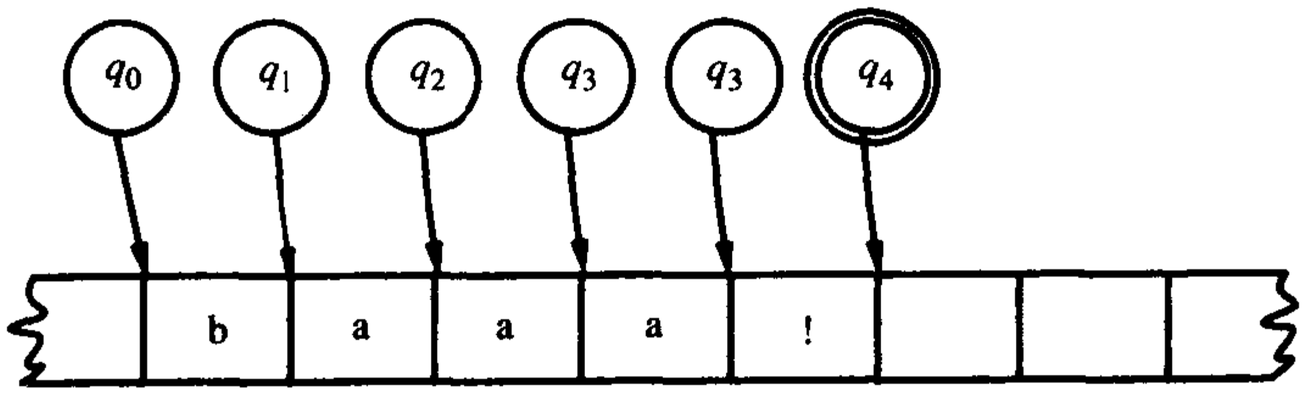


图 2.14　把 FSA#1 用于处理羊的语言的追踪过程

在检查带子的开头时，机器处于状态 q_0；在输入带子上找到 b，于是根据图 2.12 转移表[q_0, b]中的内容的指示，转入状态 q_1；然后，在带子上发现一个 a，于是转入状态 q_2；接着又发现 a，转入 q_3；在状态 q_3，接着又发现第三个 a，于是离开 q_3，沿着回路又返回到这个 q_3；接着，读到带子上的最后一个符号！，于是转入 q_4。由于这时输入带子上不再有符号，输入带子变空，在回路的开始输入终点（End of input）的条件首次得到满足，机器在状态 q_4 停止。状态 q_4 成为接收状态，因此机器接收了符号串“baaa!”，这个符号串是羊的语言中的成立符号串。

如果对于某一对由给定的状态及其相应的输入符号构成的结合，在转移表中不存在合法的转移，那么算法就将失败。例如，由于状态转移表中状态 q_0 和输入符号 a 构成的结合不存在合法的转移（也就是说，在图 2.12 的转移表中，q_0 和 a 构成的结合的值为 $\emptyset$），所以算法不能处理输入符号串 abc。也就是说，即使自动机允许以 a 开始的符号串，但如果在输入中出现 c，算法也一定要失败，因为在羊的语言的字母表中根本就没有 c 这个字母。我们可以把转移表中的这些“空”元素想像成全都指向一个“空”状态。我们可以把这个空状态称为失败状态（fail state）或吸收状态（sink state）。在这个意义上说，如果我们给自动机添加了一个失败状态，就能把自动机看成是具有空转移的自动机，而对于这种自动机的每个状态来说，都可以画出附加的弧来对应于这些空状态。这样，对于任何状态和任何可能的输入，自动机总是能够找到一个可以转移的地方。图 2.15 是在图 2.10 中 FSA 的基础上加了失败状态 q_F 而构成的一个更加完整的自动机。

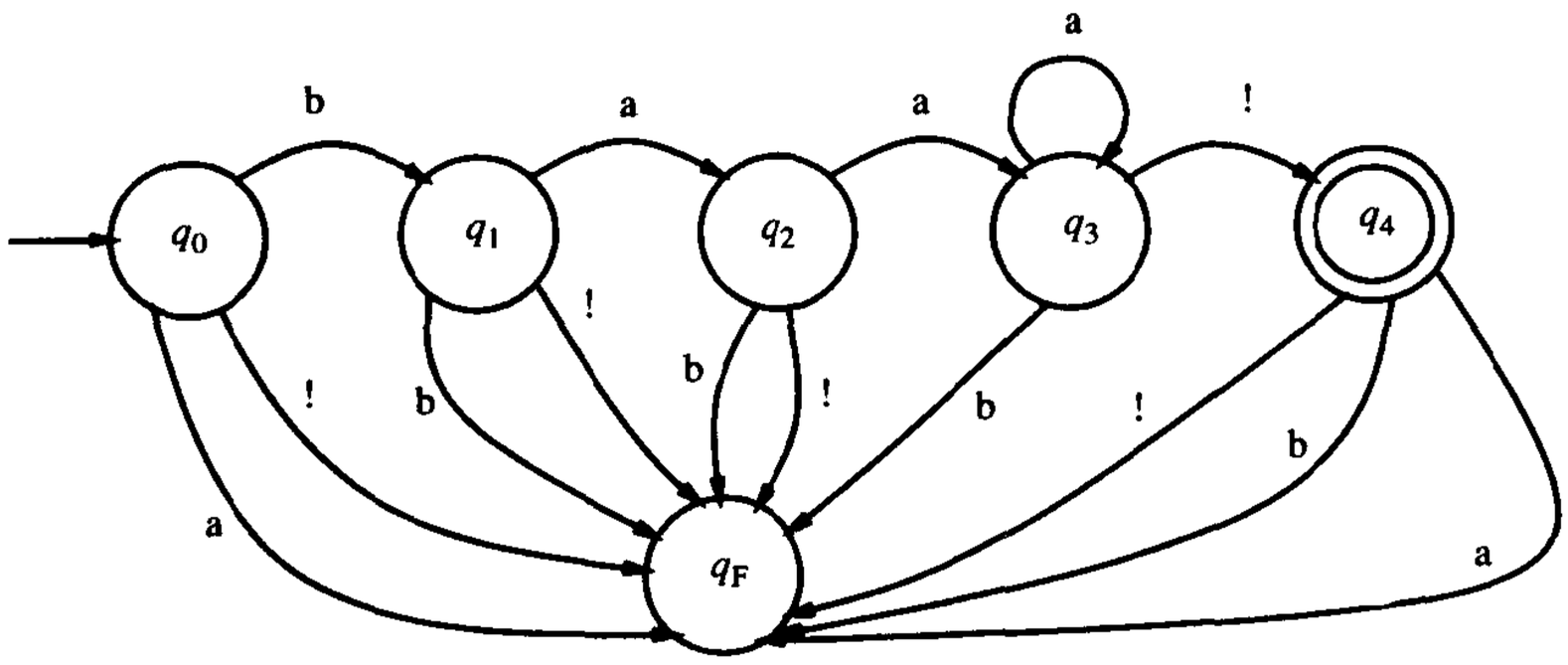


图 2.15　给图 2.10 中的自动机再添加一个失败状态

2.2.2　形式语言

我们可以用图 2.10 中的同一个状态图来生成（generating）羊的语言。如果这样做，应让自动机从状态 q_0 开始，穿过弧而到达新的状态，并且逐一地把标在相应的弧上的符号显示出来。当自动

机到达终极状态时，它就停止了。需要注意的是，在状态q_3时，自动机将面临选择的局面：它或者显示一个!并进入状态q_4，或者显示一个a而返回到状态q_3。这时，无论自动机做什么样的决策，也许可以用翻转硬币的方法来做出决策。这样一来，我们不管羊的语言的精确符号串是怎样的，只要它是羊的语言中上述正则表达式所允许的符号串就行了。

关键概念#1　形式语言（formal language）：它是一个模型，这个模型能够而且只能够生成或识别满足形式语言定义所要求的某一形式语言的符号串。

形式语言是符号串的集合，而每个符号串由称为字母表（alphabet，与前面用来定义自动机的字母表相同！）的有限的符号集合组合而成。羊的语言的字母表就是集合$\Sigma = \{a, b, !\}$。给定一个模型m（这是一个特定的FSA），我们可以使用$L(m)$来表示“由m刻画的形式语言”。所以，图2.10（或者图2.12）中的羊的语言自动机所定义的形式语言就是无限集合：

$$L(m) = \{\text{baa!,baaa!,baaaa!,baaaaa!,baaaaaa!,...}\} \quad (2.1)$$

这样定义语言自动机的用处在于它能够在一个封闭的形式中表示无限的集合（如上面的例子所示）。形式语言与自然语言（natural language）不同，自然语言是现实中人们所说的语言。事实上，形式语言可能与现实语言完全不同（例如，形式语言可以用来模拟一个汽水机的不同状态）。然而，我们通常使用形式语言来模拟自然语言的某些部分，例如音系、形态、句法的某些部分。在语言学中，生成语法（generative grammar）这个术语有时就用来表示形式语言的语法，这个术语的来源就是使用自动机来定义能够生成一切可能的符号串的语言。

2.2.3　另外的例子

在前面的例子中，我们的形式字母表只包含字母，然而还可以有包含单词的更高级的字母表。用这样的字母表，能够写出有限状态自动机来模拟关于单词的组合。例如，我们可以写出一个自动机FSA来模拟英语中表示钱的数量的那一部分。这样的形式语言可以模拟包含诸如*ten cents*, *three dollars*, *one dollar thirty-five cents*等短语的英语的某个子集合。

我们分几步来解决这个问题。首先，建立一个能够数1~99等数字的自动机。我们可以用这些数字来计算分币（cent），如图2.16所示。

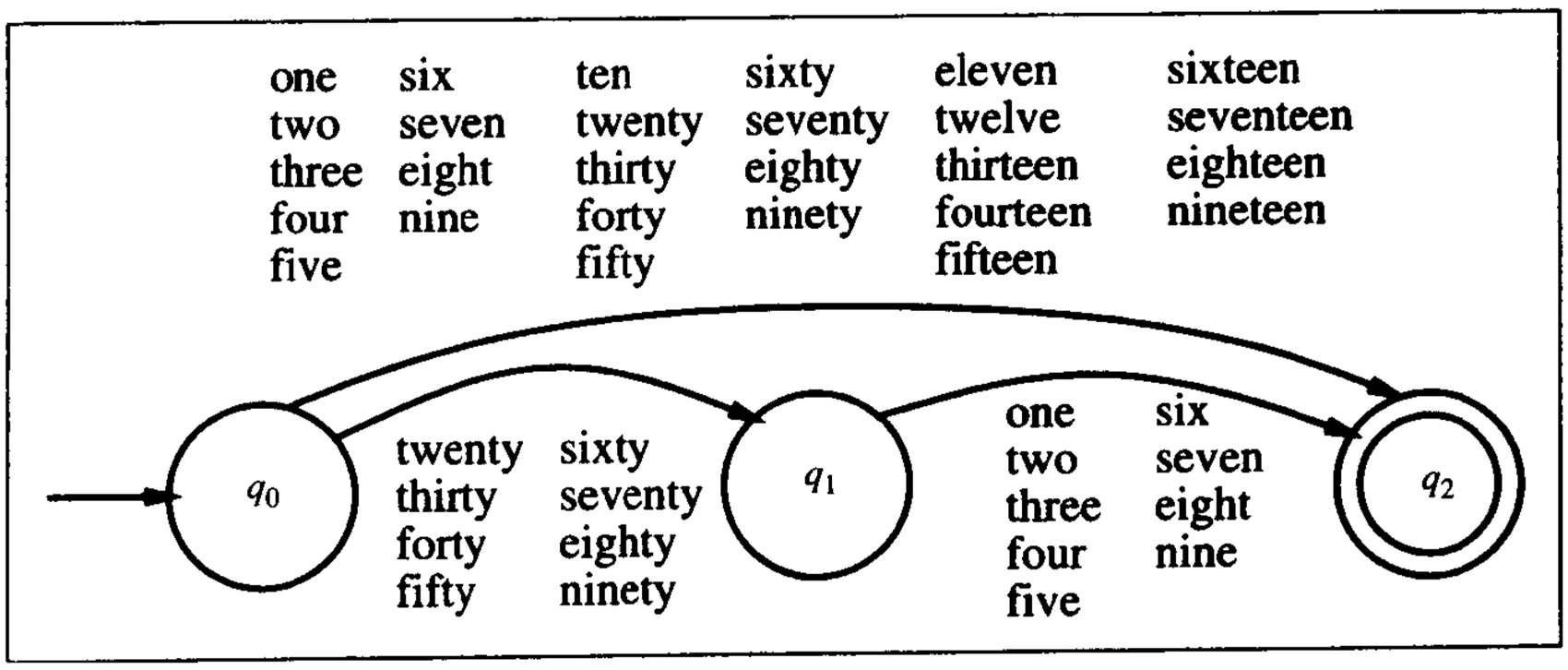


图2.16　刻画表示英语1~99的数字的数词FSA

然后，我们在这个自动机中再加上*cents*（美分）和*dollars*（美元）。图2.17描述了一个简单的办法。我们只要把图2.16中的自动机复制两个，并在其中的适当位置加上*cents*和*dollars*等单词。

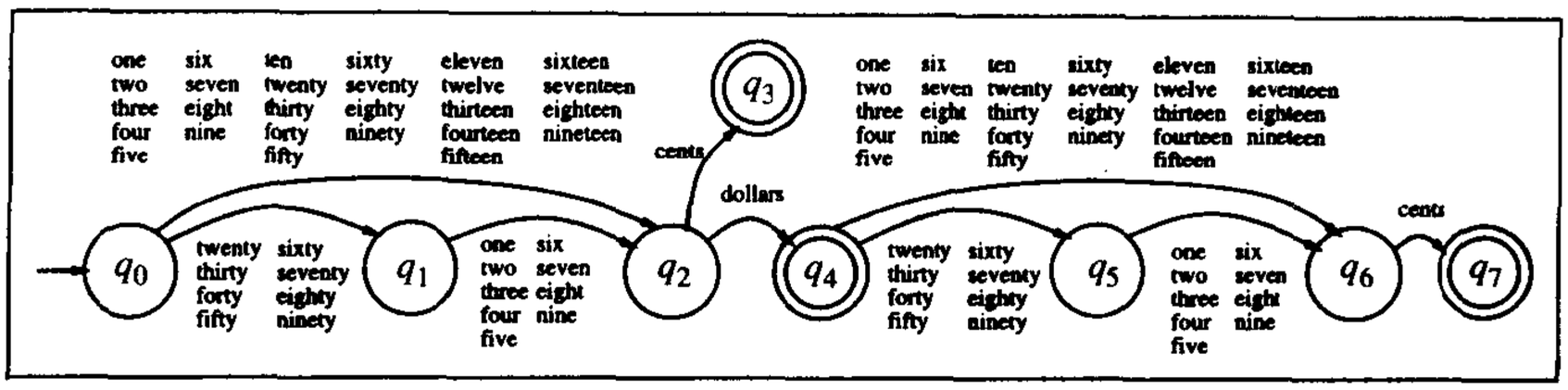


图 2.17　表示简单的 cents 和 dollars 的 FSA

为了表示不同的美元数，把 *hundred*（百）和 *thousand*（千）这样的较大数目的词也包括在内，我们现在需要在语法中添加一些东西。另外，*cents* 和 *dollars* 这样的单词还可以有单数形式（如 *one cent* 和 *one dollar*）和复数形式（如 *ten cents* 和 *two dollars*）的区别，我们要适当地改变它们的词尾。我们将把这个问题留做练习让读者自己来解决。我们把图 2.16 和图 2.17 中的 FSA 看成是英语的一个非常简单的语法。在本书的第二部分，特别是在第 9 章中，我们将回过头来再讨论语法的构建问题。

2.2.4　非确定 FSA

现在让我们来讨论另一种 FSA：非确定的有限自动机（non-deterministic FSA 或 NFSA）。我们来研究图 2.18 中的羊的语言的自动机，这个自动机与前面讨论过的图 2.10 中的自动机十分相似。

这个自动机与前面图 2.10 中的自动机的惟一区别在于：图 2.18 中的自动机的自返圈在状态 2，而图 2.10 中的自动机的自返圈在状态 3。我们也可以使用这个自动机来识别羊的语言。当到达状态 2 的时候，在输入带子上看到一个 a，我们难以判断是沿着自返圈返回到状态 2，还是继续往前进入状态 3。带有这种两难判定点的自动机称为非确定的 FSA（或 NFSA）。相比之下，图 2.10 中的自动机则是确定的（deterministic）自动机。当确定的自动机在识别时，当它处于某个状态以及要查找某个符号时，它的行为是完全确定的。确定的自动机可以表示为 DFAS（Deterministic FSA），而图 2.18 中的自动机（NFSA #1）则是非确定的。

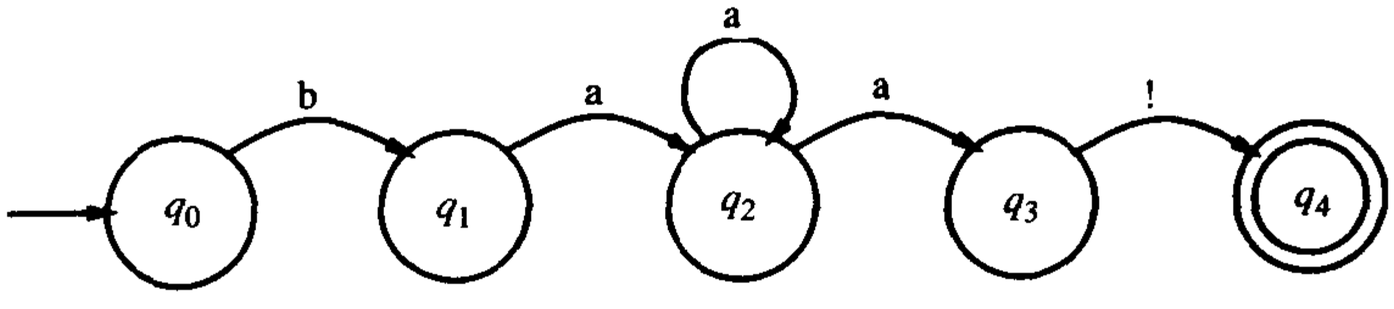


图 2.18　描述羊的语言的非确定的自动机（NFSA #1）。可以与图 2.10 中的确定的自动机进行对比

还有另一种类型的非确定的自动机，这种非确定的自动机是由没有符号的弧（称为 ε- 转移 [ε-transition]）引起的。图 2.19 中的非确定自动机可以处理前面分别描述过的羊的语言，不同的是这个自动机使用了 ε- 转移。

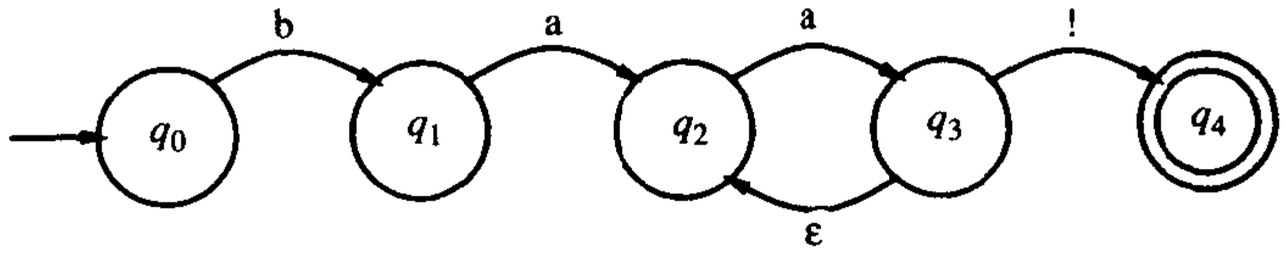


图 2.19　处理羊的语言的另一个 NFSA（NFSA #2）。它与图 2.18 中的 NFSA #1 的不同之处在于使用了 ε- 转移

对于这个 ε- 转移的新弧，我们可以这样来解释：如果到达了状态 3，可以不看输入带子上的符号就转移到状态 2，或者说可以向前推进输入指针。所以，这是另一种类型的非确定性。我们不用知道是做 ε- 转移还是进入带有标记 ! 的弧。

2.2.5 使用 NFSA 接收符号串

如果我们想知道一个符号串是不是属于前面所说的羊的语言的一个实例，并且又要使用非确定的自动机来识别这个符号串，就有可能顺着错误的弧走下去，而当我们试图接收这个符号串时，自动机却拒绝了它。在这种情况下，由于在某个点上存在着一种以上的选择可能性，我们有可能做出错误的选择。当我们建立计算模型时，在非确定的自动机模型中，特别是对于剖析来说，这样的选择问题总是一个又一个地不断发生。

对于这样的问题，存在着三种解决办法：

- **回退**（backup）：每当我们走到这样的选择点时，我们可以做一个记号（marker），记录下在输入中的什么位置以及自动机处于什么状态。当确认我们确实是做了错误的选择时，可以退回去并试探其他的路径。
- **前瞻**（look-ahead）：我们可以在输入中向前看，以帮助判定应该选择哪一条路径。
- **并行**（parallelism）：每当我们走到选择点时，我们可以并行地查找每一条不同的路径。

这里，我们着重讨论回退的方法，而对于前瞻和并行的方法，将在后面的章节中讨论。

所谓“回退方法”，就是当我们知道了总是能够返回到未曾探测过的选择时，就会无忧无虑地改变原来的选择，返回到未曾探测过的选择，以避免走进死胡同。这种方法有两个关键点。第一，在每个选择点上必须记住所有不同的选择；第二，对于每一个不同的选择必须存储足够的信息，以便在需要时能够返回。当返回算法到达某一个点而其进程不能再进展时（或者因为输入已经读完，或者因为不再有合法的转移），算法就要返回到前一个选择点，选择一个未曾探测过的点，并由此继续进行探测。把这样的概念应用于我们的非确定识别器。在每一个选择点上，我们只需要记住两个东西：第一，机器将进入的状态或结点；第二，输入带子上相对应的位置。我们把结点与位置的结合体称为识别算法的搜索状态（search-state）。为了避免混淆，我们把自动机的状态称为结点（node）或者机器状态（machine-state），以区别于搜索状态。图 2.21 给出了基于这种方法的识别算法。

在描述这个算法的主要部分之前，我们需要注意驱动这个算法的转移表中有两个重要的改变。首先，为了表示 ε-转移的结点，我们在转移表中加了一个新的列，即 ε-列。如果一个结点有 ε-转移，那么就在 ε-列与该结点所在的行的交接处，标出 ε-转移所指的方向上的相应结点。其次，为了表示从同一个输入结点到多个结点的多重转移，我们允许在转移所指的方向结点处出现由一个以上的结点组成的一个方向结点表，而不只是一个单独的结点。图 2.20 就是与图 2.18 中的自动机（NFSA #1）相应的转移表，其中没有 ε-转移，但对于机器状态 q_2 和输入符号 a，可以转移回 q_2，或者转移到 q_3。

	输入			
状态	b	a	!	ε
0	1	Ø	Ø	Ø
1	Ø	2	Ø	Ø
2	Ø	2,3	Ø	Ø
3	Ø	Ø	4	Ø
4:	Ø	Ø	Ø	Ø

图 2.20 图 2.18 中 NFSA #1 的转移表

图 2.21 是使用非确定 FSA 来识别输入符号串的算法。这个 ND-RECOGNIZE 的功能中使用了称为进程表（agenda）的变量来记录在处理过程中产生的所有尚未进行过的选择。每一个选择（搜

索状态）由自动机的一个结点（状态）和带子上的一个位置组合而成。变量 *current-search-state*(当前搜索状态）表示当前正在进行的选择。

```
function ND-RECOGNIZE(tape, machine) returns accept or reject

  agenda ← {(Initial state of machine, beginning of tape)}
  current-search-state ← NEXT(agenda)
  loop
    if ACCEPT-STATE?(current-search-state) returns true then
      return accept
    else
      agenda ← agenda ∪ GENERATE-NEW-STATES(current-search-state)
    if agenda is empty then
      return reject
    else
      current-search-state ← NEXT(agenda)
  end

function GENERATE-NEW-STATES(current-state) returns a set of search-states

  current-node ← the node the current search-state is in
  index ← the point on the tape the current search-state is looking at
  return a list of search states from transition table as follows:
    (transition-table[current-node,ε], index)
    ∪
    (transition-table[current-node, tape[index]], index + 1)

function ACCEPT-STATE?(search-state) returns true or false

  current-node ← the node search-state is in
  index ← the point on the tape search-state is looking at
  if index is at the end of the tape and current-node is an accept state of machine
then
    return true
  else
    return false
```

图 2.21　用于NFSA识别的算法。词*node*表示FSA的一个状态，词*state*或者*search-state* 表示“搜索过程的状态”，也就是 *node* 和 *tape-position* 的结合体

ND-RECOGNIZE首先建立一个初始的搜索状态，并把它放入进程表中。现在我们暂不说明搜索状态在进程表中的顺序。这个搜索状态包括自动机的机器状态和指向带子开始位置的指针。函数NEXT的功能是从进程表中检索一个项目并且把它赋给变量*current-search-state*（当前搜索状态）。

就像在D-RECOGNIZE中一样，主要回路的首要任务是确定输入带子上的全部内容是否都被自动机成功地识别了。这可以通过调用“ACCEPT-STATE?”来实现，如果当前搜索状态既包含一个接收的机器状态，也包含一个指向带子结尾的指针，那么就返回*accept*。如果不行，自动机就通过调用 GENERATE-NEW-STATE（生成新状态）来生成一系列可能的下一状态，这时

GENERATE-NEW-STATE要对转移表中的任何ε-转移和任何正常的输入符号转移创建搜索状态。所有这些搜索状态都要加到当前的进程表中。

最后，我们还要处理进程表中的新的搜索状态。如果进程表变空，我们无法进行选择，只好拒绝输入符号串，否则就选择其他还没有尝试过的可能性，把回路继续进行下去。

重要的是应该理解为什么ND-RECOGNIZE仅仅在进程表变空时返回一个拒绝的值。这与D-RECOGNIZE不同，当D-RECOGNIZE在非接收的机器状态时，或者当它发现自己不能从某个机器状态在带子中往前走而到达了带子终点时，它并不返回拒绝的值。其原因在于，在非确定的场合，这种道路阻塞的情况仅仅说明了在某条路径上失败了，但并不能说明它在所有的路径上都失败了。只有当检查了所有可能的选择，并且发现确实无路可走的时候，我们才可以很有把握地拒绝输入的符号串。

图2.22描述了ND-RECOGNIZE处理输入符号串“baaa!”的过程。每一个长条描述在处理过程中某个给定点上的算法的一个状态。变量*current-search-state*用一个粗线的圆圈来表示，它代表机器状态，箭头指向带子上的当前进程。图中的每个长条由上而下逐一表示从一个*current-search-state*到下一个*current-search-state*的进程。

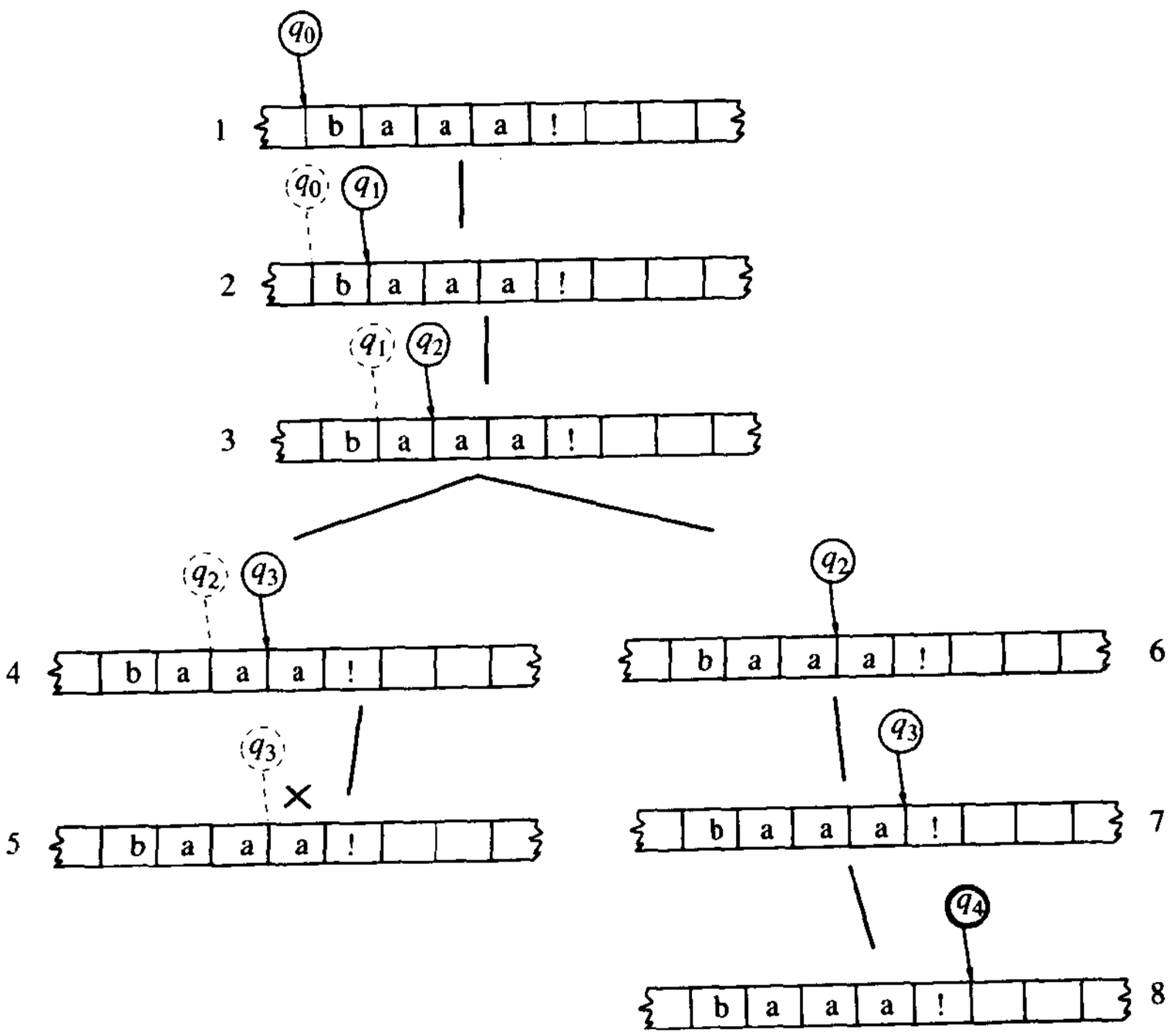


图2.22 NFSA #1（见图2.18）处理羊的语言的踪迹

当算法处于状态q_2时，它要查找带子上的第二个a，这时会出现一些有趣的事情。从转移表中我们可以看出，对于项目[q_2, a]，它可以返回到q_2和q_3。对于每个这样的选择，都要建立搜索状态，这些搜索状态都记录在进程表中。遗憾的是，这时我们的算法选择移动到状态q_3，这样移动的结果进入的不是一个接收状态，也不是任何新状态，因为在转移表中项目[q_3, a]进入的状态为空。在这一点上，算法需要简单地向进程表提问，看究竟还可以进入什么样的新状态。因为在这时的进程表中，从状态q_2返回到q_2是惟一没有检查过的选择，而这样的选择可以使带子上的指针向前推进到下一个a。但令人迷惑的是，ND-RECOGNIZE发现自己又面临着同样的选择。转移表中的项目

[q_2, a]仍然指出，不论返回到状态q_2，或者向前推进到状态q_3，都是可以允许的选择。如前所述，表示这两种选择的状态都要记录到进程表中。不过，这些搜索状态不同于前述的搜索状态，因为这时在带子上面输入符号串的索引值已经向前推进了。这时，进程表指示下面的移动就是移动到状态q_3。接着，根据带子和转移表惟一确定的操作，就是移动到q_4，算法获得成功。

2.2.6　识别就是搜索

ND-RECOGNIZE的任务在于，提出一种办法来系统地探索自动机中所有可能的路径，从而识别正则语言中的符号串。如果这样的探索找到了一条路径，以接收状态结束，那么自动机就接收该符号串，否则就拒绝它。使用进程表这样的机制，使得这样的探索成为可能。进程表在每次出现重复操作时，要选择探索一部分路径，并且要记住还没有探索过的那部分路径。

ND-RECOGNIZE的这种算法称为状态空间搜索（state-space search）算法，这种算法要系统地搜索问题的解。在这样的算法中，所要解决的问题被定义为一个空间，这个空间包含所有可能的解。算法的目标在于搜索这个空间，当发现了一个解时返回一个回答，或者当空间被穷尽地搜索完毕时，输入就被拒绝。在ND-RECOGNIZE中，搜索状态包括由机器状态和输入带子上的位置组成的偶对（pairing）。状态空间由给定自动机中一切可能的机器状态和带子位置的偶对所组成。搜索的目标就是从一个状态到另一个状态遨游这个空间，查找出由接收状态和带子位置的终点所组成的偶对。

提高这种程序的效率的关键通常在于状态在空间中的顺序。状态的顺序不好，可能会导致检查大量的无结果状态之后才能找到一个成功的解答。可惜在典型的情况下，我们并不能从一个不好的选择预见到好的选择，能够尽量做到的是让每个可能的选择都是经过充分考虑的。

细心的读者可能会注意到，我们还没有说明，在ND-RECOGNIZE中如何处理状态的顺序。我们只知道，当状态产生之后要把还没有探测过的状态加入进程表，并且当问到时给（还没有定义的）函数NEXT从进程表中返回一个未探测过的状态。怎样来定义函数NEXT呢？我们来研究一个处理顺序的策略，使下一次将要考虑的状态成为当前马上就要处理的状态。要实现这样的策略，需要把马上就要处理的状态放在进程表的前端，并且当调用函数NEXT时使NEXT返回到处于进程表中最前端的状态。这样，进程表就可以用栈（stack）来实现。这种策略通常被称为深度优先搜索（depth-first search）或后进先出（Last In First Out，简称LIFO）策略。

把这样的策略引进搜索空间，在当前的进程被卡住时只需返回到前一个状态。图2.22显示了ND-RECOGNIZE处理符号串“baaa!”的踪迹，这是一种深度优先搜索。当识别了ba之后，算法遇到了第一个选择点，它必须决定是停留在状态q_2还是推进到状态q_3。在这个选择点上，算法选择了其中的一个状态，继续往前走，直到发现这种选择是错误的时，算法才向后回溯到这个选择点，选择另一个比这个的状态稍老一些的状态。

深度优先策略有一个很大的缺点：在某些情况下可能会导致无限的环路。一种情况是当状态空间建立时搜索状态被再次访问；另一种情况是存在着无限数目的搜索状态。在第10章中，当我们研究了剖析中的一些更复杂的搜索算法之后，再回过头来讨论这个问题。

第二种处理搜索空间中状态顺序的办法是，按照状态建立时的顺序来进行处理。为了实现这样的策略，要把新建立的状态放在进程表的后面，函数NEXT在进程表的前端返回状态。这样，进程表就可以用队列（queue）来实现。这种策略通常被称为广度优先搜索（breadth-first search）或先进先出（First In First Out，简称FIFO）策略。图2.23显示了ND-RECOGNIZE处理符号串“baaa!”的另一个踪迹。在识别了ba之后，算法再次遇到了第一个选择点，它必须决定是停留在状态q_2，还

是推进到状态q_3。不过，这次算法不再只选择其中的一个状态并继续往前走，而是检查所有可能的选择，在同一时候对搜索树的同一个层面进行扩充。

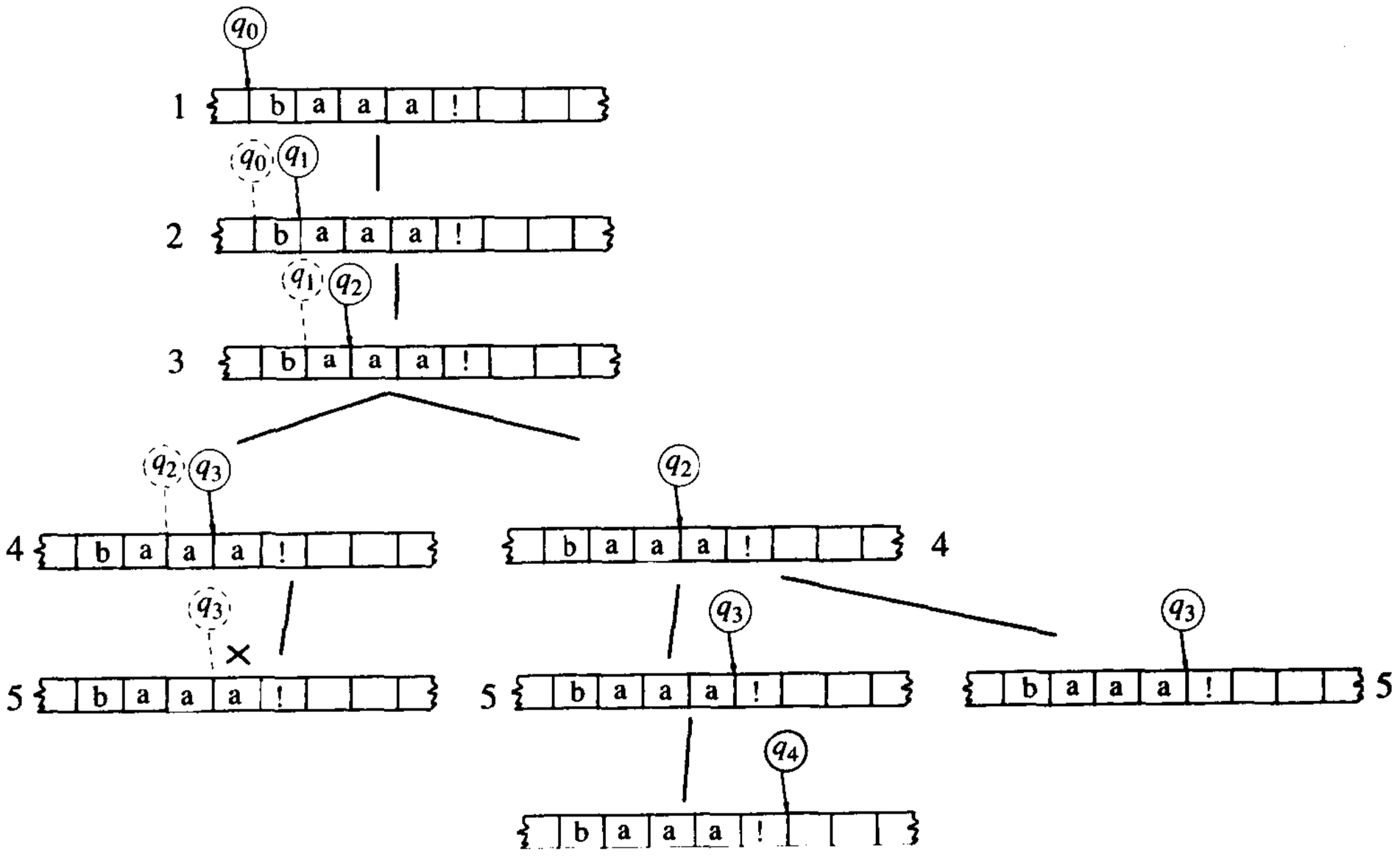


图 2.23　FSA #1 处理羊的语言的广度优先的踪迹

正如深度优先策略有缺点一样，广度优先策略也有缺点。使用深度优先策略，当状态空间是无限的时，搜索可能永不终止。更重要的是，由于进程表容量的增长，如果搜索空间比较大，搜索时就可能要求相当大的存储量，这样大的存储量实际上很难实现。对于规模比较小的问题，无论深度优先或广度优先策略都可能是合适的，尽管深度优先一般倾向于更有效地使用存储。对于规模很大的问题，就要使用更复杂的搜索技术，如动态规划（dynamic programming）和 A* 等技术，参见第 7 章和第 10 章。

2.2.7　确定自动机与非确定自动机的关系

允许NFSA具有诸如 ε -转移这样的非确定性特征似乎能够使NFSA比DFSA具有更强的能力，其实不然。对于任何 NFSA，都存在着一个完全等价的 DFSA。事实上，有一种简单的算法可以把 NFSA 转换成等价的 DFSA，当然在这个等价的确定自动机中的状态数目会多得多。关于这种等价关系的证明，可参阅 Lewis and Papadimitriou(1981)或 Hopcroft and Ullman(1979)。不过，这种证明的基本直觉在这里是值得一提的，并且这是按照NFSA剖析其输入的办法来做的。前面说过，NFSA 和 DFSA 之间的区别在于，在 NFSA 中的一个状态 q_i，对于给定的输入 i，下一步可能会存在着若干个状态（例如 q_a 和 q_b）。图 2.21 中的算法在处理这个问题时，或者选择 q_a，或者选择 q_b。如果选择被证明是错误的，就进行回溯（backtrack）。前面我们还说过，在并行算法中这两条道路（走向 q_a 和 q_b）是同时进行的。

把NFSA转换为DFSA的算法很像这种并行算法；我们建立一个有确定路径的自动机，对于每一条路径，我们的并行识别器都可以在搜索空间中跟踪。可以想像，我们同时跟踪两条路径，并且还要把到达同一个输入符号时所对应的各个不同状态（例如，q_a 和 q_b）合并成一个等价类。这时，我们对于这个新的等价类状态给出一个新的状态标记（例如标记为 q_{ab}）。对于每个可能的输入，对

于每一组可能的状态我们都继续进行这样的操作。这样得到的DFSA，由于要区别在原来的NFSA中的不同状态集合，它的状态数目可能会很多。因为具有N个元素的集合的不同的子集合数是2^N，所以新的DFSA具有2^N个状态。

2.3　正则语言与FSA

前面我们说过，由正则表达式所定义的语言类恰好也就是由有限自动机（无论是确定的还是非确定的）所刻画的语言类。由于这个原因，我们把这些语言都称为正则语言（regular language）。为了给正则语言的类一个形式的定义，我们需要参照前面的两个概念：一个概念是字母表Σ，它是语言中所有符号的集合；另一个概念是空符号串ε，按规定它不能包括在Σ中。此外，我们还要用到空集$\emptyset$（与ε不同）的概念。在Σ上的正则语言的类（或者正则集）可以形式地定义如下：[①]

1. $\emptyset$是正则语言
2. $\forall a \in \Sigma \cup \varepsilon, \{a\}$是正则语言
3. 如果L_1和L_2是正则语言，那么：
   (a) $L_1 \cdot L_2 = \{xy \mid x \in L_1, y \in L_2\}$，即$L_1$和$L_2$的毗连（concatenation）也是正则语言
   (b) $L_1 \cup L_2$，即L_1和L_2的合取（union）与析取（disjunction）也是正则语言
   (c) L_1^*，即L_1的Kleene闭包也是正则语言

只有满足上述性质的语言的集合才是正则语言。由于正则语言是由正则表达式所刻画的语言的集合，所以本章介绍的所有正则表达式的算符（存储器除外）都可以用上面定义正则语言的三种运算来实现：这三种运算是毗连、析取/合取（也称为结合“|”），Kleene闭包。例如，所有的计数符（*, +, {n, m}）都是迭代Kleene闭包的特殊情况。所有的锚号都可以被想像成独立的特殊符号。方括号[ ]是一种析取。例如，[a, b]意味着“a或者b”，或意味着a和b的析取。也就是说，任何正则表达式都能转换成一个（可能很大的）表达式，这个表达式只使用三种基本的运算。

正则语言对于下列的运算也是封闭的（Σ^*表示由字母表Σ所构成的一切可能的符号串的无限集合）：

- **交（intersection）**：如果L_1和L_2是正则语言，那么同时包含在L_1和L_2中的符号串的集合所构成的语言也是正则语言，即$L_1 \cap L_2$也是正则语言。
- **差（difference）**：如果L_1和L_2是正则语言，那么包含在L_1而不包含在L_2中的符号串的集合所构成的语言也是正则语言，即$L_1 - L_2$也是正则语言。
- **补（complementation）**：如果L_1是正则语言，那么不包含在L_1中的所有可能的符号串的集合所构成的语言也是正则语言，即$\Sigma^* - L_1$也是正则语言。
- **逆（reversal）**：如果L_1是正则语言，那么由L_1中的所有符号串的逆的集合所构成的语言也是正则语言，即L_1^R也是正则语言。

关于正则表达式与自动机等价的证明参见Hopcroft and Ullman(1979)。证明包括两部分，第一部分证明对于每个正则语言都可以建立一个自动机，第二部分又反过来证明对于每个自动机都可以建立一个正则语言。这里不给出此类证明，但是我们要通过直觉来说明怎么来证明第一部分，即如何从任何正则表达式来建立自动机。我们的直觉解释是归纳性的：作为基底，我们对应于包含一个单独符号的正则表达式（例如，表达式a）来建立相应的自动机，创建一个初始状态和一个可接收

① 依据 van Santen and Sproat(1998)，Kaplan and Kay(1994)，Lewis and Papadimitriou,(1981)。

的终极状态，把这两个状态之间的弧标为a。在归纳步骤中，我们将说明正则表达式的每个这样的基本操作（毗连、结合和闭包）都可以通过自动机来模拟。

- **毗连**（concatenation）：我们使用ε-转移，把FSA_1的终极状态与FSA_2的初始状态连接起来，如图2.24所示。
- **闭包**（closure）：我们使用ε-转移，把FSA的所有终极状态回过头去与它的初始状态连接起来（实现Kleene*的迭代部分），然后使用ε-转移，建立初始状态和终极状态之间的直接联系（实现可能出现的零转移），如图2.25所示。这里不讨论如何实现Kleene+。
- **结合**（union）：我们增加一个新的初始状态q'_0，并且从这个状态q'_0，用新的转移把它和两个自动机中要联合的初始状态连接起来，如图2.26所示。

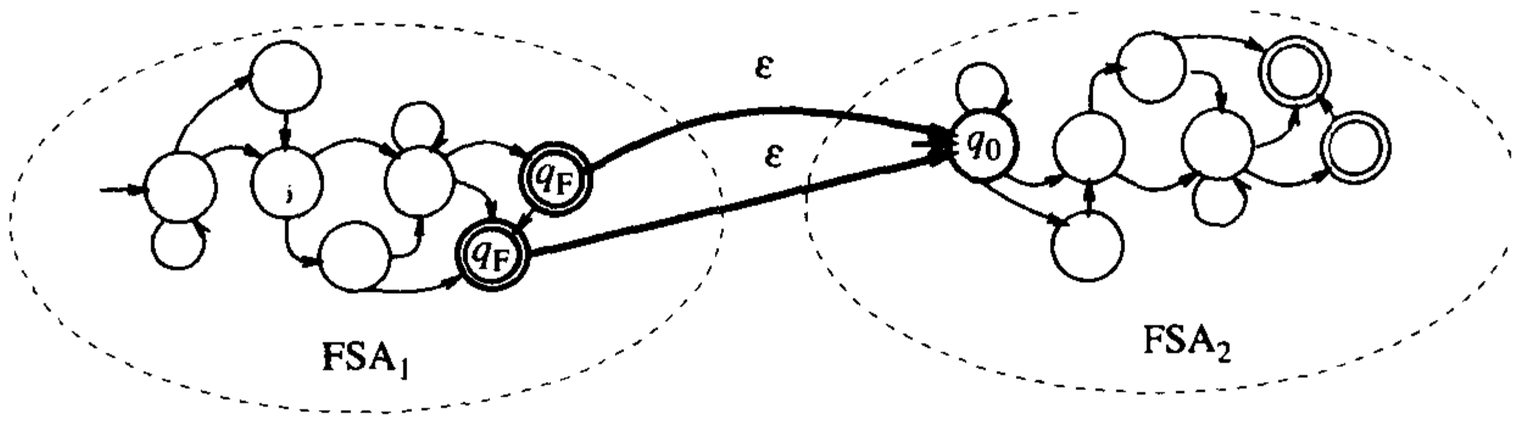


图2.24 两个自动机的毗连

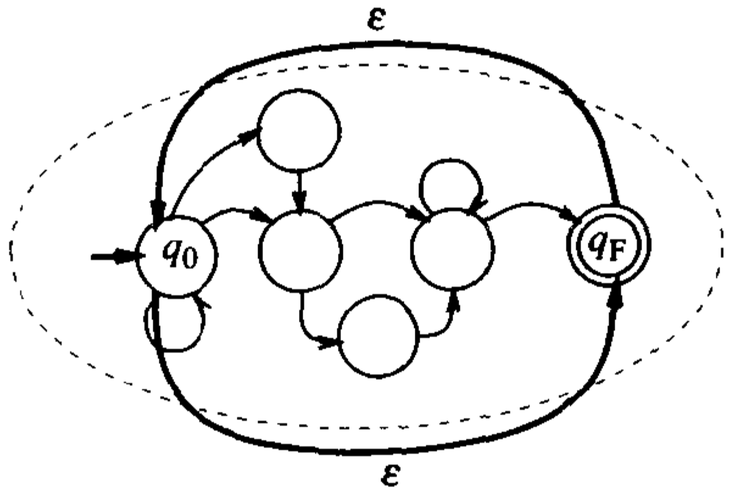


图2.25 FSA的闭包（Kleene*）

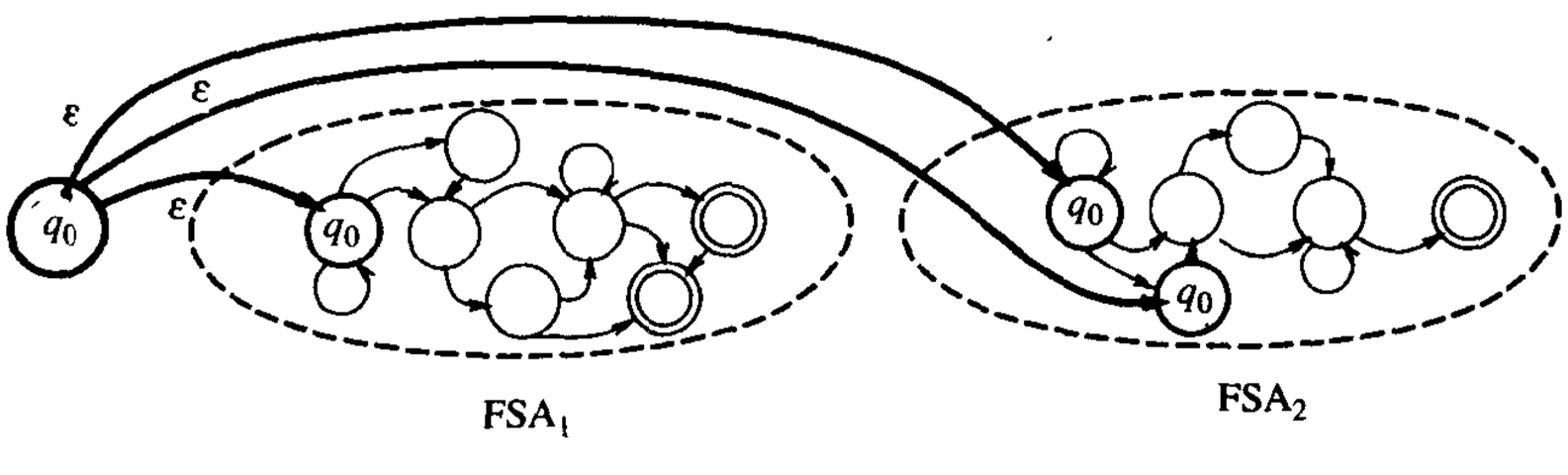


图2.26 两个FSA的结合（I）

2.4 小结

本章介绍了自然语言处理的最重要的基本概念，即自动机的概念，还介绍了一种基于自动机的实用工具，即正则表达式。下面总结了关于这些内容的要点：

- 正则表达式语言是模式匹配的有力工具。
- 正则表达式的基本运算包括符号的毗连、符号的析取（[]，|以及.）、计数符（*，+以及{n,m}），锚号（^和$）和前于关系运算（(和)）。

- 任何正则表达式都可以实现为一个有限状态自动机。
- 存储器（\1以及()）是一种高级运算，它经常作为正则表达式的一部分，但不能实现为有限自动机。
- 自动机把形式语言定义为隐含地被自动机所接收的符号串的集合。
- 自动机可以使用任何符号集合作为它的词汇，包括字母、单词甚至图形。
- 确定的自动机（DFSA）的行为完全由它的状态来决定。
- 非确定的自动机（NFSA）对于相同的当前状态和下一状态，有时必须在多条路径之间进行选择。
- 任何一个NFSA都可以转换为一个DFSA。
- NFSA在进程表中探索下一个状态的顺序决定了它的搜索策略。深度优先和后进先出策略相当于把进程表看成栈；宽度优先和先进先出策略相当于把进程表看成队列。
- 任何正则表达式都可以被自动编译为NFSA，因此也就可以被编译为FSA。

2.5　文献和历史说明

20世纪50年代产生的有限自动机的理论来源于Turing(1936)的算法计算模型，很多人都认为这种模型是现代计算机科学的基础。Turing机器是具有一个有限控制器和一个输入/输出带子的抽象机器。在一次移动时，Turing机器能够读带子上的一个符号，在带子上写不同的符号，改变状态并向左或向右移动。可见，Turing机器不同于有限状态自动机之处在于它有改变带子上的符号的能力。

在Turing工作的鼓舞之下，McCulloch和Pitts研制了类似于自动机的神经元（neuron）模型参见 von Neumann(1963, p.319)。现在这个模型通常称为McCulloch-Pitts神经元模型(McCulloch and Pitts, 1943)。这个模型是关于神经元的一个简化模型，它把神经元看成是可以用命题逻辑来描述的某种类型的“计算单元”（computing element）。这种神经元模型是二元的装置。在任何一个点上，无论该点是否被激发，都可以从另外的神经元得到激发性的或抑制性的输入。而如果激发的程度超过一定的阈值（threshold），这个神经元就被激活。在McCulloch-Pitts的神经元模型的基础上，Kleene (1951, 1956)定义了有限自动机和正则表达式，并且证明了二者的等价性。非确定的自动机是Rabin and Scott(1959)提出的，并证明了非确定的自动机和确定的自动机之间的等价关系。

Ken Thompson是首先研制正则表达式编译器的学者之一，他使用正则表达式编译器进行文本搜索工作(Thompson, 1968)。他的文本搜索编辑器ed包含一个“g/regular expression/p”命令，或者称为通用正则表达式打印命令，后来变成了UNIX grep。

有不少关于自动机的数学理论的一般性入门读物，例如Hopcroft and Ullman(1979)和Lewis and Papadimitriou(1981)。本章讲述了简单自动机的数学基础，第3章讲述有限状态转录机，第9章讲述上下文无关语法，第13章讲述Chomsky层级分类。Friedl(1997)对于正则表达式的高级用法做了详尽的介绍，是关于这方面研究的非常有用的指导性读物。

把问题求解比喻为搜索是人工智能（AI）的基础；关于搜索的更详尽的介绍可以在任何一本人工智能教材中找到，例如Russell and Norvig(1995)。

第3章　形态学与有限状态转录机

A writer is someone who writes, and a stinger is something that stings. But fingers don't fing, grocers don't groce, haberdashers don't haberdash, hammers don't ham, and humdingers don't humding[①].

Richard Lederer, *Crazy English*

在第2章中我们介绍了正则表达式，举例说明了一个简单的搜索符号串怎样帮助搜索引擎同时找出 *woodchuck* 和 *woodchucks*。找出 *woodchuck* 的单数形式和复数形式很容易，复数只要把 *s* 加在词的尾部就行了。但是，假如我们要查找其他生活在森林中的动物，例如，*fox*（狐狸）、*fish*（鱼）、乖戾的 *peccary*（野猪，西貒）和加拿大的 *wild goose*（野鹅），就会出现问题，因为这些动物的复数形式并不是简单地在词尾加上一个 *s*。*fox* 的复数形式是 *foxes*，*peccary* 的复数形式是 *peccaries*，*goose* 的复数形式是 *geese*。更加令人迷惑的是，*fish* 的复数形式通常并不改变它在单数时的形式（正如 Dr. Seuss 指出的，我们可以有 *one fish, two fish, red fish, blue fish* 等）。

要正确地搜索这些词的单数和复数需要两种知识：拼写规则的知识和形态规则的知识。拼写规则（spelling rule）告诉我们，英语中以 *y* 为结尾的词，其复数要把 *-y* 变为 *-i-*，然后再加上 *-es*。形态规则（morphological rule）告诉我们，*fish* 的复数具有零形式，而 *goose* 的复数要改变元音。

把 *foxes* 分析为 *fox* 和 *-es* 两个语素的问题称为形态剖析（morphological parsing）。

关键概念 #2　剖析就是取一个输入并且产生关于这个输入的各类结构。

在本书中，我们是在广义上使用剖析这个术语的。剖析产生的结果可以是形态结构、句法结构、语义结构或语用结构；剖析产生的形式可以是符号串、树或网络。在信息检索中，把 *foxes* 映射为 *fox* 的类似的（但不是等同的）问题称为词干还原（stemming）。除了复数变化之外，形态剖析和词干还原还可以应用于处理词缀。例如，我们可以处理以 *-ing* 结尾的英语动词（*going, talking, congratulating*），把它们剖析为动词词干再加上 *-ing* 这个语素。因此，给出表层形式或者输入形式 *going*，就可以产生出剖析形式 VERB-go + GERUND-ing。本章将研究不同语言中需要的各种形态学知识，并且介绍形态剖析中的一种重要算法的主要部分，即有限状态转录机（finite-state transducer）。

为什么在词典中不列出英语名词的所有复数形式和动词的所有 *-ing* 形式呢？其主要的原因是 *-ing* 是一个普遍使用的有能产性（productive）的后缀；也就是说，它可以用于英语中的所有动词。类似地，*-s* 可用于所有名词。因此，列出每个名词和动词的所有形式的想法会使效率降低。进一步说，能产性的后缀还可以用于新词（所以新词 *fax* 会自动使用 *-ing* 形式，从而形成 *faxing*）。由于每天都会产生新词（特别是首字母缩写词和专有名词），英语名词的类也是与日俱增的，我们需要有能力把复数语素 *-s* 加到每个新词上。另外，这些新词的复数形式与它们的单数形式的拼写或发音有关系；例如，如果名词以 *-z* 结尾，那么它的复数形式就是加 *-es* 而不是加 *-s*。我们有必要对这些规则进行某种编码，使之系统化。最后，在一些形态复杂的语言中，我们绝对不可能列举出每个单词的全部形式。例如，土耳其语的单词形式如下：

---

① 本章的这一段开场白讲的是英语的构词法的一些实例，如果翻译成中文反而让人看不懂，所以我们选择了提供原文。读者从英文原文不难理解这段开场白的含义。——译者注

uygarlaştıramadıklarımızdanmışsınızcasına　(3.1)

uygar *+laş* *+tır* *+ama* *+dık* *+lar* *+ımız*

civilized +BEC +CAUS +NEGABLE +PPART +PL +P1PL

+dan *+mış* *+sınız* *+casına*

+ABL +PAST +2PL +AsIf

这个土耳其语单词的英语解释是：(behaving) as if you are among those whom we could not civilize/cause to become civilized。它的中文意思是举止不文明。

这个单词中的各个片断（语素）的含义分别如下：

+BEC	英语的 *become*
+CAUS	动词的使役态标志
+NEGABLE	英语的 *not able*
+PPart	表示过去分词形式
+P1PL	表示第一人称复数物主一致关系
+2PL	第二人称复数
+ABL	离格（*from/among*）标志
+AsIf	从限定式动词形式变为副词的派生标志

在这样的语言中显然应该对输入进行剖析，因为不可能存储每个可能存在的词。Kemal Oflazer（通过私人通信）提供了这个例子。他指出，如果不考虑派生后缀，那么土耳其语中的动词有40 000个形式；如果把派生后缀计算在内，那么从理论上说土耳其语中的词的数目将是无限的。这种说法是可信的，因为任何动词都可能像上面的例子中的词一样“成为导因”，而多重的导因可能嵌入到一个单词中（*You cause X to cause Y to ... do W*）。这种单词可能存在的事实意味着，要事先把土耳其语中的所有单词都存储起来是非常困难的。

形态剖析不仅仅对于信息检索是必要的，在机器翻译中也需要形态剖析，以便把法语中的 *va* 和 *aller* 两者都翻译为英语的动词 *go*。在拼写检查中，我们在下面将会看到，形态学的知识使我们知道，*misclam* 和 *antiundoggingly* 不是词。

下一节将总结英语的形态事实，然后介绍有限状态转录机（finite-state transducer）。

3.1　英语形态学概观

形态学研究如何从比较小的意义单位（语素）构成词。语素（morpheme）通常被定义为语言中负荷意义的最小单位。例如，*fox* 这个词只包含一个单独的语素（即语素 *fox*），而 *cats* 这个词则包含两个语素：一个是语素 *cat*，一个是语素 *-s*。

刚才的例子还说明，把语素分为词干（stem）和词缀（affix）两大类通常是很有帮助的。这种区分的细节会因语言的不同而不同，但在直观上我们会觉得词干是词中的主要语素，它提供主要的意义，而词缀则提供各种类型的“附加”意义。

词缀还可以进一步分为前缀（prefix）、后缀（suffix）、中缀（infix）和位缀（circumfix）。前缀位于词干之前，后缀紧接在词干之后，位缀则同时处于词干的前面和后面，中缀则插入到词干中。例如，单词 *eats* 由词干 *eat* 和后缀 *-s* 组成。单词 *unbuckle* 由词干 *buckle* 和前缀 *un-* 组成。英语中找不到位缀的恰当例子，但是很多其他语言中有位缀。例如，在德语中，某些动词的过去分词通过在词干前面加 *ge-* 并在词干后面加 *-t* 构成；所以，动词 *sagen*（说）的过去分词是 *gesagt*（说过了）。中

缀要插入到单词的中间，这样的中缀在菲律宾语的Tagalog语中是很普遍的。例如，在Tagalog语中，表示行为施事者的词缀*um*就是一个中缀，它插入到词干*hingi*（借）中形成*humingi*。在英语的某些方言中也存在一个中缀，用于表示禁忌语。例如，“f**king”或“bl**dy”等，它们被插入到其他词的中间（“Man-f**king-hattan”，“abso-bl**dy-lutely”）(McCawley, 1978)[①]。

前缀和后缀这样的语素通常称为毗连性语素（concatenative morphology），因为词是由一定数目的语素毗连在一起而组合成的。有些语言中存在着很多非毗连性语素（non-concatenative morphology），其中语素的组合方式极为复杂。在上述例子中，Tagalog语的中缀就是非毗连性语素，因为*hingi*和*um*这两个语素是混杂在一起的。还有另外的非毗连性语素，称为模板语素（templatic morphology）或词根与模式语素（root-and-pattern morphology）。它们在阿拉伯语和其他闪美特语系（Semitic language）的语言中是普遍存在的。例如，在希伯来语中，动词通常由词根（root）和模板（template）两个部分构成。词根通常由3个辅音（CCC）组成，负荷基本意义，模板给出辅音和元音的顺序的信息，并进一步给出更多的语义信息，例如关于语义态（主动态、被动态和中动态）的信息，最后形成整个动词。例如，希伯来语中的三辅音词根*lmd*的意思是“学习”或“教学”，它可以和主动态模板CaCaC结合起来形成单词*lamad*（“他学习”），或者与强化模板CiCeC结合起来形成单词*limed*（他教过学），或者与强化被动态模板CuCaC结合起来形成单词*lumad*（他被教过）。

一个单词可以具有一个以上的词缀。例如，单词*rewrites*具有前缀*re-*，词干*write*和后缀*-s*。单词*unbelievably*具有一个词干以及三个词缀（*un-*，*-able*和*-ly*）。英语中的词缀一般不超过4个或5个。前面说过，土耳其语中存在带9个或10个词缀的单词。像土耳其语这样倾向于把词缀一个一个地连起来的语言称为黏着语（agglutinative language）。

从语素构成单词的方法可以分为两大类（可能部分地交叉）：屈折（inflection）和派生（derivation）。屈折把词干和一个语法语素结合起来，所形成的单词一般与原来的词干属于同一个词类，还会产生一些诸如“一致关系”之类的句法功能。例如，英语的屈折语素*-s*表示名词的复数，英语的屈折语素*-ed*表示动词的过去时态。派生也把词干和一个语法语素结合起来，所形成的单词一般属于不同的词类，产生的新意义经常难以精确地预测。例如，动词computerize可以加上派生后缀*-ation*，形成名词computerization。

3.1.1 屈折形态学

本节讨论屈折形态学（Inflectional Morphology）。英语具有相对简单的屈折系统。只有名词、动词和部分形容词有屈折变化，可能的屈折词缀的数目也很少。

英语的名词只有两个屈折变化：一个词缀表示复数（plural），一个词缀表示领属（possessive）。例如，英语的很多名词（但不是全部）或者以单独的词干或单数形式出现，或者出现时还带有一个复数后缀。下面的例子分别表示规则的复数后缀*-s*、变异的拼写形式*-es*以及不规则的复数形式：

	规则名词		不规则名词	
单数	cat	thrush	mouse	ox
复数	cats	thrushes	mice	oxen

大多数名词是用规则复数，拼写时在名词后面加*-s*，在以*-s*（*ibis/ibises*），*-z*（*waltz/waltzes*），*-sh*（*thrush/thrushes*），*-ch*（*finch/finches*）以及（有时）*-x*（*box/boxes*）为结尾的名词后面，都

① 创作“My Fair Lady”（我的美丽女郎）的抒情诗人Alan Jay Lerner把这些粗话从抒情诗“Wouldn't It Be Loverly”中的*abso-bloomin'lutely*删除了(Lerner, 1978, p.60)。

加-*es*，如前面括号中的例子所示。以-*y*为结尾的名词，当-*y*前面是一个辅音时把-*y*改为-*i*（*butterfly/butterflies*）。

对于领属后缀，规则单数名词（*llama's*）和不以-*s*结尾的复数名词（*children's*）是通过加“省略符（′）+ -*s*”来实现的；在规则复数名词后面（*llamas*′）以及某些以-*s*或-*z*结尾的人名后面（*Euripides*′ *comedies*），通常只加一个单独的省略符，如前面括号中的例子所示。

英语动词的屈折变化比名词的屈折变化复杂得多。首先，英语有三类动词：主要动词（如*eat, sleep* 和 *impeach*）、情态动词（如 *can, will* 和 *should*）和基础动词（如 *be, have* 和 *do*）。我们在这里使用了Quirk et al. (1985)中的术语。在本章中我们将集中讨论主要动词和基础动词，因为它们具有屈折词尾。这些动词的大部分是规则的，也就是说，这些规则动词具有同样的词尾，表示同样的功能。这些规则动词（例如，*walk, inspect*）有如下4种形态形式：

形态形式的类别	规则屈折动词			
词干	walk	merge	try	map
-*s*形式	walks	merges	tries	maps
-*ing*分词	walking	merging	trying	mapping
过去形式或-*ed*分词	walked	merged	tried	mapped

这些动词称为规则动词，因为只要知道了词干就能够预见到它的其他形式，在词干后面分别加上三个可预见的词尾 -*s*, -*ing* 和 -*ed*，然后再进行某些有规律的拼写变化（关于有规律的语音变化，参见第4章），就可以预见到它的所有形式。这些规则动词和形式在英语形态学中是有意义的，首先是因为它们涵盖了英语动词的大多数，其次是因为这样的规则类别是能产的。前面说过，这一类能产的词能够自动包容进入语言中的任何新词。例如，最近新创的动词*fax*（*My mom faxed me the note from cousin Everett*）可以自动加上规则的词尾 -*ed*, -*ing* 和 -*es*。注意，-*s*形式被拼写为*faxes*，而不拼写成*faxs*；我们将在下面讨论拼写规则。

非规则动词是那些在屈折变化时具有或多或少的惯用句法形式的动词。英语中的非规则动词一般具有5个不同的形式，最多具有8个不同的形式（例如，动词*be*），最少具有3个不同的形式（例如，*cut* 或 *hit*）。在把它们分成很多小类时，Quirk et al. (1985)估计，如果不考虑情态动词，英语中只有250个不规则动词。Quirk的分类包括了语言中的一些高频度动词[①]。下面的表列出了某些非规则形式。注意，有一个非规则动词在其过去时形式中（又称为preterite），改变了它的元音（*eat/ate*），或者改变它的元音和某些辅音（*catch/caught*），或者根本就没有词尾（*cut/cut*）。

形态形式的类别	非规则屈折动词		
词干	eat	catch	cut
-*s*形式	eats	catches	cuts
-*ing*分词	eating	catching	cutting
过去形式	ate	caught	cut
-*ed*分词	eaten	caught	cut

这些形式在句子中的使用方法将在第8章至第12章中讨论，这里只做简要说明。-*s*形式用于“通常现在时”形式，以便把第三人称单数的词尾（*She jogs every Tuesday*）和其他的人称和数的形式区别开（I/you/we/they joy every Tuesday）。词干形式用于不定式中和某些动词之后（*I'd rather walk home, I want to walk home*）。当动词作为名词来看待时，就采用 -*ing* 分词形式，这种把动词作为名词用的特殊形式称为动名词（gerund）用法（*Fishing is fine if you live near water*）。-*ed* 分词用于完

① 一般来说，频度越高的词越容易具有惯用的性质，这是由关于语言变化的事实决定的；频度非常高的词总是喜欢保持着它们惯常的形式，尽管它们周围的其他词正在变得越来越规范。

成时（perfect）结构中（*He's eaten lunch already*）或被动结构中（*The verdict was overturned yesterday*）。

此外还要注意什么样的后缀要连接到什么样的词干上。我们要认识到这样的事实：一些有规律的拼写变化总是发生在语素的边界上。例如，在加后缀*-ing*和*-ed*时，前面的单独辅音字母要重叠（*beg/begging/begged*）。如果最后一个字母是*-c*，则其重叠形式拼写为*-ck*（*picnic/picnicking/picnicked*）。正如在名词中那样，词尾*-s*加在以*-s*（*toss/tosses*），*-z*（*waltz/waltzes*），*-sh*（*wash/washes*），*-ch*（*catch/catches*）或*-x*（不常见，*tax/taxes*）结尾的动词词干后面时，*-s*要拼写为*-es*。像名词那样，以*-y*结尾的动词，如果它前面是一个辅音则把*-y*变为*-i*（*try/tries*）。

英语的动词系统比欧洲西班牙语的动词系统简单得多，西班牙语的每个规则动词都有50种不同的动词形式。图3.1中给出了动词*amar*（爱）的几个例子。其他语言还存在着比西班牙语例子中的形式更多的形式。

现在时陈述式	未完成过去时	未完成过去时陈述式	将来时	过去时	现在时虚拟式	条件式	未完成过去时虚拟式	将来时虚拟式
amo		amaba	amaré	amé	ame	amaría	amara	amare
amas	ama	amabas	amarás	amaste	ames	amarías	amaras	amares
	ames							
ama		amaba	amará	amó	ame	amaría	amara	amáreme
amamos		amábamos	amaremos	amamos	amemos	amaríamos	amáramos	amáremos
amáis	amad	amabais	amaréis	amasteis	améis	amaríais	amarais	amareis
	amáis							
aman		amaban	amarán	amaron	amen	amarían	amaran	amaren

图3.1 西班牙语中的动词*amar*（爱）的不同形式

3.1.2 派生形态学

本节讨论派生形态学（Derivational Morphology）。英语的屈折比其他语言相对简单，但英语的派生却相当复杂。前面说过，派生把词干和一个语法语素结合起来，所形成的单词一般属于不同的词类，产生的新意义经常难以精确预测。

英语中最普通的派生就是新的名词的形成，它们常常是从动词或形容词变来的。这个过程称为名词化（nominalization）。例如，后缀*-ation*可以从以后缀*-ize*结尾的动词构成名词（*computerize* → *computerization*）。下面是英语中几个能产性的名词化后缀的例子。

后缀	原来的动词/形容词	派生出的名词
-ation	computerize (V)	computerization
-ee	appoint (V)	appointee
-er	kill (V)	killer
-ness	fuzzy (A)	fuzziness

形容词也可以从名词和动词派生。下面是从名词和动词派生形容词的几个后缀的例子：

后缀	原来的名词/动词	派生出的形容词
-al	computation (N)	computational
-able	embrace (V)	embraceable
-less	clue (N)	clueless

英语的派生比屈折复杂，有若干个原因。首要的原因在于派生的能产性一般比较低；甚至像*-ation*这样的名词化后缀，尽管可以加在几乎全部以*-ize*结尾的动词后形成名词，但也不是绝对

地对于所有动词都行得通。因此，我们不能说 **eatation* 或 **spellation*（我们使用星号“*”来表示英语中不成立的例子）。另外的原因在于，名词化后缀之间往往有复杂的细微意义差别。例如，*sincerity* 的意义与 *sincereness* 的意义之间就有细微的差别。

3.2　有限状态形态剖析

现在我们讨论英语的形态剖析问题。我们来考虑一个简单的例子：能产性名词复数（*-s*）和动词进行式（*-ing*）的剖析。我们的目标是，取下面表中的第一列作为输入，能够产生第二列中的输出。

输入	形态剖析输出
cats	cat +N +PL
cat	cat +N +SG
cities	city +N +PL
geese	goose +N +PL
goose	(goose +N +SG)或者(goose +V)
gooses	goose +V +3SG
merging	merge +V +PRES-PART
caught	(catch +V +PAST-PART)或者(catch +V +PAST)

第二列中包含词干和有关的形态特征（feature）。这些特征说明了附加在词干上的有关信息。例如，+N 这个特征表示该词是名词，+SG 表示单数，+PL 表示复数。我们将在第 11 章中讨论特征，现在只把+SG 理解为一个基本单位，知道其意思是“单数”即可。注意，上面的某些输入形式（例如 *caught* 和 *goose*）在不同的形态剖析中是有歧义的。

为了建立一个形态剖析器，至少需要如下的东西：

1. **词表（lexicon）**：词干和词缀表及其基本信息（例如一个词干是名词词干还是动词词干等）。
2. **形态顺序规则（morphotactics）**：关于形态顺序的模型。它要解释，在一个词内什么样的语素跟在什么样的语素后面。例如，英语表示复数的语素跟在名词后面而不是前面。
3. **正词法规则（orthographic rule）**：正词法规则要说明，当两个语素结合时在拼写上发生什么变化（例如，前面讨论过的拼写规则 $y \rightarrow ie$ 就是关于 *city* + *-s* 要变为 *cities* 而不是 *citys* 的正词法规则）。

本节的下一部分将讨论，在形态识别这样的子问题中，简单词表的表示方法如何利用 FSA 给形态知识建模。然后我们将介绍有限状态转录机（finite-state transducer，FST），用它作为给词表的形态特征建模和进行形态剖析的一种工具。

3.2.1　词表和形态顺序规则

词表是存储词的宝库（repository）。最简单的词表应该给所描述的语言中的所有词列出一个清单，这个清单中还要包含缩写词（“AAA”）和专有名词（“Jane”，“Beijing”）。下面是这种词表的一个片断：

a
AAA
AA
Aachen

aardvark
aardwolf
aba
abaca
aback
…

然而，由于前面讨论过的种种原因，要建立这样的词表，把语言中所有的词都一一列举出来，通常很不方便或不可能。计算机词表通常是这样来构造的：它要列出语言中的每个词干和词缀，并且表示出形态顺序规则，告诉我们怎样把这些词干和词缀组合在一起。给形态顺序规则建模的方法有很多种，最常见的方法是有限状态自动机。图3.2就是一个模拟英语名词屈折变化的最简单的有限状态模型。

图3.2中的FSA假定词表中包含规则名词，记为reg-noun，它们都采用规则的*-s*为复数词尾（例如*cat*, *dog*, *fox*和*aardvark*），现在我们暂时忽略*fox*这样的词在复数词尾前要插入*e*的语言事实，这样，名词的规则复数就占了绝大多数。这个词表中也包括不采用*-s*的非规则名词形式，并且分两种情况：一种是非规则单数名词irreg-sg-noun（*goose*或*mouse*），另一种是非规则复数形式irreg-pl-noun（*geese*或*mice*）。

reg-noun	irreg-pl-noun	irreg-sg-noun	plural
fox	geese	goose	-s
cat	sheep	sheep	
dog	mice	mouse	
aardvark			

英语动词屈折变化的简单模型如图3.3所示。

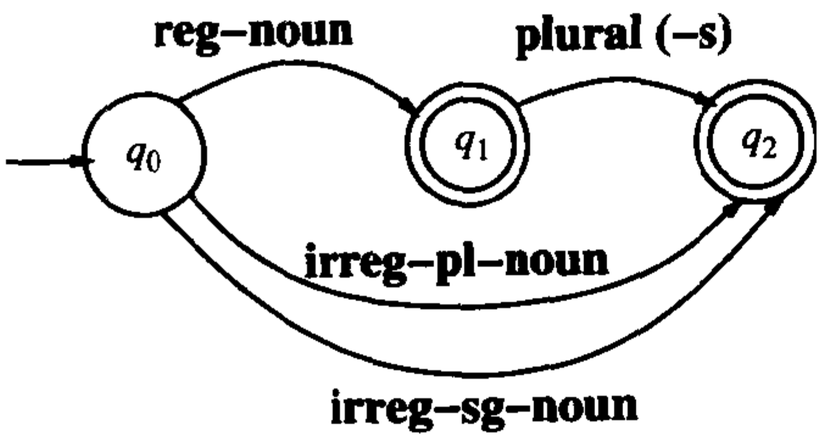


图3.2 模拟英语名词屈折变化的有限状态自动机

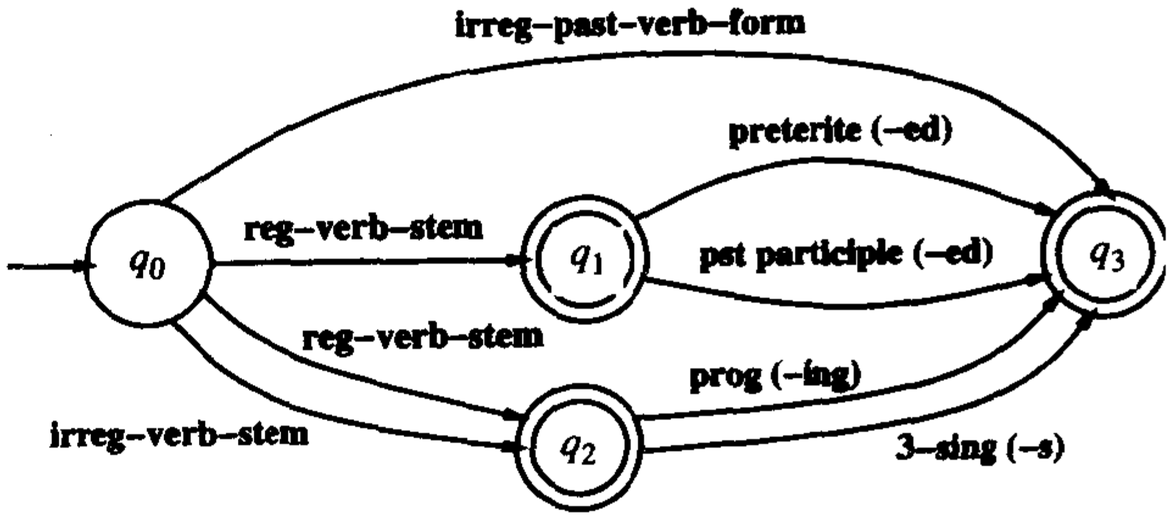


图3.3 模拟英语动词屈折变化的有限状态自动机

这个词表包括3个词干类（reg-verb-stem, irreg-verb-stem和irreg-past-verb-form）以及4个以上的词缀类（*-ed* past, *-ed* participle, *-ing* participle和third singular *-s*）。

reg-verb-stem	irreg-verb-stem	irreg-past-verb	past	past-part	pres-part	3sg
walk	cut	caught	-ed	-ed	-ing	-s
fry	speak	ate				
talk	sing	eaten				
impeach	sang					
	spoken					

英语的派生形态学要比英语的屈折形态学复杂得多，所以模拟英语派生的自动机也相当复杂。事实上，某些英语的派生模型是建立在第9章中将要讲述的更复杂的上下文无关语法基础上的(Sproat, 1993; Orgun, 1995)。

作为形态分析的一个初步例子，我们在这里介绍英语形态顺序规则的一个简单分析实例，该实例取自 Antworth(1990)。Antworth 提出了如下的英语形容词数据：

big, bigger, biggest
cool, cooler, coolest, coolly
red, redder, reddest
clear, clearer, clearest, clearly, unclear, unclearly
happy, happier, happiest, happily
unhappy, unhappier, unhappiest, unhappily
real, unreal, really

首先我们假定，这些英语形容词可以具有一个随选的前缀（*un-*），一个必选的词干（*big* 或 *cool* 等）和一个随选的后缀（*-er*, *-est* 或 *-ly*）。如图 3.4 所示。

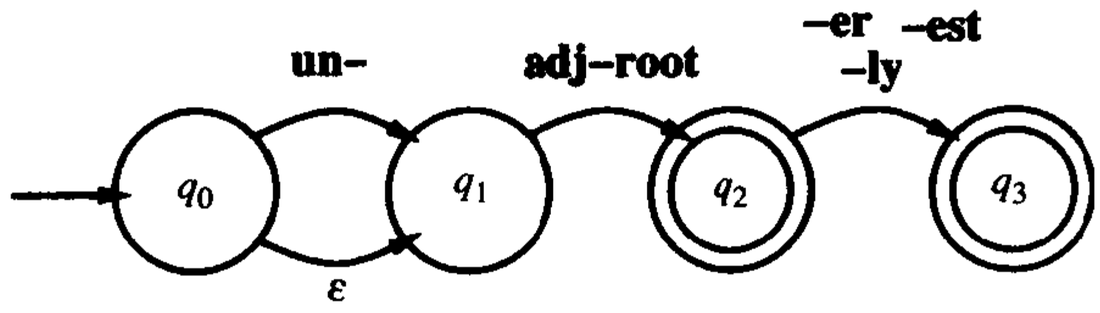


图 3.4　用于分析英语形容词某个片断的 FSA，Antworth 方案 #1

遗憾的是，这个FSA在识别上表中的所有形容词的同时，也可以识别某些不合语法的形式，例如 *unbig*, *redly* 和 *realest*。因此，我们有必要给词干再进行分类，分为 adj-root$_1$ 和 adj-root$_2$，说明什么样的词干可以与什么样的后缀一起出现。因此，adj-root$_1$ 包括能够与 *un-* 和 *-ly* 共现的形容词（*clear*, *happy* 和 *real*），adj-root$_2$ 包括不能与 *un-* 和 *-ly* 共现的形容词（*big*, *cool* 和 *red*）。图 3.5 是 Antworth (1990)对于这个问题的局部解决方案。

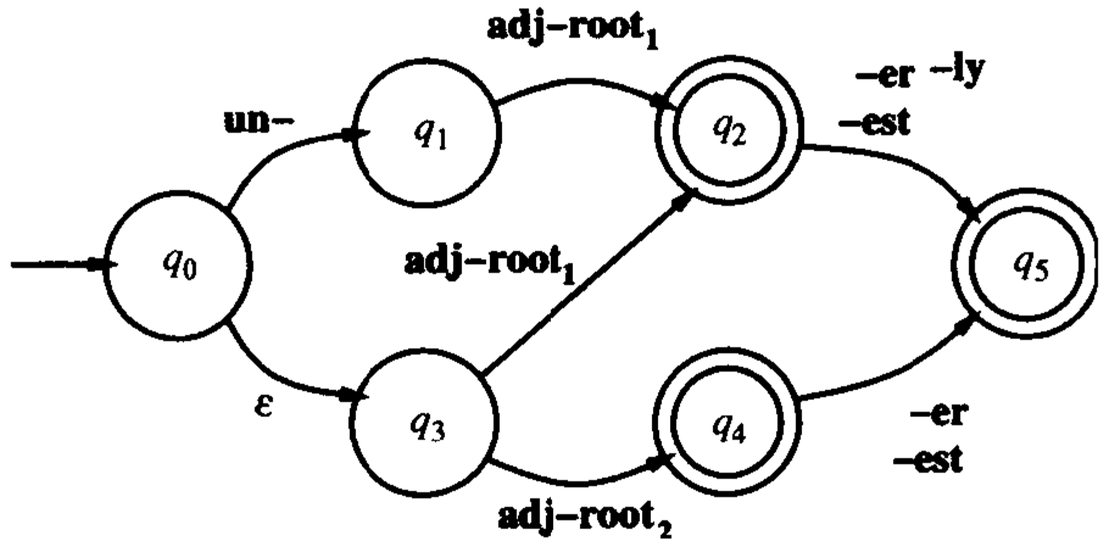


图 3.5　用于分析英语形容词的某个片断的 FSA，Antworth 方案 #2

这些例子使我们看到了英语派生的复杂性。作为进一步的例子，我们在图3.6中给出了描述英语名词和动词派生形态学的另一个FSA片断，该FSA根据的是Sproat(1993)，Bauer(1983)和Porter (1980)。这个 FSA 模拟了一些派生事实，诸如众所周知的以 *-ize* 结尾的动词后面可以接名词后缀 *-ation*（Bauer, 1983; Sproat, 1993）。例如，对于单词 *fossilize*，在图中顺次通过状态 q_0，q_1 和 q_2，我们可以预测单词 *fossilization*。类似地，在状态 q_5，以 *-al* 或 *-able* 结尾的形容词（*equal*, *formal*, *realizable*）可以取后缀 *-ity*，有时可以取后缀 *-ness*（*naturalness*, *casualness*）之后进入状态 q_6。对于上述条件限制，读者会发现一些例外，读者也可以对于上面的名词和动词再提出一些其他类别的例子，这些都可以作为练习留给读者自己去做。

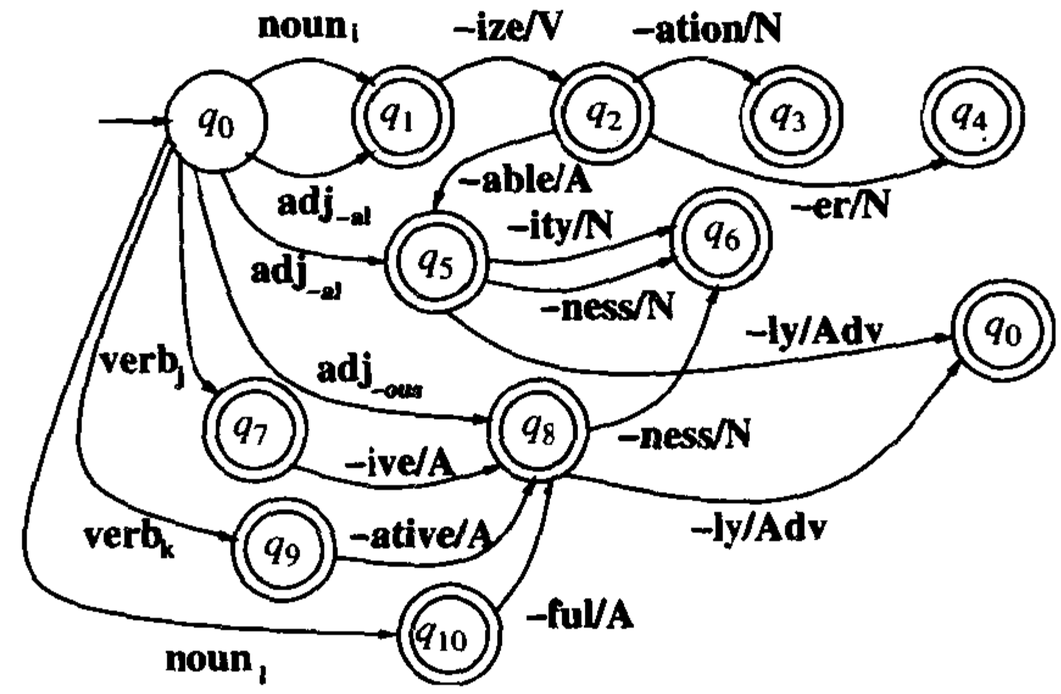


图 3.6 用于英语派生形态学的另一个片断的 FSA

现在我们使用FSA来解决形态识别（morphological recognition）的问题。所谓“形态识别”，就是判断由字母构成的输入符号串是不是合法的。我们使用形态FSA来进行形态识别。在FSA中，对于每个单词再插入一个“子词表”。也就是说，我们在扩充每个弧（例如reg-noun-stem这个弧）时，把reg-noun-stem这个弧看成是由它的字母组成的集合。这样做的结果是FSA可以在单独字母的平面上定义。

图3.7是一个识别名词的FSA。这个FSA是图3.2中用于处理名词屈折的FSA的扩充，它可以处理规则名词和非规则名词的一些实例。我们可以使用图3.7来识别*aardvarks*这个符号串，如在第2章中所看到的，从初始状态开始，把输入的单词的每个字母逐个与输出弧上的字母相比较，就可以进行识别。

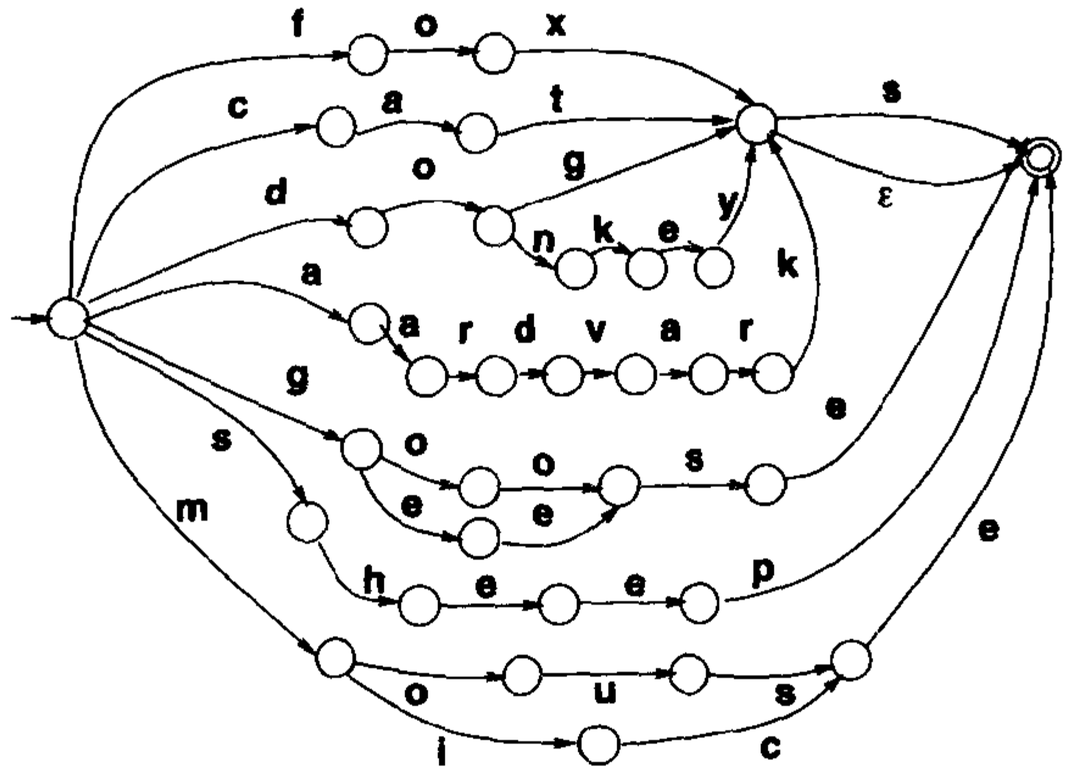


图 3.7 为少量的带屈折变化的英语名词编制的FSA。注意，这个自动机会错误地接受*foxs*这个输入。我们在下面将会看到怎样处理*foxes*中插入的*-e*的问题

3.2.2 用有限状态转录机进行形态剖析

我们已经知道怎样用FSA来表示词表以及怎样进行形态识别，现在来讨论形态剖析问题。例如，给出输入*cats*，经过形态剖析后可以得到输出cat+N+PL，这样我们就知道*cat*是一个复数名词。我们使用由 Koskenniemi(1983)首次提出的双层形态学（two-level morphology）的方法来进行这样的形态剖析。双层形态学把一个词表示为词汇层（lexical level）和表层（surface level）之间的对应，词汇层表示组成该词的语素之间的简单毗连关系，表层表示该词实际拼写的最终情况。形态剖析要建立映射规则，把在表层上的字母序列（如*cats*）映射为词汇层上的语素和特征的序列（如

cat+N+PL)。图 3.8 说明了单词 *cats* 的这两个层。注意，在词汇层上是该词的词干后面跟着该词的形态信息 +N+PL，告诉我们 *cats* 是一个复数名词。

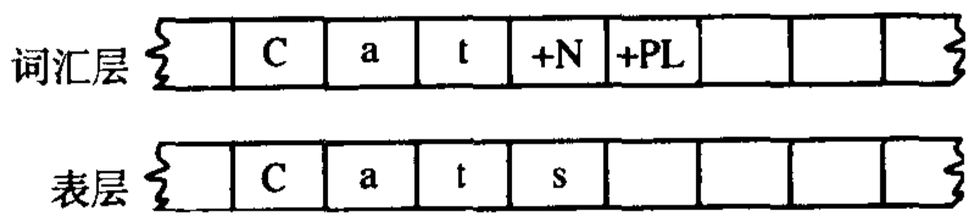


图 3.8　词汇层带子和表层带子示例

这种用来进行两个层之间的映射的自动机称为有限状态转录机(finite-state transducer或FST)。转录机实现两个符号集合之间的映射；有限状态转录机通过有限自动机来实现这种转录。因此，我们通常把 FST 看成是具有两个带子的自动机，它可以识别或者生成符号串 *paris* 的偶对。FST 具有比 FSA 更多的功能；FSA 通过确定符号集合来定义形式语言，而 FST 则定义符号串 *relation* 之间的关系。这样就可以从另一个角度把 FST 看成是读一个符号串并生成另一个符号串的机器。总之，我们可以从 4 个途径来看待转录机：

- **作为识别器 (recognizer) 的 FST**：转录机取符号串的偶对作为它的输入和输出，如果该符号串偶对也在语言的符号串偶对中就接收，否则就拒绝。
- **作为生成器 (generator) 的 FST**：转录机输出语言的符号串偶对，因此这时输出的是 yes 或 no 以及输出符号串的偶对。
- **作为翻译器 (translator) 的 FST**：转录机读一个符号串，并输出另一个符号串。
- **作为关联器 (relater) 的 FST**：转录机计算两个集合之间的关系。

FST 可以通过不同的方式形式地加以定义。我们采用如下的定义，这个定义的依据是把 Mealy 机（Mealy machine）扩充为简单的 FSA：

- Q：状态 $q_0, q_1, \cdots, q_N$ 的有限集合。
- Σ：复杂符号的有限字母表。每个复杂符号由输入–输出偶对 $i:o$ 构成；i 是输入字母表中的一个符号 I，o 是输出字母表中的一个符号 O，因此，$\Sigma \subseteq I \times O$。$I$ 和 O 都包含空符号 ε。
- q_0：初始符号。
- F：最后状态的集合，$F \subseteq Q$。
- $\delta(q, i:o)$：转换函数或状态之间的转换矩阵。给定一个状态 $q \in Q$ 和复杂符号 $i:o \in \Sigma$，$\delta(q, i:o)$返回一个新状态 $q' \in Q$。因此，δ 是从 $Q \times \Sigma$ 到 Q 的一个关系。

FSA 接收在单个符号的有限字母表（如羊的语言的字母表）上表述的语言，这时字母表为：

$$\Sigma = \{b, a, !\} \tag{3.2}$$

但是，FST 则接收在符号偶对上表述的语言，这时字母表为：

$$\Sigma = \{a:a, b:b, !:!, a:!, a:\varepsilon, \varepsilon:!\} \tag{3.3}$$

在双层形态学中，Σ 中的符号偶对又称为可行偶对（feasible pairs）。

FSA 与正则语言同构 (isomorphic)，FST 与正则关系同构。正则关系是符号串偶对的集合，作为符号串集合的正则语言的自然扩充。正如 FSA 和正则语言，FST 和正则关系对于并运算是封闭的。一般来说，它们对于差运算、补运算和交运算不封闭 (虽然 FST 的某些有用的子集合对于这些运算是封闭的；没有用 ε 提升的 FST 一般更倾向于具有这些闭包性质)。除了并运算之外，FST 具有下面两个附加的非常有用的闭包特性：

- **逆反**（inversion）：转录机的逆反T（T^{-1}）可以简单地在输入标记和输出标记之间切换。因此，如果T从输入字母表I映射到输出字母表O，那么，T^{-1}从O映射到I。
- **组合**（composition）：如果T_1是从I_1映射到O_1的转录机，T_2是从I_2映射到O_2的转录机，那么，$T_1 \circ T_2$可以从I_1映射到O_2而得到。

逆反的用处在于通过它可以方便地实现从作为剖析器的FST到作为生成器的FST转换。组合的用处在于通过它可以采用两个连续作用的转录机，并可以用一个更复杂的转录机来替换它们。组合用代数可以表示为：将$T_1 \circ T_2$应用于一个输入序列S等同于将T_1应用于S，再将T_2应用于它们，即$T_1 \circ T_2 = T_2(T_1(S))$。下面将会看到一些组合的例子。

我们说过，在双层形态学中，把FST看成是具有两个带子的机器，处理起来会很方便。词汇带子在上层，由*a*:*b*偶对的左侧字符构成；表层带子在下层，由*a*:*b*偶对的右侧字符构成。因此，在转录机字母表Σ中的每个符号偶对*a*:*b*表示一个带子中的*a*如何映射到另一个带子中的符号*b*。例如，*a*:ε表示，上层带子中的*a*对应于下层带子中的“无”（*nothing*）。正如对于FSA那样，我们可以写出复杂字母表Σ上的正则表达式。由于符号映射于自身是很常见的，在双层形态学中，我们把像*a*:*a*这样的偶对称为默认偶对（default pairs），并且只用一个单独的字母*a*来引用它。

现在就可以通过为早先提及的形态FSA和词汇增加“词汇”带子和恰当的形态特征来建立一个FST形态剖析器。图3.9是对图3.2的扩充，增加了每个语素所对应的名词形态特征（+SG和+PL）。注意，由于输出带子上没有这些特征对应的片断，它们映射到空字符ε或单词/语素的边界符号#。

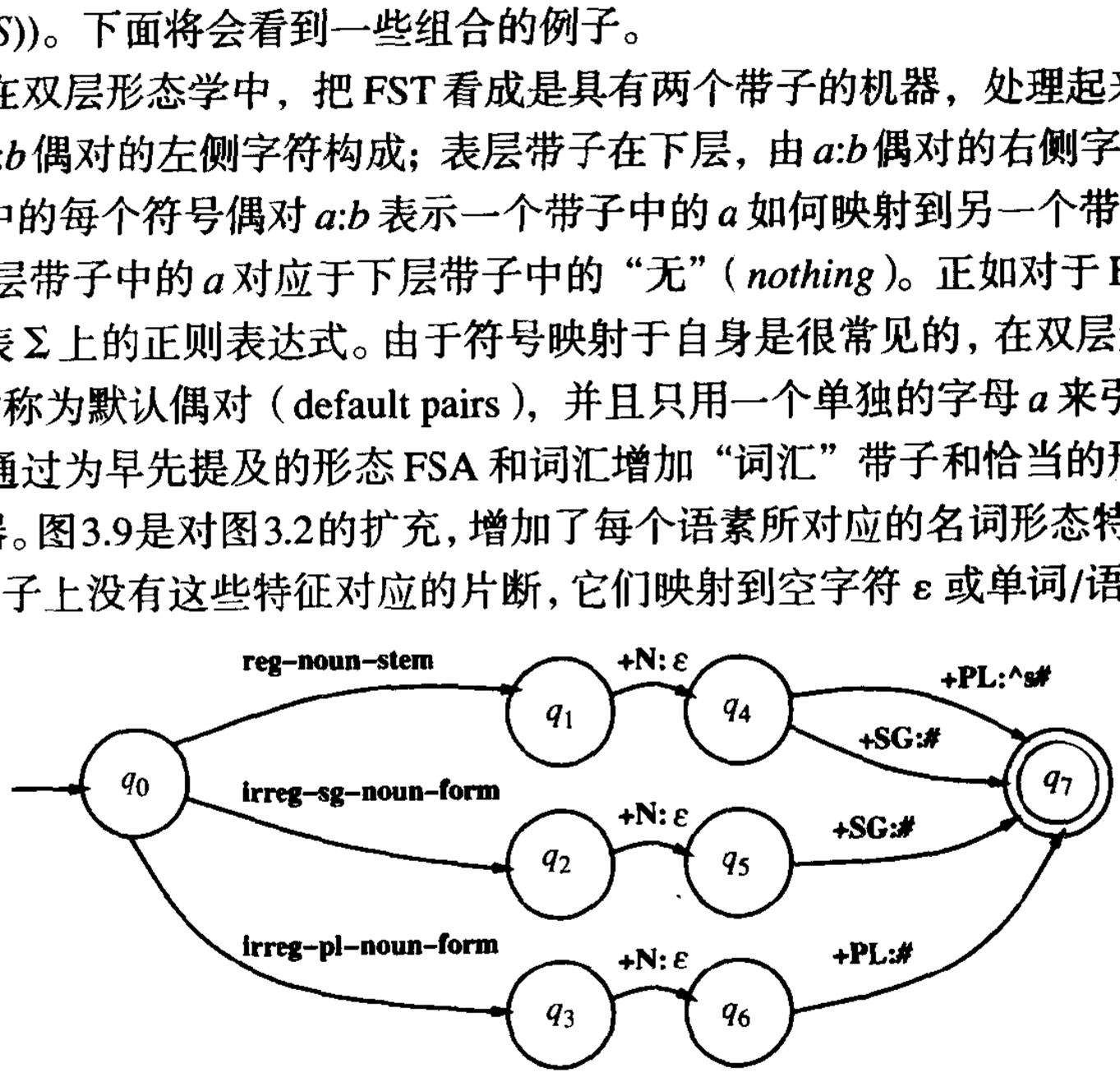


图3.9 表示英语名词复数屈折变化的转录机Tnum。因为q_1和q_2都是接收状态，规则名词可以有复数后缀，也可以没有。语素边界符号^和单词边界标志#将在下面讨论

为了把图3.9用做名词的形态剖析器，有必要用所有的规则和非规则名词的词干来替换regular-noun-stem这个标记。为此，我们需要按转录机的要求来更新词表，使得像*geese*这样的非规则复数名词能够被剖析为正确的词干goose+N+PL。为了这样做，我们要允许词表也具有两个层次。由于表层*geese*映射为*goose*，新的词汇条目可以写为“g:g o:e o:e s:s e:e”。规则形式的写法要简单一些，fox的双层条目现在可以写为“f:f o:o x:x”。但是，根据正词法的规定，f表示f:f，我们可以简单地用fox来引用它，把*geese*写为“g o:e o:e s e”。这样，词表看起来稍微有一点复杂：

reg-noun	irreg-pl-noun	irreg-sg-noun
fox	g o:e o:e s e	goose
cat	sheep	sheep
dog	m o:i u:ε s:c e	mouse
aardvark		

我们提出的形态剖析器需要从像*geese*这样的表层形式映射到像goose+N+SG这样的词汇形式。为此，我们可以把图3.9这样的处理单数/复数的自动机与词表进行层叠式连接（cascading）。对两个自动机进行层叠式连接，就是在运行时把第一个自动机的输出作为第二个自动机的输入。我们首

先把上表中的词干词表用图 3.10 中的有限状态转录机 T_{stems} 来表示。例如，这个 FST 可以把 *dog* 映射为 reg-noun-stem。为了允许分析可能的后缀，图 3.10 中的 T_{stems} 可以接上广义符号 @；@:@ 表示“任何可行的偶对”。例如，:形式 @:x 表示“在表层有 x 的任何可行的偶对”，形式 x:@ 的含义也可以做相应的解释。这个 FST 的输出接着将输入到处理数的自动机 T_{num} 中去。

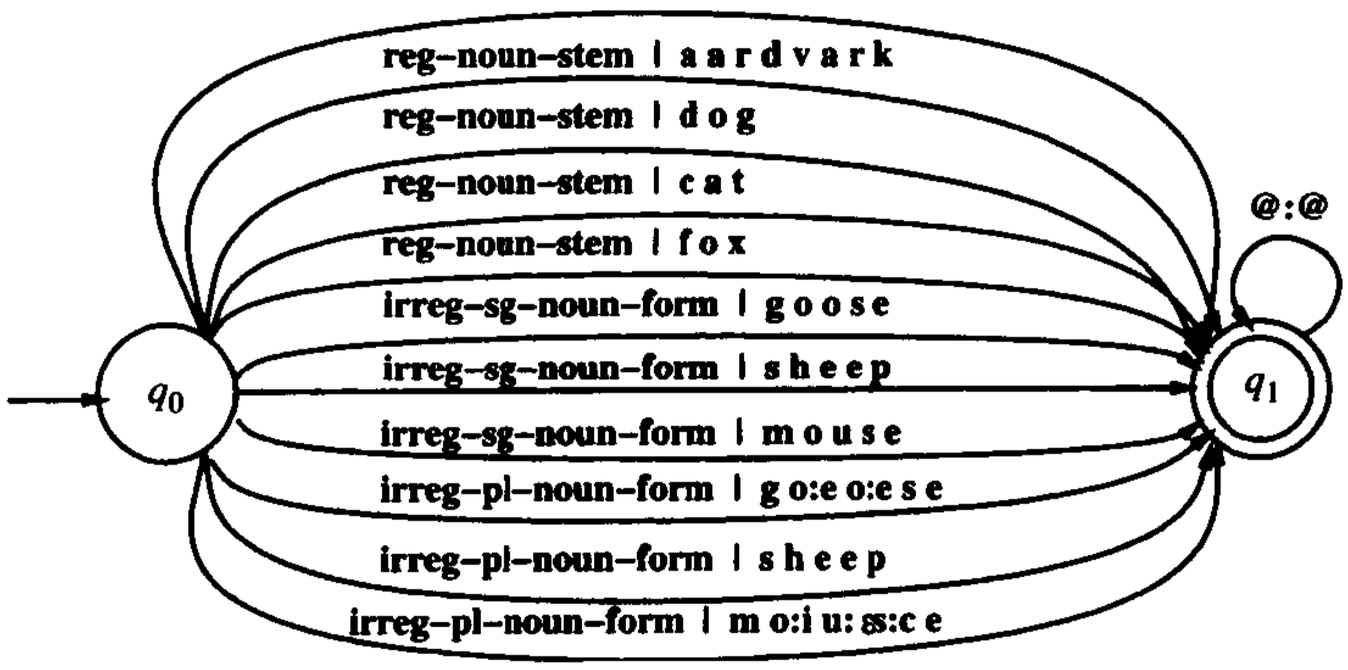


图 3.10　可以把词根映射为词根类的转录机 T_{stems}

我们也可以不用把两个转录机进行层叠式连接，而使用前面定义过的组合算符对这两个转录机进行组合。所谓组合，就是把原来具有若干个输入和输出层的转录机的层叠式连接，转化为只具有一个输入和一个输出的单独“双层”转录机。组合算法类似于前面讲过的 FSA 的确定性算法。给定具有状态集合 Q_1 和 Q_2 以及转换函数 δ_1 和 δ_2 的两个自动机 T_1 和 T_2，我们给每个状态偶对 $x \in Q_1$ 和 $y \in Q_2$ 造出一个新的状态 (x, y)，新的自动机的转换函数如下：

$$\begin{gathered}\delta_3((x_a,y_a),i:o)=(x_b,y_b)\\ \text{如果}\exists c \text{ 使得 } \delta_1(x_a,i:c)=x_b\\ \text{并且}\delta_2(y_a,c:o)=y_b\end{gathered} \tag{3.4}$$

这样组合成的自动机为 $T_{lex} = T_{num} \circ T_{stems}$，参见图 3.11（与图 3.7 中处理词表的 FSA 对比）[①]。注意，最后得到的自动机还有两个层，层与层之间用“:”分开。这些层还保存着冒号，而在图 3.10 的 T_{stems} 中，上层带子和下层带子之间则用“|”号隔开。

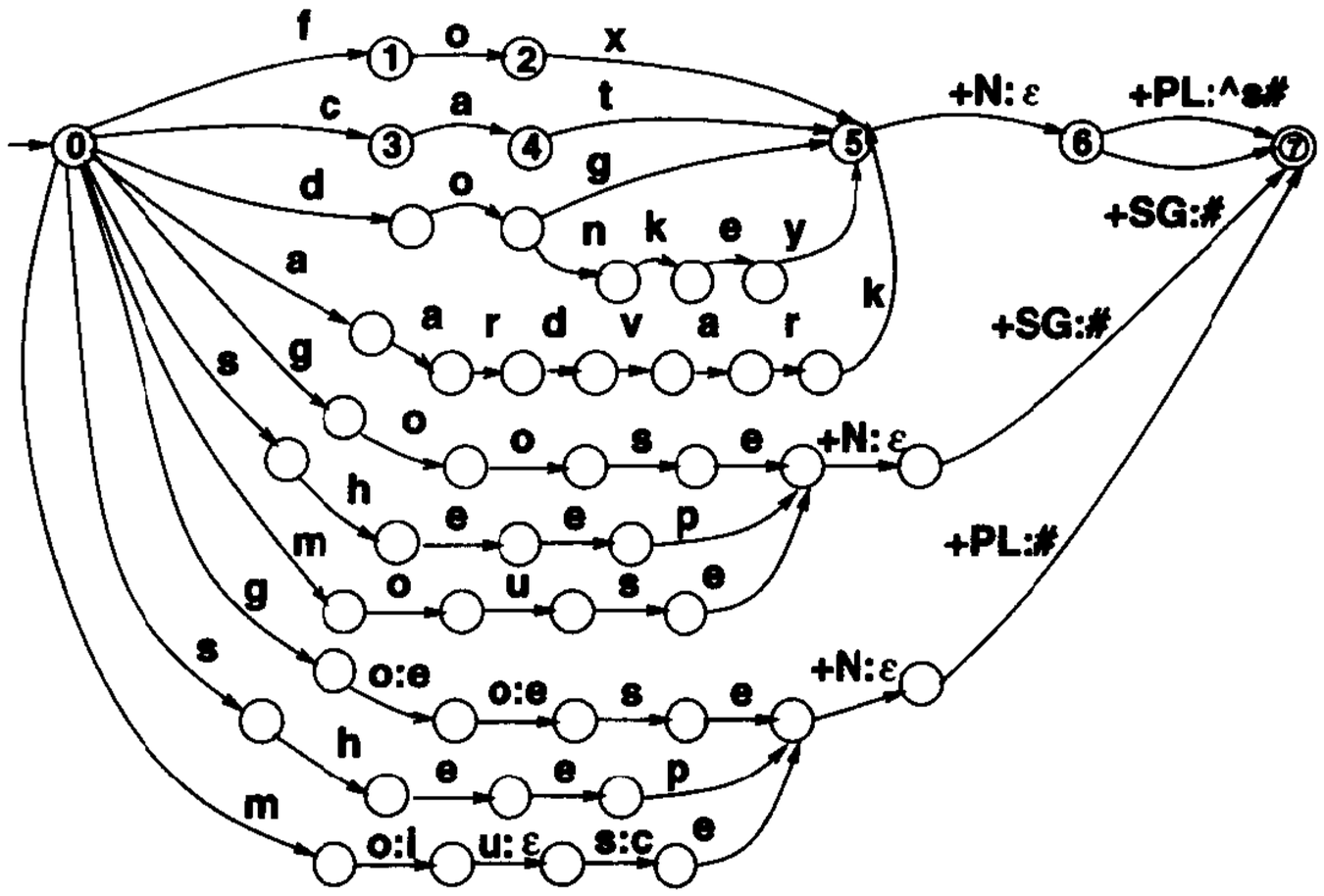


图 3.11　处理英语中名词屈折变化的一个有血有肉的 FST：$T_{lex} = T_{num} \circ T_{stems}$

① 注意，为了表述得更加清晰，我们在图 3.11 中没有采用图 3.7 中的简化表达方式。

这个转录机可以把复数名词映射到词干+形态标志+PL中，把单数名词映射到词干+形态标志+SG中。因此，一个表层形式 *cats* 可映射到 cat+N+PL 中，具体地说就是：

```
c:c  a:a  t:t  +N:ε  +PL:^s#
```

这时，c映射为它自身，a和t也是如此，形态特征 +N（表示名词）映射为空（ε），特征 +PL（表示复数）映射为^s#，符号^表示语素边界（morpheme boundary），符号#表示单词边界（word boundary）。在图3.12中，把语素边界标志称为中间带子（intermediate tape），我们将在下一节中说明如何去掉这些边界标志。

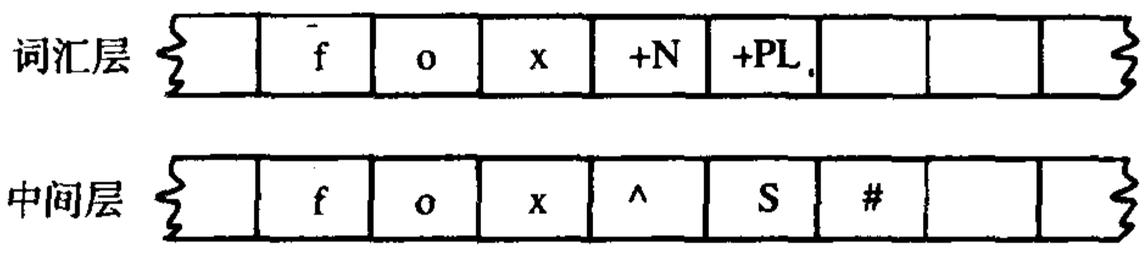


图 3.12　词汇带子和中间带子的一个实例

3.2.3　正词法规则和有限状态转录机

上一节中所描述的方法可以成功地识别 *aardvark* 和 mice 这样的单词。但是，当出现拼写变化时，刚才那种毗连语素的方法将不能工作，它将会错误地拒绝 *foxes* 这样的正确输入，而接收 *foxs* 这样的错误输入。因此，我们有必要引入拼写规则（正词法规则）来处理英语中经常在语素边界发生拼写变化的问题。本节将介绍书写这些规则的一些方法，并说明怎样在转录机上实现这些规则。下面是一些拼写规则：

名称	规则描述	例子
辅音重叠	在*-ing*和*-ed*之前重叠单字母辅音	beg/begging
E的删除	在*-ing*和*-ed*之前删除不发音的*e*	make/making
E的插入	在*-s, -z, -x, -ch*和*-sh*之后，*-s*之前加*e*	watch/watches
Y的替换	*-y*在*-s*之前变为*-ie*，在*-ed*之前变为*-i*	try/tries
K的插入	以元音+*-c*结尾的动词，*-c*后加*-k*	panic/panicked

我们可以把这种拼写变化想像成以语素的简单毗连作为输入，以稍微变化了的（正确拼写的）语素毗连作为输出。我们在这里采用了三层：词汇层、中间层和表层，图3.13是这三层的说明。这样，我们可以写一条E的插入规则，图3.13说明了从中间面到表层映射的情况。

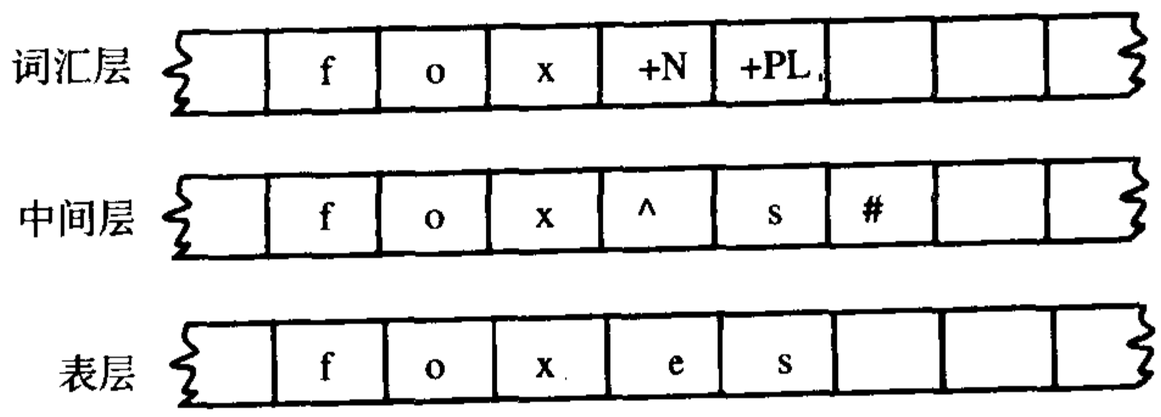


图 3.13　词汇带子、中间带子和表层带子的一个实例。在每两个带子之间是一个双层转录机。图3.11中的词汇转录机处于词汇层和中间层之间，E插入的拼写规则处于中间层和表层之间。当中间层有一个语素边界^后接语素 *-s* 时，E 插入的拼写规则在表层带子中插入一个 *e*

这个规则的意思是："当词汇带子有一个以x（或z等）为结尾的语素而下一个语素是-s时，在表层带子中插入一个e"。这个规则的形式描述如下：

$$\varepsilon \rightarrow e / \left\{ \begin{matrix} x \\ s \\ z \end{matrix} \right\} \hat{}__ s\# \tag{3.5}$$

这是Chomsky and Halle(1968)的规则记法。形式为$a \rightarrow b / c__d$的规则的意思是"当a在c和d之间出现时，把a改写为b"。由于符号ε表示空转换，替换它意味着在空位置插入某种东西。符号^表示语素边界。在转录机取默认偶对时，用符号表示为^:ε，这时，边界就被删除了。可见，语素边界标志在表层是用默认值来删除的。注意，在中间形式和表层形式中，我们是用冒号来把符号分开的。符号#是用于表示单词边界的一个专门符号。所以，式（3.5）的含义是："在以x, s或z结尾的语素之后，语素s之前，插入一个e"。图3.14是相应于这个规则的一个自动机。

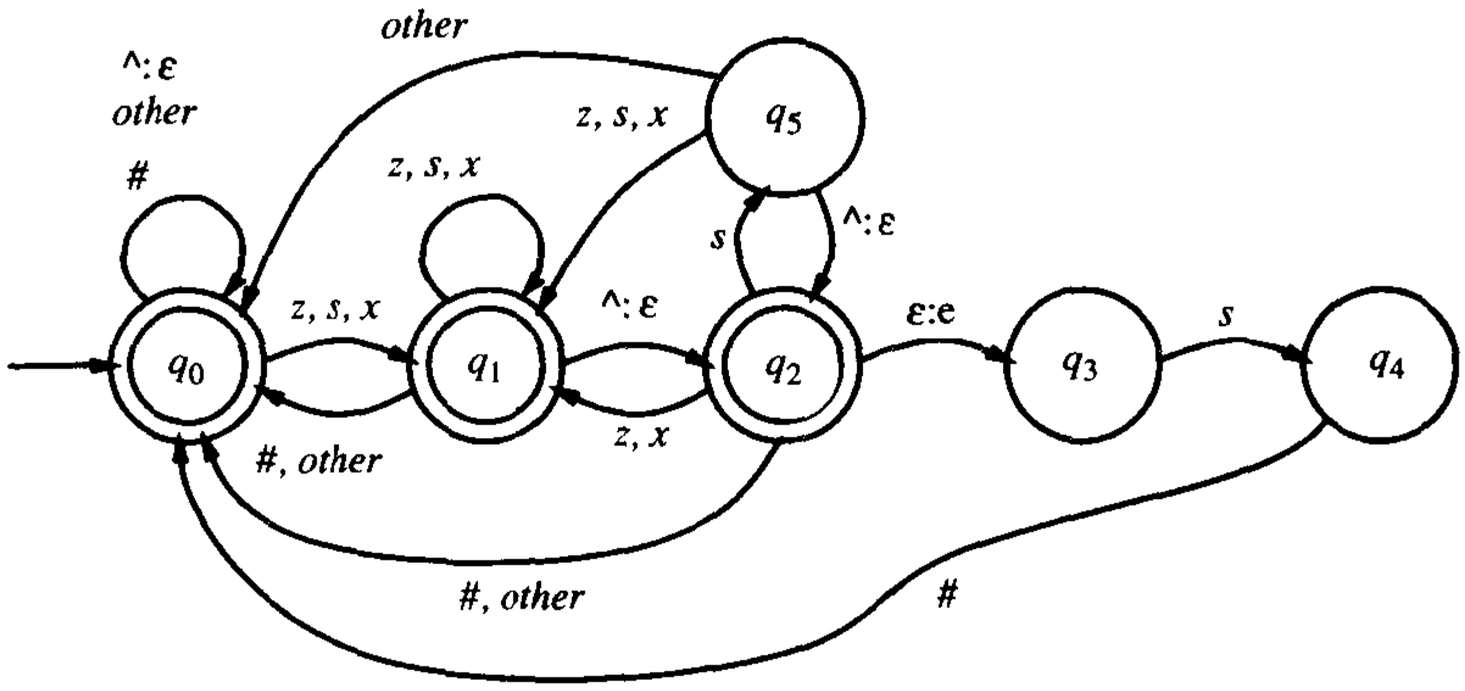


图3.14　表示式（3.5）的E插入规则的转录机，从Antworth(1990)的类似转录机扩充而成

为特定的规则设计转录机的思想只是为了表示对于该规则所需的那些限制，而允许其他的符号串通过转录机时不做任何变化。使用这个规则是为了确保当我们在适合的上下文中时，才去查找$\varepsilon : e$。状态q_0只模拟那些与该规则无关的默认偶对，它是一个接收状态；状态q_1也是一个接收状态，它模拟查找z, s或x时的情况。状态q_2模拟在z, s或x之后查找语素边界的情况，它也是一个接收状态；状态q_3模拟查找E-插入时的情况，它不是一个接收状态，因为只有在后面紧跟着是语素s并且再往后是单词结尾符号#时才允许插入。

在图3.14中使用符号*other*是为了安全地通过那些E-插入规则不起作用的单词中的其他任何部分。符号*other*的含义是："任何不在这个转录机中的可行的偶对"；所以它是@:@的翻版，在转录机与转录机连接时，@:@是依赖于上下文的。例如，当离开状态q_0时，我们通过符号z, s或x的弧进入q_1，而不走*other*的弧或停留在状态q_0。符号*other*的语义依赖于在其他弧上是什么符号，在某些弧上还提到符号#，（根据定义）它不包含在*other*中，例如，从q_2到q_0的弧上就清楚地提到了符号#。

当一个符号串可以应用某个规则而实际上并没有应用时，转录机应该正确地拒绝这个符号串。当一个符号串具有进行E-插入的正确环境但并没有插入时，这样的符号串就可能是坏的符号串。状态q_5就是用于保证当环境适合时总能插入e；只有当转录机在适合的语素边界找到s时才可以进入状态q_5。如果转录机在状态q_5，下面一个符号是#，转录机就要拒绝这个符号串（因为从q_5到#不存在合法的转移）。图3.15是一个转移表，对于不合法转移的规则，标以"-"号。

状态\输入	s:s	x:x	z:z	^:ε	ε:e	#	other
q_0:	1	1	1	0	-	0	0
q_1:	1	1	1	2	-	0	0
q_2:	5	1	1	0	3	0	0
q_3	4	-	-	-	-	-	-
q_4	-	-	-	-	-	0	-
q_5	1	1	1	2	-	-	0

图 3.15　图 3.14 的 E- 插入规则的状态转移表，从 Antworth(1990)的类似转录机扩充而成

下一节中将以一个输入符号串为实例，讲述 E- 插入转录机的追踪过程。

3.3　把 FST 词表与规则相结合

现在我们准备把转录机的词表和规则结合起来进行剖析和生成。图 3.16 说明了一个双层形态学系统的结构，它既能用于剖析也能用于生成。词表转录机把表示词干和形态特征的词汇层映射于表示语素简单毗连的中间层。然后，若干个转录机的主体部分开始运行，每个转录机表示一个单独的拼写规则限制，所有这些转录机并行地运行，在中间层和表层之间进行映射。并行地运行所有的拼写规则需要进行选择，当然也可以按顺序来运行所有的拼写规则（就像一个很长的层叠式连接），这时需要对每个规则稍做改变。

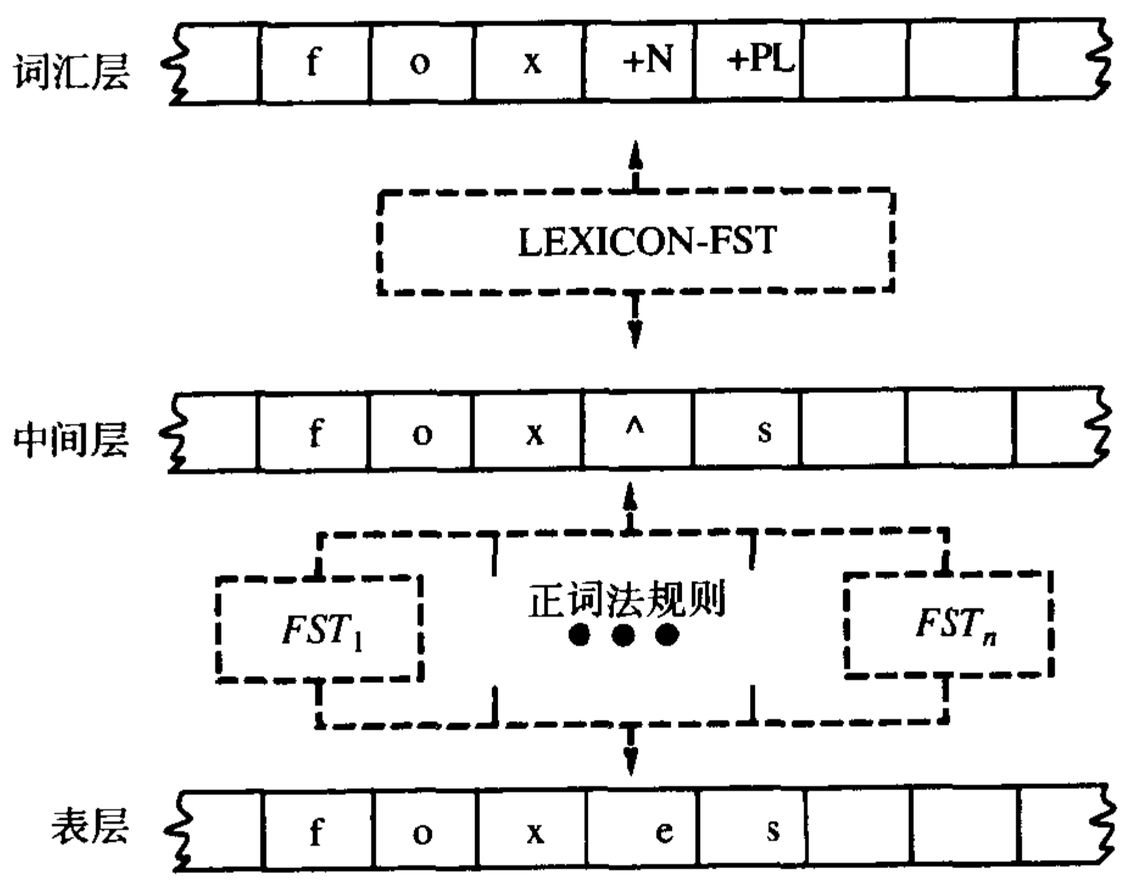


图 3.16　用 FST 的词表和规则来生成和剖析

图 3.16 表示了一个双层的层叠式转录机的结构。前面说过，层叠式转录机是若干个转录机按顺序排列的集合，其中一个转录机的输出作为另一个转录机的输入，层叠的深度可以是任意的，而在每层可以有多个单独的转录机。图 3.16 中的层叠式转录机按顺序有两组转录机。第一组转录机只有一个（LEXICON-FST），从词汇层映射到中间层，第二组转录机是若干个并行转录机的集合，从中间层映射到表层。这个层叠式转录机可以自顶向下地运行，以便生成一个符号串；或者也可以自底向上地运行，以便对符号串进行剖析。图 3.17 显示了系统接收从 *fox^s* 到 *foxes* 映射的追踪过程。

有限状态转录机的强大之处在于，当它从词汇带子生成表层带子时，以及当它从表层带子剖析词汇带子时，都可以使用带有同样状态序列的同样的层叠式转录机。例如，在生成时我们可以想像中间带子和表层带子开始时都是空白的。如果我们运行词表转录机，这时在词汇带子上是

fox+N+PL，它将在中间带子上产生*fox^s#*，而它所通过的状态与前面例子中该转录机接收词汇带子和中间带子时的状态是完全一样的。因此，如果我们把所有表示正词法规则的转录机都并行地运行起来，将可以产生同样的表层带子。

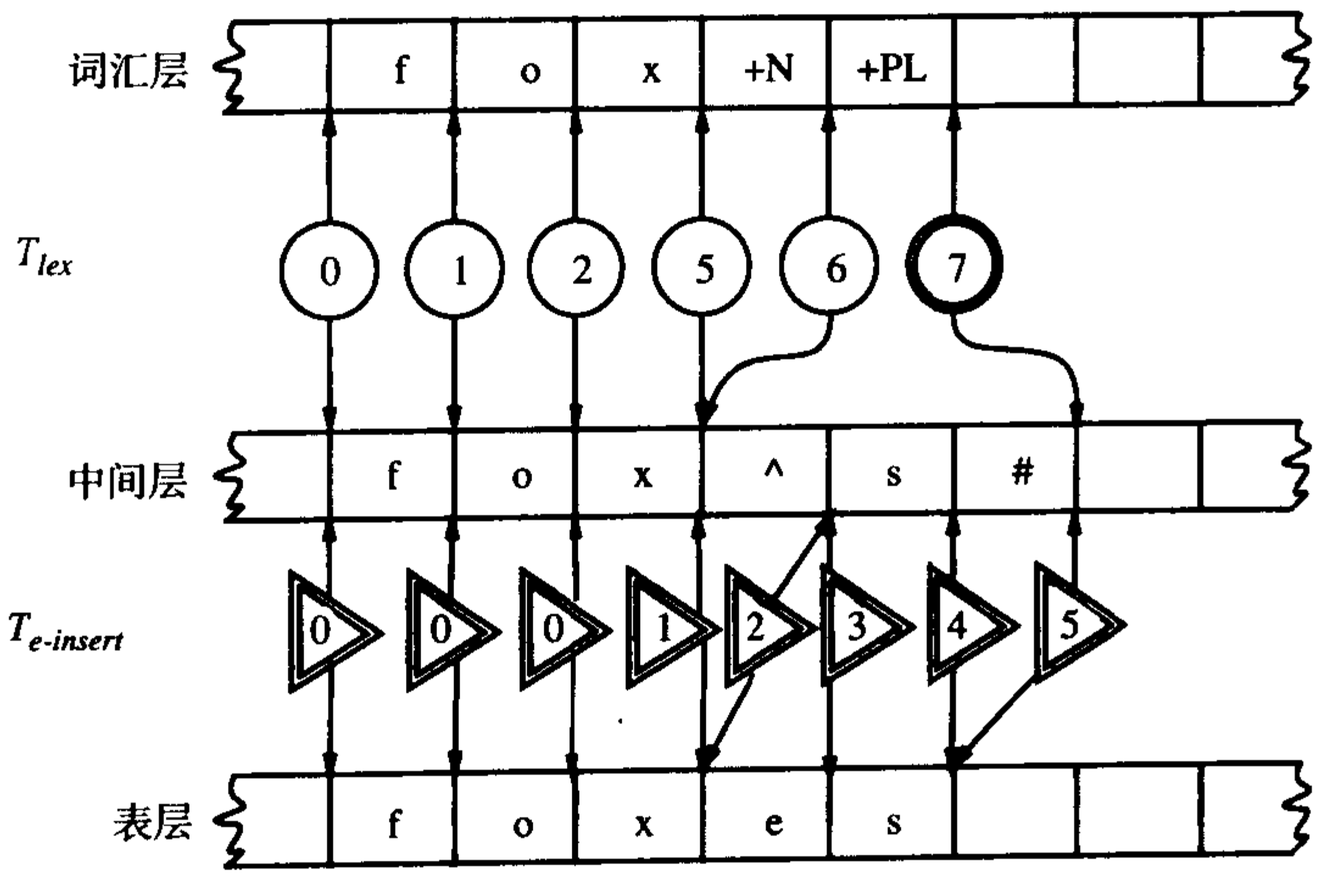


图 3.17　接收*foxes*的过程。图 3.13 中的词表转录机 T_{lex} 与图 3.14 中的 E- 插入转录机进行层叠式连接

剖析比生成稍微复杂一些，因为在剖析中存在歧义（ambiguity）的问题。例如，*foxes*还可以是动词（很少使用，其意思是“弄糊涂，难住”），所以，对*foxes*词汇剖析的结果还可以是 fox +V +3SG 以及 fox +N +PL。我们怎样才能知道哪个才是正确的剖析结果呢？事实上，在出现这样的歧义的场合，转录机也是没有能力判断的。歧义消解（disambiguating）要求某些外部的证据，例如要求知道周围的词的环境等。例如，在词的序列*I saw two foxes yesterday*中，*foxex*是名词，但是在词的序列*That trickster foxes me every time!*（这个骗子每次都把我难住了！）中*foxes*则是动词。我们将在第 8 章和第 11 章中讨论歧义消解的问题。如果没有这样的外部环境，我们的转录机最多只能枚举出可能的选择，因此我们能把*fox^s#*转录为 fox +V +3SG 和 fox +N +PL。

还有一种需要处理的歧义，即发生在剖析过程中的局部歧义。例如，在对输入动词*assess*进行剖析的过程中，当处理了*ass*之后，我们的 E- 插入转录机可能会认为后面跟着的*e*应该用拼写规则来插入（例如，根据我们迄今为止对转录机的了解，可能会把它剖析为单词*asses*）。但是我们在*asses*之后看不到符号 #，而看到另一个 *s*，这时我们才认识到路径走错了。

由于这种非确定性，FST 的剖析算法需要引入一些搜索算法。作为练习，我们请读者修改第 2 章的图 2.21 中的非确定 FSA 的识别算法，使之能进行 FST 的剖析。

运行层叠式FST，特别是当它具有多层时，操作起来可能很不容易。不过，我们已经知道如何把一系列层叠式转录机组合成一个单独的但更复杂的转录机。并行的转录机可以用自动机交合（automaton intersection）的方式进行组合。自动机交合算法就是求状态的笛卡儿乘积，也就是说，对于在自动机 1 中的每个状态 q_i 和在自动机 2 中的每个状态 q_j，我们造一个新的状态 q_{ij}。对于任何输入符号 a，如果自动机 1 把它转移到状态 q_n，自动机 2 把它转移到状态 q_m，那么，我们就把它转移到状态 q_{nm}。

图 3.18 简要地说明了如何进行转录机的交合（∧）与组合（○）。

因为有一系列规则可用于 FST 的编译，实际上几乎用不着亲手写一个 FST 程序。Kaplan and Kay(1994)给出了确定从规则到双层关系的映射的数学原理，Antworth(1990)给出了规则编译算法的细节，Mohri(1997)给出了转录机最小化和确定化的算法。

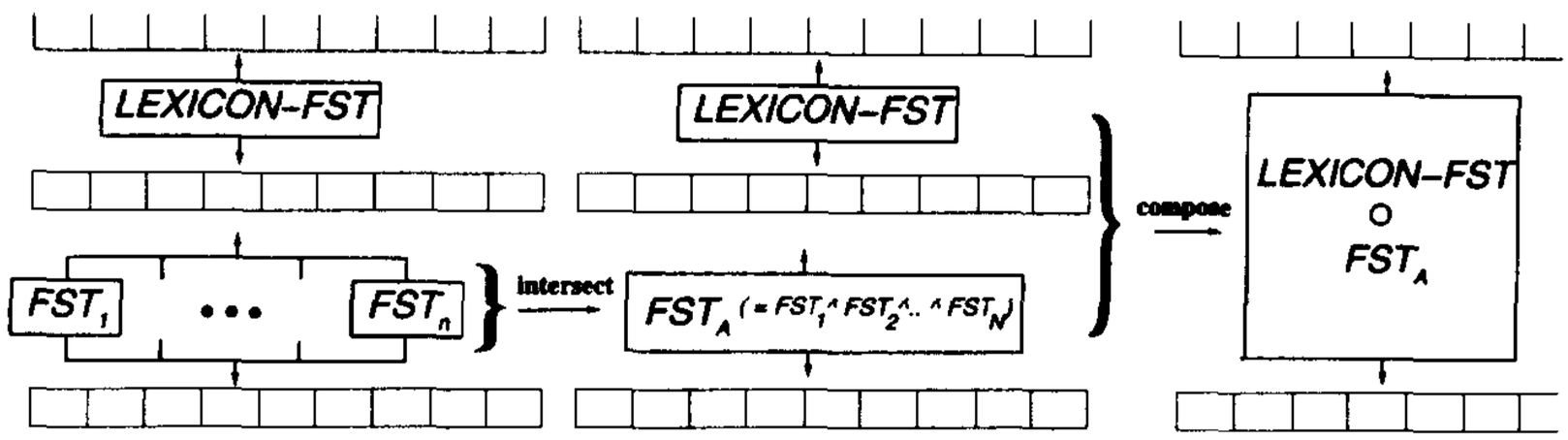


图 3.18　转录机的交合与组合

3.4　与词表无关的 FST：PORTER 词干处理器

建立词表加规则的转录机是形态剖析的标准算法，但与此同时还存在着一些比较简单的算法，并不要求有标准算法所需的大规模联机词表。这样的简单算法可以应用于完成信息检索(Information Retrieval，简称 IR ）这样的任务（见第 17 章）。在信息检索中，用户需要某些信息，因此要查找相关的文献（在互联网上或数字图书馆的数据库中）。用户给系统一个提问，这个提问要注明他所希望查找的文献的某些重要特征，然后 IR 系统检索它认为相关的文献。通常使用的提问把有关的关键词和短语用布尔算符结合起来，例如（*marsupial OR kangaroo OR koala*）。最后，系统返回其中包含这些词的文献作为回答。因为包含单词 *marsupials*（有带动物的复数形式）的文献不一定与关键词 *marsupial*（有带动物的单数形式）相匹配，某些 IR 系统使用词干处理器（stemmer）对关键词以及文献中的词进行处理。由于 IR 中的形态剖析只是用来帮助形成等价的类别的，所以对于后缀的细节并不关心，也不考虑究竟是什么因素可以确定两个词具有同样的词干。

简单而有效的Porter词干处理器(Porter Stemmer, 1980)是使用得最为广泛的词干处理算法之一。这个算法建立在一系列简单的层叠式重写规则的基础上。由于层叠式重写规则正好可以很容易地用 FST 实现，所以我们可以把 Porter 算法想像成一个与词表无关的 FST 词干处理器（这个思路请读者在练习中做进一步推广）。这个算法包含的规则举例如下：

$$\text{ATIONAL} \rightarrow \text{ATE (例如, relational} \rightarrow \text{relate)} \tag{3.6}$$

$$\text{如果词干包含元音，则ING} \rightarrow \varepsilon\text{(例如, motoring} \rightarrow \text{motor)} \tag{3.7}$$

这个算法在附录 B 中有详细介绍。

词干处理器确实改善了信息检索引擎吗？问题是词干处理器还不完善。例如，Krovetz(1993)把 Porter 算法的错误总结为多检错误（errors of commission）和遗漏错误两类（errors of omission）。

多检错误		遗漏错误	
organization	organ	European	Europe
doing	doe	analysis	analyzes
generalization	generic	matrices	matrix
numerical	numerous	noise	noisy
policy	police	sparse	sparsity
university	universe	explain	explanation
negligible	negligent	urgency	urgent

Krovetz还对于Porter词干处理器是否确实改进了IR的效果进行了检测，给出了一些试验结果。他发现，总的情况是有改善的，特别是当文献比较小的时候（文献越大，提问中关键词出现于精确形式的场合也就越多）。由于改善的地方不多，IR 引擎通常并不使用词干处理器。

3.5　人是怎样进行形态处理的

本节介绍心理学的一些研究结果，从而说明多语素词在说英语的人的心中是如何表示的。例如，我们来考虑 *walk* 这个词及其屈折形式 *walks* 和 *walked*。这三个形式都存储在人的心理词表中吗？还是心理词表中只保存*walk*, *-ed*和*-s*？我们可以想像，这样的表示方法在理论上分别处于两个极端。一个极端是完全枚举法（full listing），另一个极端是最小羡余法（minimum redundancy）。完全枚举法假定，在人的心理词表中，语言中的全部单词无论内部形态结构如何，都要一一枚举出来。按照这样的观点，形态结构只是一种没有因果关系的现象（epiphenomenon），因此，在心理词表中，*walk*, *walks*, *walked*, *happy* 和 *happily* 都是分开来枚举的。这种假设对于土耳其语这样的形态复杂的语言显然是行不通的。Hankamer(1989)估计，在土耳其语中有2000亿个不同的词形。最小羡余法假定，在心理词表中，只表示那些有组合能力的语素，因此，当处理 *walks* 的时候（无论是读、听还是交谈），我们都必须分别访问这两个语素（*walk* 和 *-s*），并且把它们组合在一起。

但是，新近的一些实验表明上述两种说法都不完全正确。事实上，某些类的形态关系在心理上是要表示出来的（特别是屈折和某些派生），但是其他的形态关系却不表示出来，而要在心理词表中一一枚举。例如，Stanners et al. (1979)发现，在心理词表中派生形式（*happiness* 或 *happily*）是与它们的词干（*happy*）分开存储的，而很有规则的那些屈折形式（*pouring*）并不与它们的词干（*pour*）分开存储。他们使用重复优先实验（repetition priming experiment）得到了这样的结论。简言之，重复优先实验认为，如果一个词在前面已经出现过，那么它在重复出现时就会具有优先性，所以它就会被更快地识别（这显然是一种优先性）。他们发现，*lifting*优先于*lift*，*burned*优先于*burn*，但*selective* 并不优先于 *select*。图 3.19 是这种发现的一种可能表示的说明。

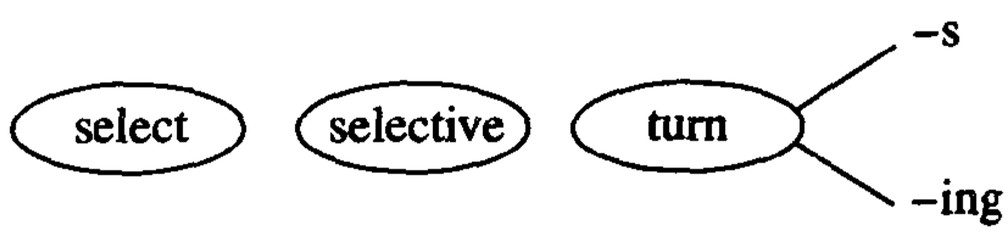


图 3.19　Stanners et al. (1979)的结果：屈折和派生的不同表示

在更新的一些研究中，Marslen-Wilson et al. (1994)发现，口语中的派生词可能优先于它们的词干，但这种优先性只出现在派生形式的意义与词干的意义有紧密联系时。例如，government 优先于 govern，但是 department 并不优先于 depart。Grainger et al. (1991)发现，带前缀的词也有类似的结果（但带后缀的词不然）。Marslen-Wilson et al. (1994)提出了一个与他们所发现的相容的模型，见图 3.20。

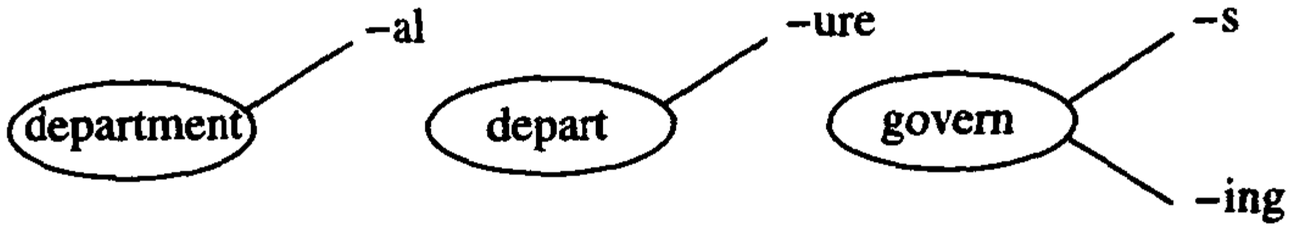


图 3.20　Marslen-Wilson et al. (1994)的结果：仅当语义上相关时，派生词才与它们的词干相联系

关于人的心理词表代表了某种形态结构的其他证据来自口语失误（speech error），又称为言语失误（slips of the tongue）。在正常的对话中，说话者常常颠倒了词的顺序或词首的发音。例如，

if you break it it'll drop
I don't have time to work to watch television because I have to work

第一个句子中的*break*和*drop*的顺序颠倒了，第二个句子中本来想说*watch*却说成了*work*，但*work*应该在后面说。

屈折和派生的词缀在口语中可能和它们的词干分开，例如，在Fromkin and Ratner(1998)和Garrett(1975)中，有如下一些例子：

it's not only us who have screw looses（应为 screws loose）
words of rule formation（应为 rules of word formation）
easy enoughly（应为 easily enough）
which by itself is the most unimplausible sentence you can imagine

这些词缀都具有离开它们的词干而生出来的能力，这说明了心理词表中一定含有表示这些词的形态结构的某种手段。

总之，这些研究结果表明，形态，特别是像屈折变化这样的能产性形态，在人的心理词表中是在起作用的。这些研究结果还特别强调语义在词的构成中的重要性，指出了人的语音词表（用词的音来表示词）和正词法词表（用字母来表示词）可能具有相同的结构。最后，这些结果似乎还说明，如形态这样的语言处理的很多性质，可以同等地（至少是类似地）应用于语言的理解（comprehension）和语言的生成（production）。

3.6 小结

本章介绍自然语言处理中的形态学，主要涉及词的构成、有限状态转录机以及应用于模拟形态规则的一些共同使用的计算工具。下面是本章要点的总结：

- 形态剖析是发现在词中所包含的连续语素的过程（例如，*cats*剖析为cat +N +PL）。
- 英语主要使用前缀和后缀来表示屈折形态和派生形态。
- 英语的屈折形态相对简单，包括人称和数的一致关系以及时态标志（*-ed*和*-ing*）。
- 英语的派生形态比较复杂，包括诸如*-ation*, *-ness*和*-able*这样的后缀以及诸如*co-*和*re-*这样的前缀。
- 英语的形态顺序规则（可允许的语素的顺序）可以用有限自动机来表示。
- 有限状态转录机是能生成输出符号的有限自动机的扩充。
- 双层形态学把有限状态转录机应用于形态表示和剖析。
- 拼写规则可以用转录机来实现。
- 存在着转录机的自动编译程序，该编译程序对于任何简单的重写规则都能造出一个转录机来。
- 词表和规则可以通过组合和交合不同的转录机而结合起来。
- Porter算法是从词干剥离词缀的简单有效的方法。它没有像包含词表的转录机模型那样精确，因此可以应用于诸如信息检索这样的领域，其中不需要精确的形态结构剖析。

3.7 文献和历史说明

尽管有限状态转录机和有限状态自动机在数学上十分相似，但这两个模型却是在不同传统的基础上发展起来的。在第2章中，我们描述了有限自动机是怎样在Turing(1936)的算法计算模型和McCulloch及Pitt的与有限状态模型极为相近的神经元模型的基础上发展起来的。但是，Turing机器对于转录机的影响却不那么直接。Huffman(1954)在Shannon(1938)关于转接电路的代数模型的基

础上，提出了状态转移表来模拟时序电路的行为。在 Turing 和 Shannon 以及鲜为人知的 Huffman 的工作基础上，Moore(1956)为了描述使用输入符号字母表和输出符号字母表并且具备有限个数目的状态的机器，引入了有限自动机（finite automaton）这个术语。Mealy(1955)进一步推广并综合了 Moore 和 Huffman 的研究成果。

Moore 在原来的文章里所描述的自动机与 Mealy 后来推广的自动机有着重要的区别。在 Mealy 自动机中，输入 / 输入符号是通过状态之间的转移来联系的。本章中所讲的有限状态转录机就是 Mealy 自动机。在 Moore 自动机中，输入 / 输出符号是通过状态来联系的，我们在第 5 章和第 7 章中将会看到 Moore 自动机的例子。这两种类型的转录机是等价的，Moore 自动机可以转换成等价的 Mealy 自动机，反之亦然。

很多早期的形态剖析程序使用词缀剥离法(affix-stripping)来进行剖析。例如，在Packard(1973)的古希腊语剖析器中，就反复地剥离输入单词中的前缀和后缀，使剩余的词根突显出来，然后再在词表中查找剩余的词根，并返回与被剥离的词缀相容的词根。这个方法等价于在第 10 章中将要介绍的自底向上（bottom-up）剖析法。

AMPLE（A Morphological Parser for Linguistic Exploration，用于语言研究的形态剖析器）(Weber and Mann, 1981; Weber et al. , 1988; Hankamer and Black, 1991)是另一个早期的自底向上的形态分析器。AMPLE 包含一个词表，在词表中存储每个语素的所有可能的表层变体（称为语素变体，allomorphs），以及它们出现时的限制（例如，英语中的 *-es* 这个表示复数的语素变体只能出现在 *s*, *x*, *z*, *sh* 或 *ch* 之后）。该系统要找出与输入相匹配的每个可能的语素序列，然后进行过滤，把那些不满足限制条件的序列去掉。

形态剖析的另一种方法称为生成检测法（generate-and-test）或综合式分析法（analysis-by-synthesis）。Hankamer(1986)的 keCi 是一个土耳其语的形态剖析器，该剖析器是在土耳其语的语素的有限状态表达式的指导下进行工作的。程序首先匹配单词左边部分的语素，对这些语素使用各种可能的音系学规则，针对输入检测每一个结果。如果一个输出成功了，程序就根据有限状态语素顺序规则继续分析下一个语素，继续对输入进行匹配。

用有限状态转录机来模拟拼写规则的思想来源于 Johnson(1972)早期关于音系规则（将在第 4 章讨论）具有有限状态性质的思想。可惜 Johnson 的这种远见卓识没有引起学术界的注意，后来 Roland Kaplan and Martin Kay 独立地发现了这样的规律，首先是在未发表的谈话中谈到这个规律(Kaplan and Kay, 1981)，后来终于发表了(Kaplan and Kay, 1994)(参见前面关于多重独立发现的论述)。Koskenniemi(1983)继续研究 Kaplan 和 Kay 的发现，并且做了大量的研究工作，描写了芬兰语的有限状态形态规则。Karttunen(1983)根据 Koskenniemi 的模型，建立了一个称为 KIMMO 的程序。Antworth(1990)细致地描述了双层形态学及其在英语中的应用。除了Koskenniemi对芬兰语的研究和Antworth(1990)对英语的研究之外，形态学的双层模型和其他有限状态模型也在很多其他语言的研究中开展起来，例如，Oflazer(1993)对土耳其语的研究，Beesley(1996)对阿拉伯语的研究等。Antworth(1990)联系了各种复杂的形态处理问题，例如，Tagalog 语的中缀化和重叠化，希伯来语的三辅音词根等，总结了语言中有限状态分析的研究情况。Karttunen(1993)很好地总结了双层形态学在音系规则中的应用。关于这个问题，我们将在第 4 章中讨论。Barton et al. (1987)提出了双层模型的几个计算复杂性问题，Koskenniemi and Church(1988)回答了这些问题。

对于自动机的数学基础的细节，有兴趣的读者可以阅读Hopcroft and Ullman(1979)或Lewis and Papadimitriou(1981)。Mohri(1997)和Roche and Schabes(1997b)给出了有关语言应用的更多算法和数学基础。Sproat(1993)是关于计算形态学的导论性读物。

第4章　计算音系学与文本－语音转换

You like po-tay-to and I like po-tah-to.
You like to-may-to and I like to-mah-to.
Po-tay-to, po-tah-to,
To-may-to, to-mah-to,
Let's call the whole thing off!

George and Ira Gershwin, *Let's Call the Whole Thing Off from Shall We Dance*, 1937①

在纯粹的现代教育的辩论中，关于“完全语言教学法”和“读音教学法”这两种教儿童阅读的方法的辩论，似乎很引人注目。正如当代的很多争辩一样，这样的争辩强烈地表现出人类书写系统发展过程中的一个重要的历史辩证法。早期独立地发明的书写系统（苏末文字、汉字、玛雅文字）主要都是表意文字（logographic），一个符号表示一个完整的单词。但是，在我们所能发现的早期文字系统中，大多数系统已经含有音节或音素文字系统的成分。其中的一些符号是用来表示构成单词的语音的。例如，苏末文字中发音为 *ba* 的符号，其意思是“救济粮”，同时它也有纯粹表示 /ba/ 这个音的功能。尽管现代汉字基本上还是表音文字，但也采用声符来拼写外来词，特别是用来拼写外来地名。纯粹的表音文字系统，无论是音节文字（例如，日文的平假名和片假名）还是字母文字（例如，本书中所用的罗马字母），或者辅音文字（例如，闪美特文字系统），一般来说都可以追溯到早期的表意－音节系统。表意和表音这两种文化最后都走在了一起。这样，阿拉伯语、阿拉姆语、希伯来语、希腊语、罗曼语的文字系统，全都来自西闪美特文字，这种西闪美特文字据说是由西闪美特的雇佣兵对古埃及的圣书字的手写体加以修改而形成的。日本语的音节文字是对中国的某些汉字的草书加以修改而形成的，一直用来表音。在中国，唐朝时曾经使用汉字来转写当时传入中国的佛经中的梵文（Sanskrit）音译词。

从理论上讲，无论文字系统的来源如何，上述基于语音的文字系统表明，口语词是由言语的最小单位组合而成的，这是作为我们所有的现代音系学理论的最基础的原始理论（Ur-theory）。在下面的4章中我们将开始探讨这种思想，将介绍现代语音识别和语音合成技术必需的基本成果和有关算法。关于这方面的研究的语言学分支称为计算音系学（computational phonology）。

现在我们来说明这个领域的有关内容。自动语音识别的核心任务是以语音的声学波形作为输入，产生单词串作为输出。反之，文本－语音转换（简称文语转换）的语音合成系统的核心任务是以文本中词的序列作为输入，产生声学波形作为输出。语音识别与语音合成的用途是多方面的，包括自动听写和转写、基于语音的计算机接口、电话、供残疾人用的基于语音的输入和输出以及其他方面的很多应用。我们将在第7章中详细讨论这些问题。

---

① 本章的这段开场白是英文的绕口令，翻译成中文反而看不懂了，因此我们把英文原文写出来，让读者自己体会它的含义。——译者注

本章将重点研究语音识别和文语转换系统中的一个重要问题：词是如何通过称为音子（phone）的单个言语单位发出声音来的。语音识别系统需要研究它所要识别的每个单词的发音，文语转换系统需要研究该系统所说出的每个单词的发音。4.1 节描写发音的语音字母表就是语音学（phonetic alphabet）的一部分。然后，我们介绍发音语音学（articulatory phonetics），研究言语的语音怎样由口腔中的发音器官产生出来。

如果给定的音子在任何上下文中的发音都是一样的，那么发音的模拟就会非常简单。可是，实际情况并非如此。我们将会看到，在不同的语音环境中[t]这个音子的发音各不相同。音系学（phonology）是语言学的一个分支，它要系统地描述音子在不同的环境中的不同实现情况，并且研究语音系统是怎样与语法的其他部分相联系的。下一节将讲述如何写出音位规则（phonological rule）来描述这些不同的实现情况。

然后，我们介绍计算音系学。计算音系学的一个重要部分是研究模拟这些音位规则的计算机制。我们将说明，如何利用第 3 章中的拼写规则转录机来给音系学建模，然后讨论音位学习（phonological learning）的计算机模型，研究如何使用机器学习算法自动推出音位规则。

最后，我们将把基于转录机的音系学模型应用于文语转换中的一个重要问题：从字母串映射到音子串。我们首先讨论一下建立大型发音词典的情况，然后说明如何使用发音的知识来增强第3章中的基于转录机的词表和拼写规则，从而把符合正词法的文字映射到发音。

本章的重点是讲述计算语言学与发音模型中的非概率部分。第5章将讲述概率模型的作用，包括诸如发音变异的概率模型、音位规则学习的概率方法等。

4.1　言语语音与语音标音法

研究单词的发音是语音学的一个部分，语音学是研究用于世界语言中的语音的科学。我们把词的发音模拟为表示音子（phone）和语段（segment）的符号串。音子就是言语的发音，我们将用语音符号来表示音子，这种语音符号与英语这样的字母表语言中使用的字母很相似。例如，用 *l* 来代表的音子一般对应于字母 *l*，用 *p* 来代表的音子一般对应于字母 *p*。事实上，我们以后将会看到，音子具有比字母更多的变异。本章还将简短地讨论一下语音学的其他方面，例如研究音高（pitch）和音长（duration）的韵律（prosody）等。

本节讨论英语（特别是美国英语）中的各个音子，说明它们是如何发出的，是如何用符号来表示的。我们将使用两种不同的字母表来描述音子。第一种是国际音标（International Phonetic Alphabet，简称IPA）。IPA 是一个逐步发展起来的标准，最早由国际语音学会于 1888 年研制出来，作为转写人类所有语言的语音的标准。IPA 不仅仅只是一个字母表，它还有一套标音的原则，随着不同标音的需要而不同。因此，同样一段话，根据 IPA 的原则可能会用不同的方式来标音。在本书中，为简明起见，我们只介绍那些与英语有密切关系的符号。图 4.1 是 IPA 的辅音标音符号的子集，图4.2是IPA的元音标音符号的子集①。这些表中也给出了ARPAbet符号。ARPAbet符号(shoup, 1980)是另一种语音字母表。它是为了给美国英语标音而特别设计的，这种字母表使用 ASCII 字符，我们可以把它看成是 IPA 的美国英语子集的一种方便的 ASCII 表示法。但是，在使用非 ASCII 字母的场合，例如在联机的发音词典中，使用 ARPAbet 常常是很不方便的。

① 为简单起见，我们使用符号[r]来表示美国英语中的“r”这个音子，而不用更标准的 IPA 符号[ɹ]。

IPA和ARPAbet中的很多符号与英语和很多其他语言的正词法中使用的罗马字母是等价的。例如，IPA和ARPAbet中的符号[p]表示处于*platypus*（鸭嘴兽），*puma*（美洲狮）和*pachyderm*（厚皮动物）等单词的开头，*leopard*（美洲豹）的中间，*antelope*（羚羊）结尾的辅音。注意，在单词*antelope*中的最后一个字母*e*只是书面上的，它实际上并不是这个单词的结尾元音，所以这个单词中*p*是最后一个音。

IPA 符号	ARPAbet 符号	单词	IPA 标音	ARPAbet 标音
[p]	[p]	parsley	['parsli]	[p aa r s l iy]
[t]	[t]	tarragon	['tærəgɑn]	[t ae r ax g aa n]
[k]	[k]	catnip	['kætnɨp]	[k ae t n ix p]
[b]	[b]	bay	[beɪ]	[b ey]
[d]	[d]	dill	[dɪl]	[d ih l]
[g]	[g]	garlic	['gɑrlɨk]	[g aa r l ix k]
[m]	[m]	mint	[mɪnt]	[m ih n t]
[n]	[n]	nutmeg	['nʌtmɛg]	[n ah t m eh g]
[ŋ]	[ng]	ginseng	['dʒɪnsɨŋ]	[jh ih n s ix ng]
[f]	[f]	fennel	['fɛnl̩]	[f eh n el]
[v]	[v]	clove	[klouv]	[k l ow v]
[θ]	[th]	thistle	['θɪsl̩]	[th ih s el]
[ð]	[dh]	heather	['hɛðɚ]	[h eh dh axr]
[s]	[s]	sage	[seɪdʒ]	[s ey jh]
[z]	[z]	hazelnut	['heɪzl̩nʌt]	[h ey z el n ah t]
[ʃ]	[sh]	squash	[skwɑʃ]	[s k w a sh]
[ʒ]	[zh]	ambrosia	[æm'brouʒə]	[ae m b r ow zh ax]
[tʃ]	[ch]	chicory	['tʃɪkɚi]	[ch ih k axr iy]
[dʒ]	[jh]	sage	[seɪdʒ]	[s ey jh]
[l]	[l]	licorice	['lɪkɚɨʃ]	[l ih k axr ix sh]
[w]	[w]	kiwi	['kiwi]	[k iy w iy]
[r]	[r]	parsley	['pɑrsli]	[p aa r s l iy]
[j]	[y]	yew	[yu]	[y uw]
[h]	[h]	horseradish	['hɔrsrædɪʃ]	[h ao r s r ae d ih sh]
[ʔ]	[q]	uh-oh	[ʔʌʔou]	[q ah q ow]
[ɾ]	[dx]	butter	['bʌɾɚ]	[b ah dx axr]
[ɾ̃]	[nx]	wintergreen	[wɪɾ̃ɚgrin]	[w ih nx axr g r i n]
[l̩]	[el]	thistle	['θɪsl̩]	[th ih s el]

图 4.1 给英语辅音标音的 IPA 和 ARPAbet 符号

可是，英语正词法和IPA系统中的字母之间的映射关系并不是如此简单。这是因为英语正词法和发音之间的映射关系是很模糊的：一个字母在不同的上下文中可以对应于很不相同的读音。图4.3说明了英语*c*在单词*cougar*（美洲狮）中读为IPA[k]，可是在单词*civet*（麝猫）中却读为IPA[s]。在IPA中，*k*这个读音除了可写为*c*和*k*之外，还可以写为*x*[在*fox*（狐狸）中]，*ck*[在*jackal*（豺狼）中]或*cc*[在*raccoon*（浣熊）中]。在其他一些语言中，例如西班牙语，正词法和读音之间的映射关系比英语要清晰得多。

4.1.1 发音器官

现在我们来讨论发音语音学（articulatory phonetics）。发音语音学研究口腔、咽喉和鼻腔等不同的器官如何改变从肺中来的气流而产生语音。

IPA 符号	ARPAbet 符号	单词	IPA 标音	ARPAbet 标音
[i]	[iy]	lily	['lɪli]	[l ih l iy]
[ɪ]	[ih]	lily	['lɪli]	[l ih l iy]
[eɪ]	[ey]	daisy	['deɪzi]	[d ey z i]
[ɛ]	[eh]	poinsettia	[pɔɪn'sɛɾiə]	[p oy n s eh dx iy ax]
[æ]	[ae]	aster	['æstɚ]	[ae s t axr]
[ɑ]	[aa]	poppy	['papi]	[p aa p i]
[ɔ]	[ao]	orchid	['ɔrkɨd]	[ao r k ix d]
[ʊ]	[uh]	woodruff	['wʊdrʌf]	[w uh d r ah f]
[oʊ]	[ow]	lotus	['loʊɾəs]	[l ow dx ax s]
[u]	[uw]	tulip	['tulip]	[t uw l ix p]
[ʌ]	[uh]	buttercup	['bʌɾɚˌkʌp]	[b uh dx axr k uh p]
[ɝ]	[er]	bird	['bɝd]	[b er d]
[aɪ]	[ay]	iris	['aɪɹɪs]	[ay r ix s]
[aʊ]	[aw]	sunflower	['sʌnflaʊɚ]	[s ah n f l aw axr]
[ɔɪ]	[oy]	poinsettia	[pɔɪn'sɛɾiə]	[p oy n s eh dx iy ax]
[ju]	[y uw]	feverfew	[fivɚfju]	[f iy v axr f y u]
[ə]	[ax]	woodruff	['wʊdrəf]	[w uh d r ax f]
[ɨ]	[ix]	tulip	['tulip]	[t uw l ix p]
[ɚ]	[axr]	heather	['hɛðɚ]	[h eh dh axr]
[ʉ]	[ux]	dude[2]	[dʉd]	[d ux d]

图4.2　给英语元音标音的IPA和ARPAbet符号①

Word	jackal	raccoon	cougar	civet
IPA	['dʒæ.kl̩]	[ræ.'kun]	['ku.gɚ]	['sɪ.vɨt]
ARPAbet	[jh ae k el]	[r ae k uw n]	[k uw g axr]	[s ih v ix t]

图4.3　在英语中，IPA的符号与正词法中的字母之间的映射关系非常复杂，IPA中的[k]和英语正词法中的[c]有很多不同的实现方式

声音是由于空气的快速运动而产生的。人类口头语言中的大多数声音是由肺中排出的空气通过气管（技术上称为trachea）然后从口或鼻中流出而产生的。当通过气管后，空气要通过喉头（larynx，Adam's apple或voicebox）。喉头上有两块小的肌肉，称为声带（vocal folds，非技术名称是vocal cord），声带可以运动到一起，也可以分开。这两个声带之间的空间称为声门（glottis）。如果声带合在一起（但不是合得很紧），当空气通过声带时就会振动；当声带分开时就会停止振动。声带合在一起并发生振动时产生的语音称为浊音（voiced）；当声带不振动时产生的语音称为清音（unvoiced或voiceless）。浊音包括[b]，[d]，[g]，[v]和[z]以及英语中的所有元音，等等。清音包括[p]，[t]，[k]，[f]和[s]等。

如图4.4所示，气管以上的部分称为声腔（vocal tract），声腔包括口腔（oral tract）和鼻腔（nasal tract）两部分。当空气离开气管之后，它能够通过口腔或鼻腔存在于人的身体中。大多数语音是空气通过口腔而产生的。通过鼻腔产生的语音称为鼻音（nasal sound）；发鼻音时要同时使用口腔和鼻腔作为共鸣腔（resonating cavity）。英语的鼻音有*m*，*n*和*ng*。

① 最后一个音子[ʉ]/[ux]在通用的美国英语中是很少见的，因此在原本ARPAbet中没有出现，而是作为“扩充”出现的。Labov(1994)注意到，20世纪70年代后期，（至少）在美国英语的西部和北部的城市方言中，把[uw]往前读为[ux]是很常见的。这种前读的现象首先是通过Moon Zappa模仿和录制“山谷女郎”（Valley Girl）的谈话而流行开来的（Zappa and Zappa, 1982）。不过，大多数人仍然把*dude*这样的单词中的音读为[uw]，而不读为[ux]。

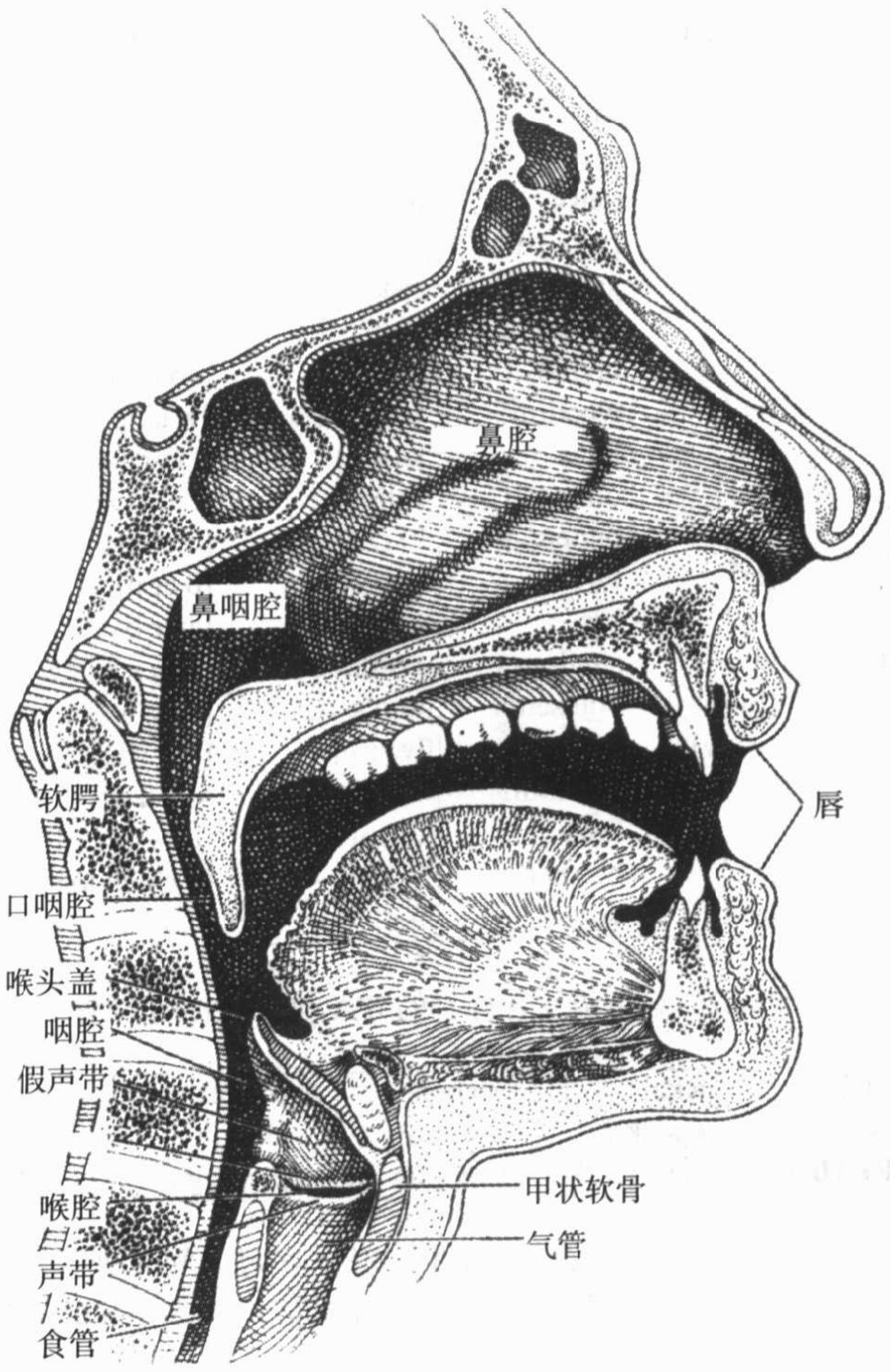


图4.4 发音器官侧面图，本图取自 Sundberg(1977)，©Scientific American, Laszlo Kubinyi 制作

语音可分为辅音（consonant）和元音（vowel）两大类。这两类语音都是空气通过口腔、咽腔或鼻腔时的运动而产生的。辅音产生时要以某种方式限制和阻挡气流的运动，可以是浊音或清音。而元音在产生时受到的阻挡较小，一般是浊音，比辅音响亮，延续时间较长。在技术上，这些术语的使用与普通用法相似；[p]，[b]，[t]，[d]，[k]，[g]，[f]，[v]，[s]，[z]，[r]和[l]等是辅音；[aa]，[ae]，[aw]，[ao]，[ih]，[aw]，[ow]和[uw]等是元音。半元音（semivowel）兼具辅音和元音的某些性质，例如[y]和[w]，它们像元音那样是浊音，但又像辅音那样发音时间较短，不成音节。

4.1.2 辅音：发音部位

由于辅音是气流以某种方式受到阻挡而形成的，根据阻挡形成的部位不同可以把不同的辅音区别开来。最大阻挡形成的部位称为辅音的发音部位（place of articulation）。图 4.5 是在语音自动识别中经常使用的发音部位图，这是一种把语音分成等价类的有效方法。

- **唇音（labial）**：阻挡主要在双唇的发音部位形成的辅音，称为双唇音（bilabial）。英语中的双唇音有[p]（如*possum*，负鼠），[b]（如*bear*，熊），[m]（如*marmot*，土拨鼠）。上齿背压住下唇，气流从上齿缝流出而形成的辅音，称为唇齿音（labiodental）。英语的唇齿音有[v]和[f]。

- **齿音（dental）：** 舌头顶住牙齿而形成的辅音称为齿音（dental）。英语中主要的齿音是[θ]和[ð]。发音时，舌头位于牙齿后面，舌端轻轻地置于齿间。
- **齿龈音（alveolar）：** 齿龈是口腔顶部上齿后面的部分。大多数说美国英语的人发[s]，[z]，[t]和[d]等辅音时，都是用舌尖顶住齿龈的。
- **上腭音（palatal）：** 上腭（palate）处于齿龈后面口腔的顶部。发龈腭音（palato-alveolar）[ʃ]（*shrimp*，小虾），[tʃ]（如*chinchilla*，南美栗鼠），[ʒ]（如*Asian*，亚洲人）和[dʒ]（如*jaguar*，美洲虎）时，舌叶向齿龈后部的上腭隆起。发上腭音（palatal）[y]（如*yak*，牦牛）时，舌面隆起接近上腭。
- **软腭音（velar）：** 软腭（velum）是口腔顶部最后面的可动肌肉盖面部分。发[k]（如*cuckoo*，布谷鸟），[g]（如*goose*，鹅）和[ŋ]（如*kingfisher*，翠鸟）时，舌头后部向上隆起接近软腭。
- **喉音（glottal）：** 发喉塞音[ʔ]时，声带合起，喉头关闭。

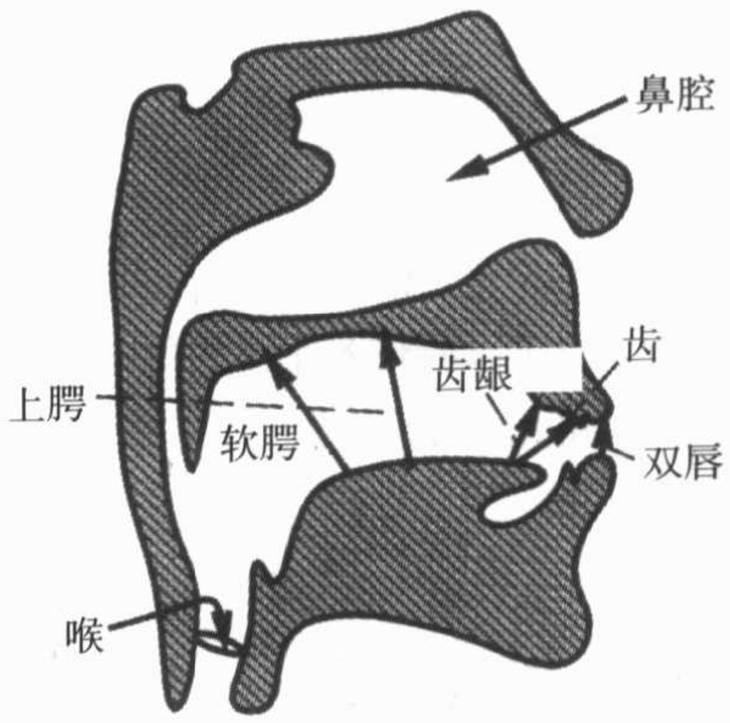


图4.5　英语的主要发音部位

4.1.3　辅音：发音方法

辅音也可以通过气流的阻挡方式的不同来区分（例如，气流是全部被阻挡还是部分被阻挡等）。这样的特征称为发音方法（manner of articulation）。把发音部位与发音方法结合起来一般就能充分地、惟一地鉴别一个辅音。英语辅音的发音方法如下：

- **塞音（stop）：** 塞音发音时，气流在短时间内被完全地阻塞。当空气解除阻塞时，就会发出爆破的声音。阻塞阶段称为成阻（closure），爆破阶段称为除阻（release）。英语中有浊塞音（如[b]，[d]和[g]）和清塞音（如[p]，[t]和[k]）。塞音又称为爆破音（plosive）。无论是ARPAbet还是IPA音标系统，都可以使用严式音标来清楚地区分塞音发音过程中的成阻部分和除阻部分。例如，[p]，[t]和[k]的成阻在ARPAbet中可以分别表示为[pcl]，[tcl]和[kcl]，在IPA中可以分别表示为[pˈ]，[tˈ]和[kˈ]。当使用这种严式音标时，在ARPAbet中，不加标记的[p]，[t]和[k]只表示这些辅音的除阻。在本章中我们不使用这样的严式音标。
- **鼻音（nasal）：** 发鼻音[n]，[m]和[ŋ]时，软腭下降，气流通过鼻腔流出。
- **擦音（fricative）：** 发擦音时气流被压缩，但是并没有被完全切断。由于压缩而振动的气流会产生特殊的摩擦声。英语的唇齿擦音[f]和[v]就是上齿压下唇时，被压缩的气流从上齿间摩擦而形成的。发齿擦音[θ]和[ð]时，气流顺着舌面从齿间摩擦而出。发齿龈擦音[s]和[z]时，舌尖顶住齿龈迫使气流从齿边流出。发龈腭擦音[ʃ]和[ʒ]时，舌头处于齿龈后面，迫使气流

通过舌面形成的凹槽摩擦而出。声音较高的擦音（如英语中的[s]，[z]，[ʃ]和[ʒ]）称为咝音（sibilant）。先摩擦而后阻塞的塞音，称为塞擦音（affricate），例如英语的[tʃ]（如*chicken*，鸡）和[dʒ]（如*giraffe*，长颈鹿）。

- **半元音（approximant）**：在发半元音时，参与发音动作的两个发音部位十分接近，但是还没有接近到能引起强气流的程度。发英语的[y]（*yellow*）时，舌头很接近口腔的上部，但是还没有接近到能够发出像擦音那样的形成强气流的程度。发英语的[w]（如*wormwood*，苦艾）时，舌根很接近软腭。在美国英语中，发[r]时至少有两种方法，或者舌尖延伸接近上腭，或者整个舌面隆起接近上腭。发英语的[l]时，舌尖顶住齿龈或牙齿，舌头的一侧或两侧下降，使气流能够顺着舌头的侧面流出。[l]称为边音（lateral），因为发[l]时气流是从舌头的边上流出来的。
- **颤音（tap）**：发颤音[ɾ]时，舌头顶住齿龈做快速运动。在美国英语的多数方言中，单词*lotus*（[loʊɾəs]）中间的辅音[t]就是一个颤音。但英语方言区的很多人说话时都把这个单词中的颤音用辅音[t]来替代。

4.1.4 元音

正如辅音一样，元音也可以通过发音部位来描述。元音有两个重要的参数：一个参数是发元音时舌位的高低（height），它大致相当于舌头最高部分所处的位置，另一个参数是发元音时嘴唇的形状（圆唇或者不圆唇）。图4.6说明了不同元音的舌位。

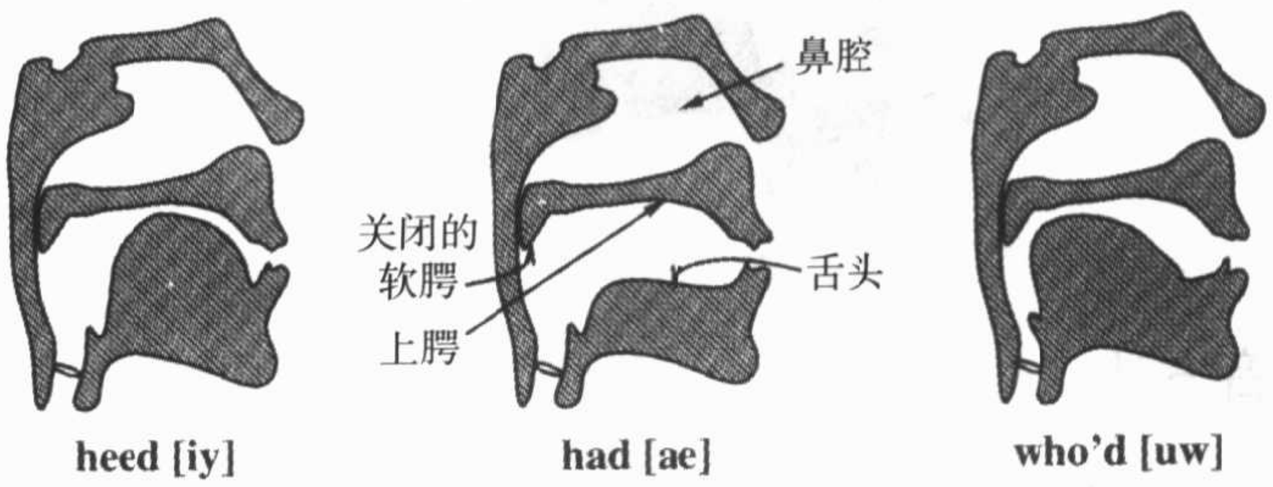


图4.6 英语中三个元音的舌位：前高[iy]、前低[ae]和后高[uw]；舌位依据Ladefoged(1996)

例如，发元音[i]时，舌头的最高点对着口腔的前部。反之，发元音[u]时，舌头的最高点对着口腔的后部。舌位处于前面的元音称为前元音（front vowel），舌位处于后面的元音称为后元音（back vowel）。注意，[I]和[ɛ]都是前元音，[I]的舌位比[ɛ]的舌位高。舌位最高点相对高的元音称为高元音（high vowel），舌位最高点的值处于中或低的元音，分别称为中元音（mid vowel）或低元音（low vowel）。

图4.7是不同的元音舌位高度的图式描述，称为元音舌位图。之所以说它是图式描述，是由于高度这种抽象的特性只是实际舌位情况的粗略描述；事实上，只有实际的舌位情况才是元音的声学性质的精确反映。注意，图式描述中有两类元音，一种元音的舌位高低用点来描述，另一种元音的舌位则用矢量来表示。在发音的过程中舌位发生变化的元音称为双元音（diphthong）。英语中的双元音很丰富；用IPA来记音时，双元音要用两个符号来表示，例如*hake*（鳕鱼）中的双元音[eɪ]，*cobra*（眼镜蛇）中的双元音[oʊ]。

描述元音发音的第二个参数是嘴唇的形状。有些元音发音时嘴唇要变圆（嘴唇形状与吹口哨时相同）。这些圆唇元音包括[u]，[ɔ]和双元音[oʊ]。

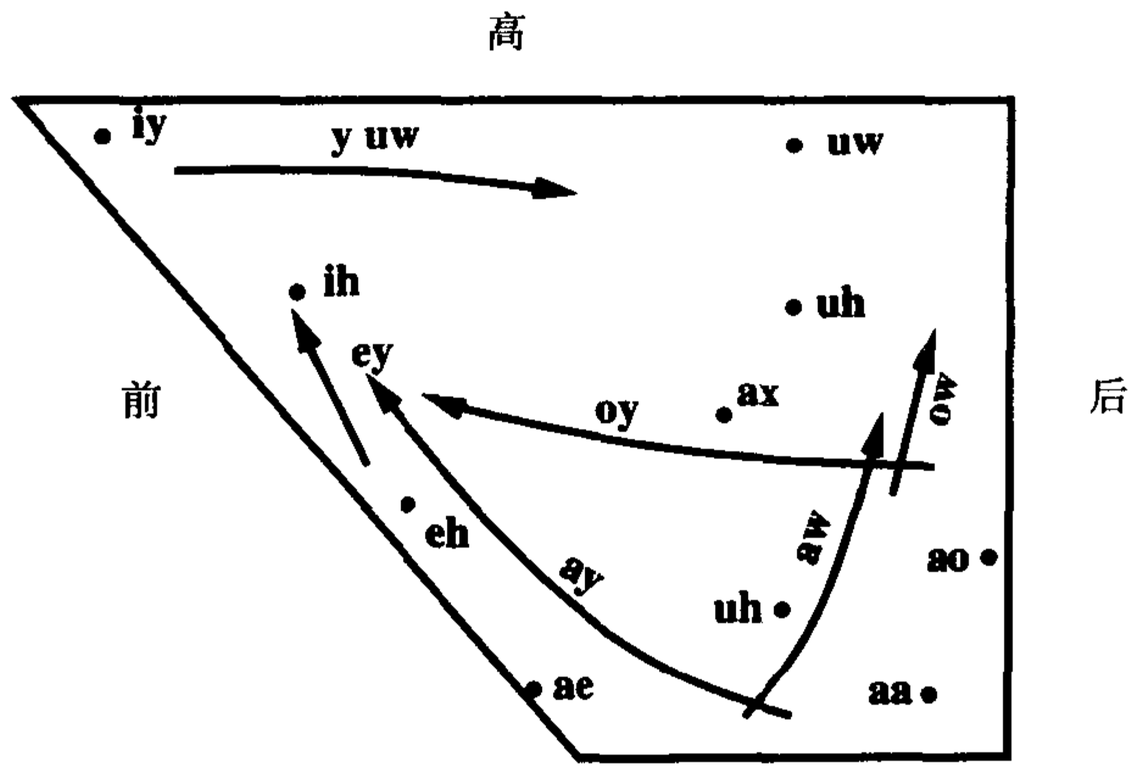


图 4.7　英语元音的性质描述图(Ladefoged, 1993)

4.1.5　音节

辅音和元音结合成音节（syllable）。关于音节，目前学术界还没有完全一致的、公认的定义。粗略地说，音节是一个元音之类的音和它周围的一些联系非常紧密的辅音结合而成的。在IPA中，用期间符号[.]来表示音节，其中的点把不同的音节分开，所以*parsley*（欧芹）和*catnip*（猫薄荷）有两个音节（分别为['par.sli]和['kæt.nɪp]），*tarragon*（龙蒿叶）有三个音节（['tæ.rə.gan]），而*dill*（莳萝）只有一个音节（[dɪl]）。一个音节通常由音节头（onset）后面跟着一个或多个元音，再跟着音节尾（coda）组成；音节头是音节开头的随选的一个辅音或一组辅音；音节尾是结尾的一个辅音或辅音序列。因此，在音节[dɪl]中，d是音节头，l是音节尾。把一个单词分割为音节的工作称为音节切分（syllabification）。虽然已经有音节切分的算法，但音节切分仍然是一个困难的问题，因为目前还没有一个公认的关于音节边界的定义。此外，尽管判定一个单词中究竟有多少个音节一般来说还是很清楚的，但Ladefoged(1993)指出，英语中有些单词，如meal（膳食），teal（水鸭），seal（图章），hire（租用），fire（火）和hour（小时），既可以被看成一个音节，也可以被看成两个音节。

在美国英语的自然句子中，某些音节显得比其他的音节更加突出。这样的音节称为重音音节（accented syllable）。重音音节之所以显得突出，其突显度之所以高，是因为它们更响、更长，它们与音高的运动有关联，或者它们与上述各种因素的组合有关联。由于重音对于意义有重要的作用，要确切地理解为什么说话人要选择某个特定的音节为重音，通常是非常复杂的。但是，重音中有一个重要因素往往需要在语音词典中表示出来，这个因素就是词重音（lexical stress）。当一个单词有重音音节时，具有词重音的音节就要发得更响一些、长一些。例如，单词*parsley*的词重音是在第一个音节，而不是在第二个音节。所以，当在句子中要强调*parsley*的时候，第一个音节就要发得有力一些。我们在音节前面加符号[']表示该音节是词重音（例如，['par.sli]）。词重音的不同会影响到词义。例如，单词*content*可以是名词（意思是“内容”），也可以是形容词（意思是“满意的”）。当单独发音的时候，这两个意思的发音是不同的，因为它们有不同的重音音节（名词发音为['kɑn.tɛnt]，形容词发音为[kən.'tɛnt]）。当单独发音时，其他的例子还有*object*（名词发音为['ɑb.dʒɛkt]，意思是“物体”；动词发音为[əb.'dʒɛkt]，意思是“反对”）。更多的例子参见Cutler(1986)。在第17章中，我们将讨论同形词（homographs）的自动歧义消解问题。在4.7节中，我们还要再讨论韵律（prosody）的作用问题。

4.2 音位和音位规则

'Scuse me, while I kiss the sky
Jimi Hendrix, *Purple Haze*
'Scuse me, while I kiss this guy
（同一抒情诗中经常听错的话）

并不是所有的[t]的发音都是完全相同的。在不同的上下文中，同一个音的发音情况也会有所不同。例如，在单词*tunafish*（金枪鱼）和*starfish*（海星）中的[t]的发音就是不完全相同的。在*tunafish*中的[t]是送气音（aspirated）。发送气音时，在发塞音的成阻之后，在发下一个元音的音节头之前，有一个声带不振动的阶段。因为声带不振动，在[t]之后和下一个元音之前，送气音的发音就像吐气一样。反之，如果在[t]前面有一个开头的[s]，那么这个[t]就是非送气音（unaspirated）；所以，在*starfish*[stɑrfiʃ]中的[t]在发塞音[t]的成阻之后，没有一个声带不振动的阶段。[t]发音的这种变化情况是可以预测的：在英语中，当[t]处于一个单词的开头或者一个非重读音节的开头时，它就是送气音。这种变化情况也发生在[k]这个音中；在前面Hendrix的抒情诗的*sky*这个词中，[k]常常被错听为[g]，因为这个[k]和[g]都是非送气音。在非常严格的音标中，我们可以在单词或非重读音节开头的[t]（或[k]和[p]）之后加符号[ʰ]来表示送气。例如，单词*tunafish*可标为[tʰunəfɪʃ]（但是，ARPAbet没有表示送气音的手段）。

[t]还有另外一种随上下文而变化的情况。例如，当[t]出现在两个元音之间，特别是当前面一个元音为重读元音的时候，[t]要发为颤音。颤音是一种带声的音，发颤音的时候，舌尖向后卷起，快速地顶着齿龈。例如，*buttercup*（毛茛科）要发为[bʌrɚkʌp]/[b uh dx axr k uh p]，而不发为[bʌtɚkʌp]/[b uh t axr k uh p]。

[t]的另一个变化是当它处于齿音[θ]之前时，[t]要齿音化，发为[t̪]。这时，舌头不对着齿龈形成阻塞，而是舌头要接触牙齿的背面。

怎样来表示[t]这个音和它在不同的上下文环境中的不同变体之间的关系呢？我们一般是把这一类的发音变化称为一个抽象的类别-音位（phoneme），这个音位在不同的上下文环境中实现为不同的音位变体（allophones）。按传统的标音习惯，我们把音位写在两个斜杠//之间。在上面的例子中，把音位记为/t/，把音位变体分别记为[tʰ]，[ɾ]和[t̪]。所以，音位是对不同实际语音的一种归纳或抽象。通常我们把音位和词汇放在同一个平面上来处理，词表中的单词也是按照音位来标音的。当标记单词的发音时，是在比较宽的音位平面上来标音的。这种音标称为宽式音标（broad transcription），标音时不考虑可能预见到的语音的许多细节。我们也可以选择使用严式音标（narrow transcription），标音时要使用附加符号（diacritics）来表示语音的各种细节，包括它的音位变体。图4.8总结了/t/的一些音位变体；图4.9说明了IPA中最常用的一些附加符号。

音子	环境	例子	国际音标
[tʰ]	处于开头位置	*toucan*	[tʰukʰæn]
[t]	处于[s]或非重读音节之后	*starfish*	[stɑrfiʃ]
[ʔ]	处于词末或元音之后，[n]之前	*kitten*	[kʰɪʔn]
[ʔt]	有时处于词末	*cat*	[kʰæʔt]
[ɾ]	处于元音之间	*buttercup*	[bʌrɚkʰʌp]
[tˀ]	处于辅音之前或词末	*fruitcake*	[frutˀkʰeɪk]
[t̪]	处于齿音（[θ]）之前	*eighth*	[eɪt̪θ]
[]	有时处于词末	*past*	[pæs]

图4.8 在通用的美国英语中/t/的音位变体

附加符号			超音段符号		
◌ʰ	清音化	[ḁ]	ˈ	第一重音	[ˈpu.mə]
	送气音	[pʰ]	ˌ	第二重音	[ˈfoʊrəˌgræf]
◌	音节化	[l̩]	ː	长音	[aː]
◌̃	鼻音化	[æ̃]	ˑ	半长音	[aˑ]
◌̚	不除阻	[t̚]	.	音节分隔	[ˈpu.mə]
◌̪	齿音化	[t̪]			

图4.9 IPA的一些附加符号和超音段符号

音位和它的音位变体之间的关系通常可以写成音位规则（phonological rule）来表示。下面是一条关于齿音化的音位规则，我们使用了Chomsky and Halle(1968)的传统记法来表示这个音位规则：

$$/t/ \rightarrow [\underline{t}] \,/\, __\, \theta \tag{4.1}$$

在这样的记法中，表层的音位变体都出现在箭头的右边，语音前后环境使用下划线（_）前后的符号来表示。这些规则有点像第3章中的双层形态学规则，但由于它们没有使用多种类型的重写箭头，在可选规则或必选规则之间是有歧义的。下面是一个颤音化规则：

$$\left/ \begin{Bmatrix} t \\ d \end{Bmatrix} \right/ \rightarrow [\mathrm{r}] \,/\, \acute{\mathrm{V}} __\, \mathrm{V} \tag{4.2}$$

4.3　音位规则和转录机

第3章中曾经说明过，拼写规则可以通过转录机来实现。音位规则同样也可以通过转录机来实现；事实上，Johnson(1972)和Kaplan and Kay(1981)在有限状态模型方面的工作与其说是建立在拼写规则的基础上的，不如说是建立在音位规则的基础上的。不同的计算音系学模型都以不同的方式使用了有限自动机来实现音位规则。我们将介绍第3章中使用过的Koskenniemi(1983)的双层形态学，有兴趣的读者还可以研究其他新模型①。在第3章中，我们曾经给出了双层规则的例子，但没有讨论为什么要用这样的规则，也没有讨论传统的顺序规则和双层规则之间的差异。现在我们就从比较这样的差异开始。

第一个例子如图4.10所示，描述了一个模拟简单的颤音化规则（4.3）的转录机：

$$/t/ \rightarrow [\mathrm{r}] \,/\, \acute{\mathrm{V}} __\, \mathrm{V} \tag{4.3}$$

图4.10中的转录机接收那些颤音发生在正确位置（处于重读元音之后，非重读元音之前）的任何符号串，并且拒绝那些不发生颤音化的符号串，或者拒绝那些颤音发生在不正确位置的符号串。当然，颤音化的因素还要复杂得多，我们将在5.7节中进一步讨论有关内容。

在传统的音位系统中，很多不同的音位规则在词汇形式与表层形式之间使用。有时，这些规则彼此会发生相互作用；一个规则的输出会影响另一个规则的输入。在一个转录机系统中，实现规则相互作用的一个途径就是使该系统内的各个转录机进行层叠式连接（cascade）。例如，用于处理英语名词复数后缀*-s*的各个规则就要进行这样的层叠式连接。这个后缀在[s], [ʃ], [z]或[ʒ]之后的发音为[ɨz]（因此，*peaches*要读为[pitʃɨz]，*faxes*要读为[fæksɨz]），在清音后的发音为[s]（*cats*要读为[kæts]）。我们写出音位规则来描述不同上下文中语素的情况，从而模拟这样的变化。首先，要从这三个形式（s, z和ɨz）中，选出这个后缀的“词汇”发音。为了简化规则的写法，我们只选择z。然

① 例如，Bird and Ellison(1994)关于自动切分音系学的多层级表示模型，在这个模型中，每个音位层级用一个有限状态自动机来表示，自动切分的联系用两个自动机的共时协同来表示。

后，写两条音位规则。一条规则与前一章中的 E- 插入拼写规则相似，在语素末尾的咝音之后和复数语素[z]之前，插入一个[ɨ]。另一条规则用于确保这个后缀 *-s* 在清辅音之后读为[s]。

$$\varepsilon \rightarrow \text{ɨ} \,/\, [\text{+sibilant}] \; \hat{} \; __ \; \text{z} \; \# \tag{4.4}$$

$$\text{z} \rightarrow \text{s} \,/\, [\text{-voice}] \; \hat{} \; __ \; \# \tag{4.5}$$

其中，[+sibilant]表示“是咝音”，[-voice]表示“不是浊音”（即“是清音”）。

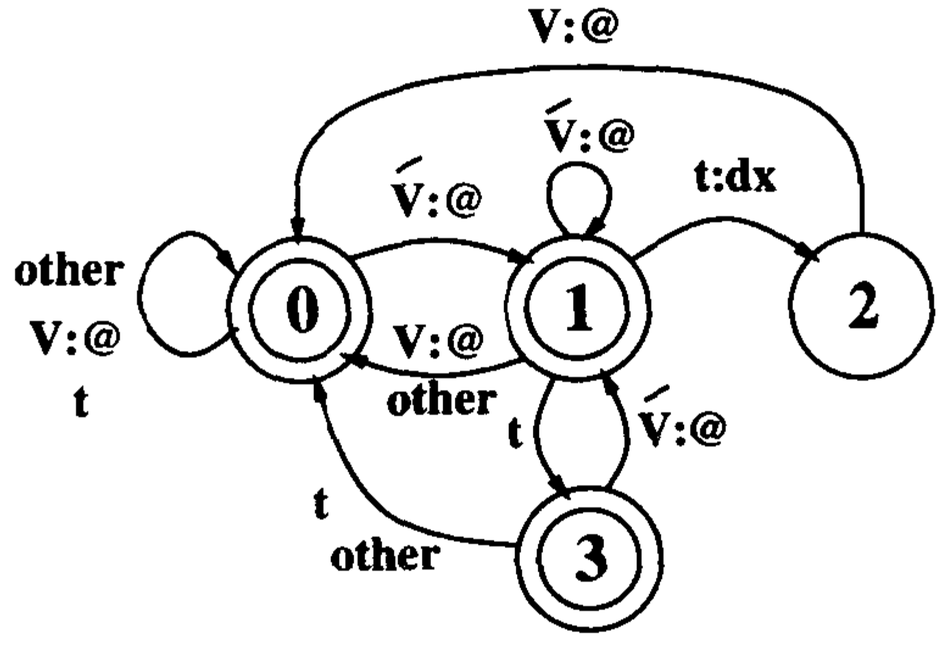


图 4.10 表示英语颤音化的转录机：ARPAbet“dx”表示颤音，符号 other 表示“在该转录机中的其他地方没有用过的任何合理的偶对”，符号@表示“在弧的任何地方都没有用过的任何符号”

这两个规则必须是有序的；规则（4.4）必须在规则（4.5）之前使用。这是因为规则（4.4）的环境中包含 z，而规则（4.5）却要改变 z。我们来考虑词汇形式 *fox* 与复数后缀 *-s* 毗连时使用这两个规则的情况。

词汇形式：	fɑks^z
使用规则（4.4）：	fɑks^ɨz
不能使用规则（4.5）：	fɑks^ɨz

如果把清音化规则（4.5）的顺序排在前面，就会得到错误的结果（这个不正确的结果是什么？）。在这种情况下，一个规则破坏了另一个规则的使用环境，使得另一个规则无以逞其能，这样的情况称为釜底抽薪（bleeding）[①]。

词汇形式：	fɑks^z
使用规则（4.5）：	fɑks^s
不能使用规则（4.4）：	fɑks^s

正如在第 3 章中说过的，这样的一些规则可以用转录机来表示。由于规则是有序的，所以转录机也有必要排序。例如，如果按照层叠式连接的方式来安排转录机的顺序，那么第一个转录机的输出就要送入第二个转录机，作为第二个转录机的输入。

通过这样的办法，很多规则可以进行层叠式连接。在第 3 章中我们说过，这样的层叠式连接，特别是当它的层级太多时，是很不容易使用和运行的，所以若干个转录机的层叠式连接通常是组合（composing）这些不同的转录机，使它们构成一个比较复杂的单独的转录机。

① 如果我们选择 *-s* 的词汇发音为[s]，而不是[z]，那么我们就会反过来写规则，把浊音之后的 *-s* 浊音化，但这个规则也要排序；把顺序简单地翻转一下。

在第3章中介绍过的Koskenniemi的双层形态学是解决规则顺序问题的另一个办法。Koskenniemi (1983)注意到，在一种语法中绝大多数音位规则是彼此独立的，也就是说，规则之间的雪中送炭（feeding）和釜底抽薪（bleeding）的情况并不多见[①]。基于这样的考虑，Koskenniemi提出，音位规则可以并行地（in parallel）运行，而不必串行地（in series）运行。当规则之间出现交互（即出现雪中送炭和釜底抽薪的情况）时，我们可以通过稍微修改某些规则的办法来解决。Koskenniemi的双层规则可以被看成是在词汇－表层映射的良构（well-formedness）的情况下，表示陈述式限制（declarative constraint）的一种方法。

双层规则也不同于传统的音位规则，它运用四个不同的规则算符（rule operator）来清楚地区分规则是必选的（obligatory）还是随选的（optional）；⇔规则相当于传统的必选音位规则，⇒规则相当于传统的随选音位规则：

规则类型	解释
$a: b \Leftarrow c __ d$	在上下文$c _ d$中，a总是实现为b
$a: b \Rightarrow c __ d$	在上下文$c _ d$中，a可以实现为b
$a: b \Leftrightarrow c __ d$	在上下文$c _ d$中，a必须实现为b，其他情况下都不行
$a: b /\Leftarrow c __ d$	在上下文$c _ d$中，a永远不能实现为b

双层规则的最重要的直觉以及它们的机制之所以能够避免雪中送炭和釜底抽薪等情况，在于它们具有表示双层限制的能力。这种能力是由于使用了冒号（":"），关于冒号我们曾经在第3章中简略地接触过。符号$a{:}b$意味着一个词汇a映射到表层b。所以，$a{:}b \Leftrightarrow {:}c___$的意思是：在表层c之后，$a$实现为$b$。反之，$a{:}b \Leftrightarrow c{:}___$的意思是：在词汇$c$之后，$a$实现为$b$。在第3章中曾经讨论过，没有冒号的符号$c$等价于$c{:}c$，它的意思是：映射到表层$c$的词汇$c$。

图4.11直观地说明了双层规则的方法怎样避免了i-插入规则和z-清音化规则的排序问题。其基本思路是，z-清音化规则把词汇的z-插入规则映射到表层s，而i-规则也只涉及到词汇z。

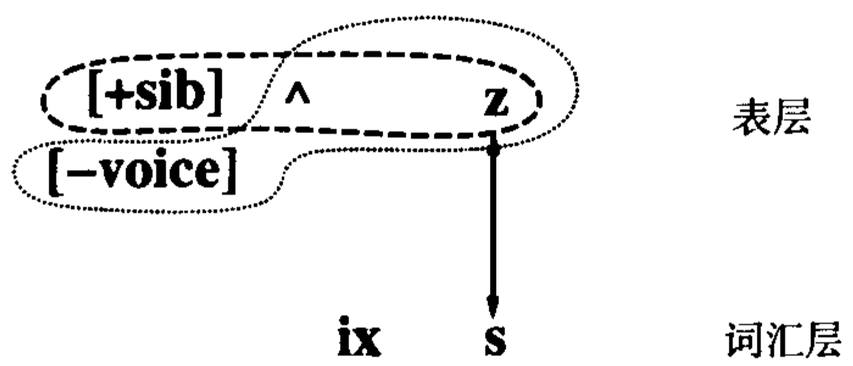


图4.11　i-插入规则和z-清音化规则两者都只涉及到词汇z，而不涉及到表层s

模拟这种限制的双层规则如式（4.6）和式（4.7）所示：

$$\varepsilon : \text{i} \Leftrightarrow \text{[+sibilant]: ˆ} __ \text{z: \#} \tag{4.6}$$

$$\text{z : s} \Leftrightarrow \text{[-voice]: ˆ} __ \text{\#} \tag{4.7}$$

我们在第3章中说过，已经有从规则到自动机的编译算法。Kaplan and Kay(1994)给出了这种算法的一般推导方法，Antworth(1990)给出了双层规则算法的推导方法。与双层规则相应的自动机如图4.12和图4.13所示。图4.12是根据第3章中的图3.14制作的，关于自动机的工作情况，参见第3章中的有关内容。注意，图4.12中的复数语素用z来表示，这意味着这里的限制z只在词汇层上表示，而不在表层上表示。

① 雪中送炭（feeding）是指一个规则给另一个规则创造使用的环境，这时，这个规则必须在另一个规则之前运行。

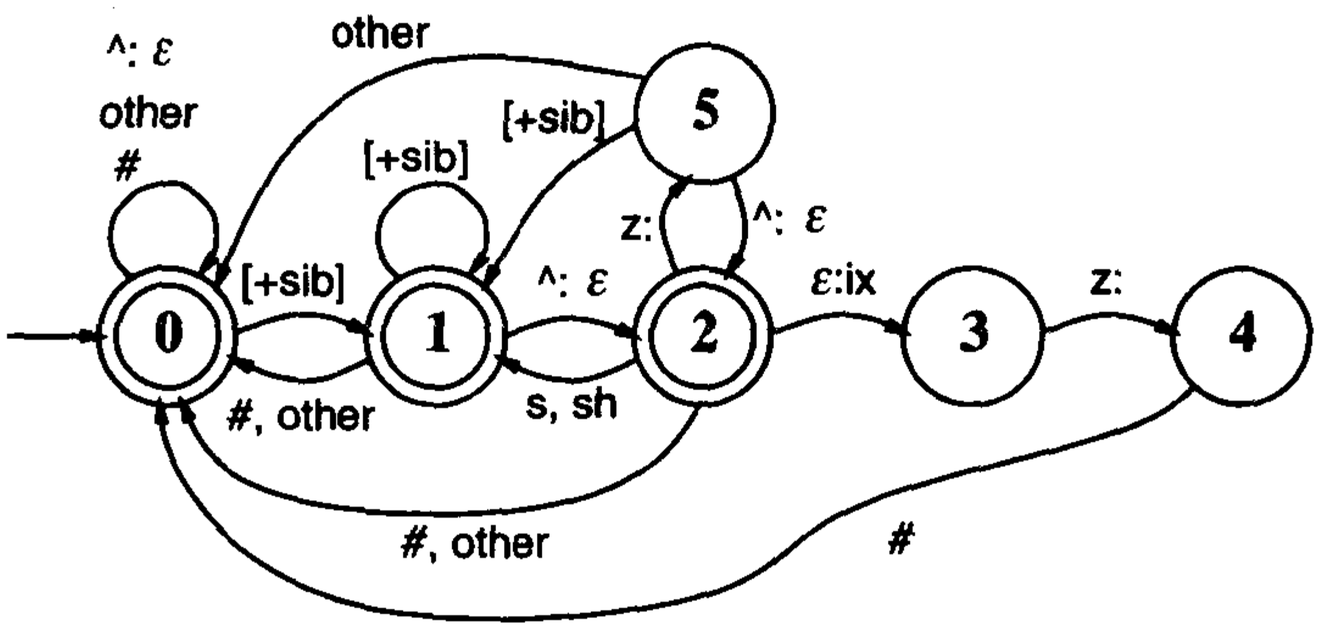


图 4.12　i- 插入规则（4.4）的转录机。这个规则的意思是：当语素以咝音结尾并且后面的一个语素是 z 时，插入[ɪ]

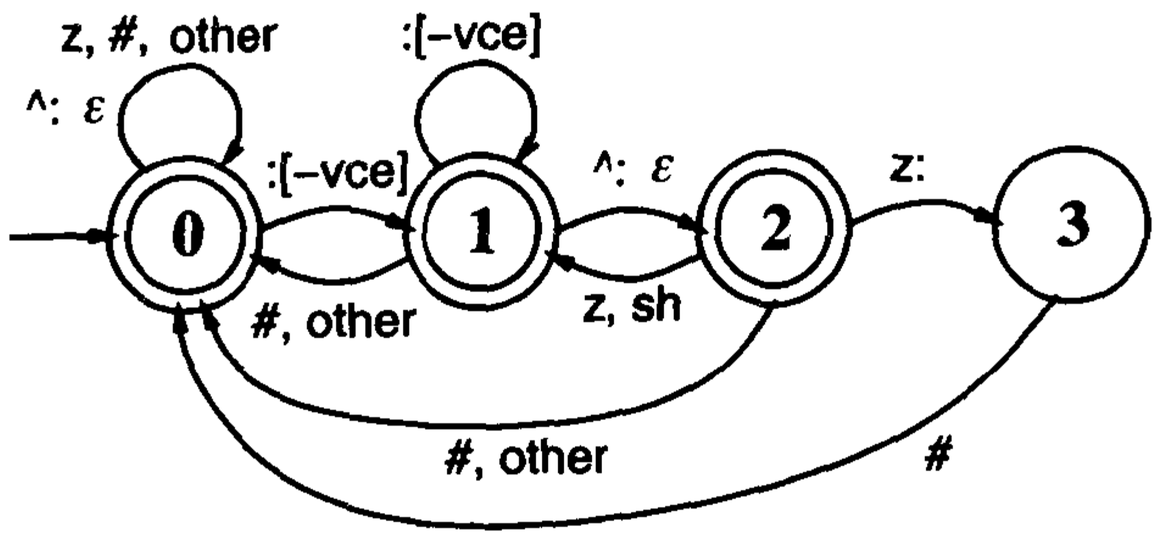


图 4.13　z- 清音化规则（4.5）的转录机。这个规则可总结为：如果语素 z 跟随着一个以清辅音结尾的语素，那么这个语素 z 变为清音

图 4.14 说明了当输入为[fɑks^z]（图中使用 ARPAbet 的记法[f aa k s ^ z]）时，两个自动机并行运行的情况。注意，在两个自动机中都假定存在着默认值映射 ^: ε 以消除语素的边界，并且两个自动机都以可接收状态终结。

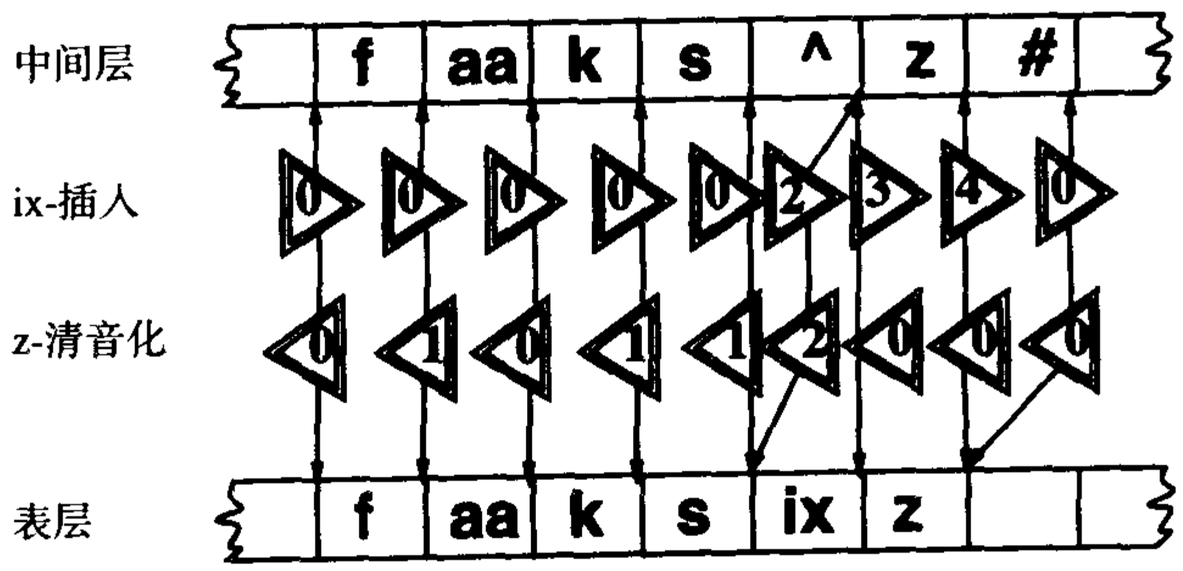


图 4.14　用于 i- 插入规则和 z- 清音化规则（4.4）的转录机并行运行的情况

4.4　计算音系学中的一些高级问题

4.4.1　元音和谐

前面讲过的颤音化、i- 插入、z- 清音化都是相对简单的音位规则。在本节中，我们将使用计算音系学中的双层模型或有限状态模型来模拟一些更复杂的现象；如果读者已经具有关于音系学的

知识，学习本节将会容易一些。Yokuts语的Yawelmani方言是加利福尼亚州讲的一种美洲土著语言，这种语言具有非常复杂的音位系统。具体来讲，这种语言有三个与元音有关的音位规则，这些规则必须按照传统的音系学来排序，并且这些规则之间的相互作用表现出有限状态音系学在实际使用中的复杂性。这些规则是Kisseberth(1969)在Newman(1944)的研究工作的基础上，根据传统的Chomsky and Halle(1968)的格式首先提出的。

首先，Yokuts语（以及其他一些语言，如土耳其语和匈牙利语）中有一种音位现象，称为元音和谐（vowel harmony）。元音和谐是元音随着它邻近的其他元音而改变其发音形式的过程。在Yokuts语中，作为后缀的元音要改变其形式以便在舌位移后（backness）和圆唇化（roundness）等方面与它前面的词干的元音取得一致。也就是说，如果词干的元音是后元音/u/，那么后缀中的前元音/i/的发音要发得像后元音一样，发为后元音/u/，以便与词干中的后元音取得和谐(Cole and Kisseberth, 1995)[①]：

词汇层		表层	注解
dub+hin	→	dubhun	“处于混乱状态，非将来时”
xil+hin	→	xilhin	“用手带领，非将来时”
bok'+al	→	bok'ol	“想吃”
xat'+al	→	xat'al	“想找”

这样的元音和谐规则还有一个限制：只有当后缀元音和词干元音的舌位高度一样时，才能使用这个规则。例如，在上面的例子中，/u/和/i/两者都是高元音，/o/和/a/两者都是低元音。

元音和谐的第二个规则是元音低化（lowering），这个规则使得长高元音变为低元音；例如，在下面的第一个例子中，/u:/变为/o:/。

词汇层		表层	注解
ʔu:t'+it	→	ʔo:t'ut	“偷窃，被动式不定过去时”
mi:k'+it	→	me:k'+it	“吞咽，被动式不定过去时”

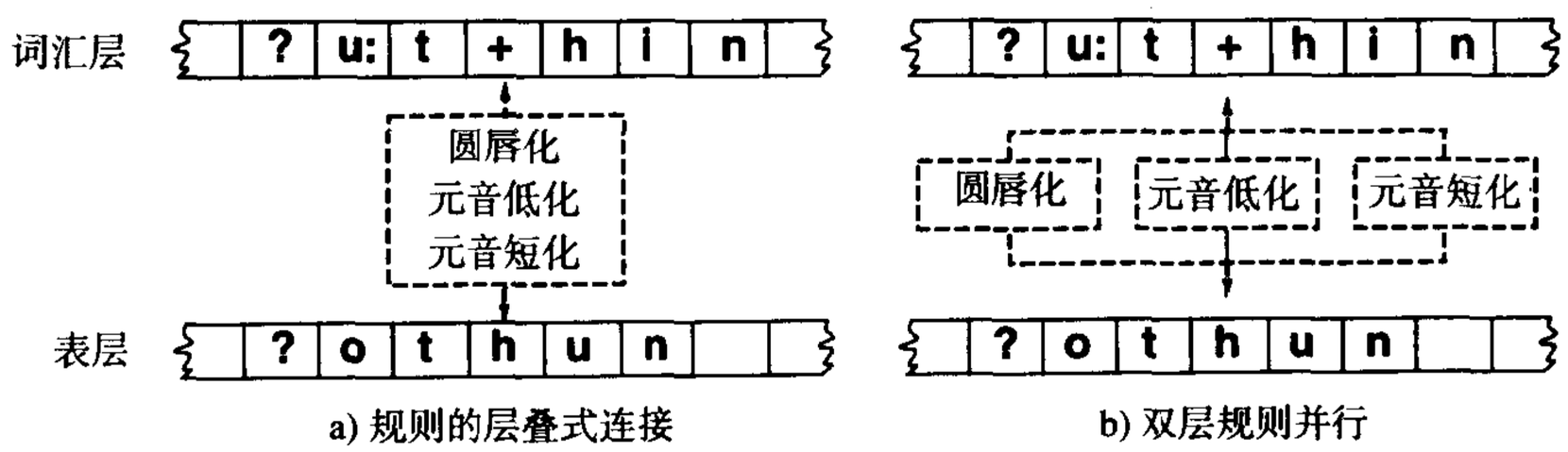
元音和谐的第三条规则是元音短化（shortening），这个规则使得出现在闭音节中的长元音变短：

词汇层		表层
sa:p+hin	→	saphin
sudu:k+hin	→	sudokhun

正如英语中的i-插入规则和z-清音化规则一样，Yokuts语的这些规则也是有顺序的。元音和谐规则应该在元音低化规则之前，因为在词汇形式/ʔu:t'+it/中的/u:/引起/i/变为/u/之后，才能使用元音低化规则得到表层形式/ʔo: t'ut /。元音低化规则应该在元音短化规则之前，因为在/sudu:k+hin/中的/u:/要先低化之后才能短化为/o/；如果先短化，那么在表层中就应该是/u/。

Goldsmith(1993)和Lakoff(1993)独立地观察到Yokuts语的数据可以用转录机之类的工具来建立模型；Karttunen(1998)提出论据，说明Goldsmith和Lakoff的限制或者可以用三个规则的串行层叠式连接来表示，或者可以把三个规则用双层形式化方法并行表示。图4.15说明了这两种不同的结构方式。正如前面介绍过的有关双层模型的例子，这些规则工作时有时涉及到词汇的上下文，有时涉及到表层的上下文。作为练习，读者可以自己写出这样的规则。

① 为了解释上的方便，我们这里忽略了系统的某些部分，如元音的次级特征的说明等(Archangeli, 1984)。



a) 规则的层叠式连接　　b) 双层规则并行

图 4.15　把圆唇化、元音低化和元音短化规则结合起来处理 Yokuts 语的 Yawelmani 方言

4.4.2　模板式形态学

音系学和形态学中的有限状态模型也可以用于模板式形态学(Templatic Morphology)。所谓模板式形态学，就是一种非毗连性的形态学，它是西闪美特语言 (如阿拉伯语、希伯来语、古代叙利亚语) 中的共同现象。McCarthy(1981)提出了模板式形态学，这种形态学可以使用不同的表示层次来模拟，Goldsmith(1976)把这种表示层次称为层级 (tier)。Kay(1987)提出了这种层级的计算模型，在这个模型中使用了一个特殊的转录机，它读的带子不是 2 个而是 4 个，如图 4.16 所示。

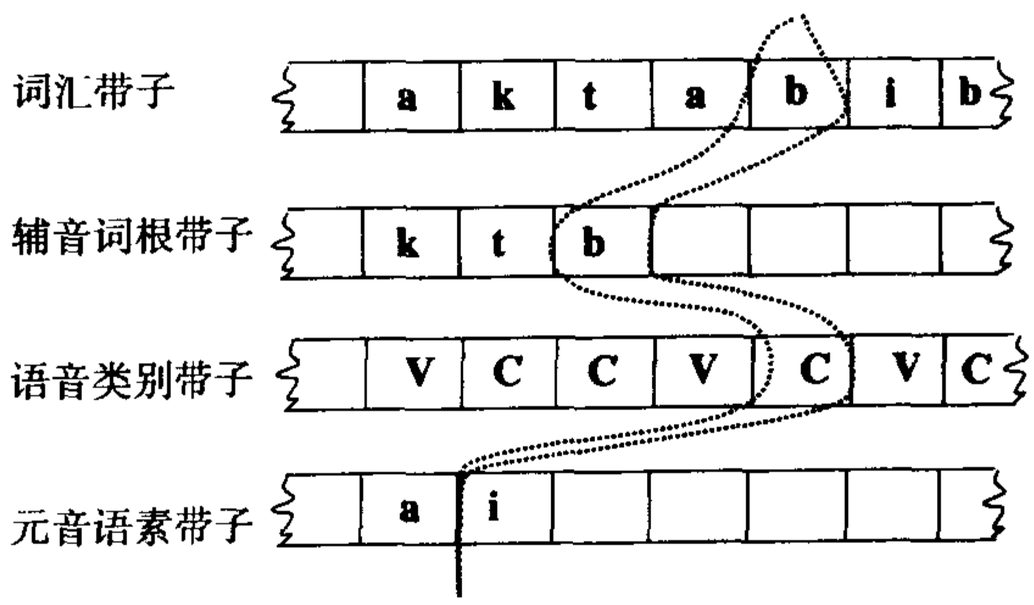


图 4.16　模板式 (非毗连) 形态学的有限状态模型。根据 Kay(1987)和 Sproat(1993)修改而成

这种模型的技术难点是设计能够正确地把带子上的不同符号串对应起来的机器; Kay根据直觉提出使用语音类别带子 (binyan tape) 来指导整个对应。Kay 的这种直觉引出了一系列关于闪美特语言有限状态模型的研究工作，例如，Beesley(1996)关于阿拉伯语的模型和Kiraz(1997)关于古代叙利亚语的模型。

Kornai(1991)最近的研究与 Bird and Ellison(1994)的研究表明，单带子的自动机 (即不是四带子或双带子的转录机的有限状态自动机)可以用于模拟模板式形态学和模拟那些使用基于层级的自动切分表示方法来处理的其他类型的现象(Goldsmith, 1976)。

4.4.3　优选理论

在传统的音位推导中，我们使用底层词汇形式和表层形式。因此，音位系统包含一个组成成分，这个组成成分是一个规则系列，这个规则系列把底层形式映射到表层形式。优选理论 (Optimality Theory，简称 OT)(Prince and Smolensky, 1993)提出了关于音位推导的另一种思路。优选理论的基础是两个功能 (GEN 和 EVAL) 与一套有等级而又可违反的限制 (CON)。给定一个底层形式，功能 GEN 就产生所有可想像的表层形式，尽管它们可能是从输入而来的不合法的表层形式。然后功能 EVAL 把按照限制等级排列的 CON 中的每个限制应用于表层形式，从而优选出最能满足限制要求的表层形式。

优选理论中的限制是在表层形式上良构的限制。例如，关于哪个片断跟随在哪个片断之后的音位顺序的限制，关于哪个音节结构是可允许的限制等。这样的限制也可以用来检查表层形式对于底层形式的忠实程度。

现在我们回过头来讨论我们感兴趣的复杂语言，例如Yawelmani语[①]。除了我们上面讨论过的元音和谐现象之外，Yawelmani语还有辅音序列规则中的音位顺序限制。具体地说，在一个表层的词中，不允许出现连续的三个辅音（CCC）。但是，有时一个词可包含两个连续的语素，第一个语素以两个辅音结尾，第二个语素以一个辅音开头（或者反过来）。语言怎样来解决这个问题呢？Yawelmani语采用删除其中一个辅音，或者在两个辅音之间插入一个元音的办法来解决这个问题。

例如，如果词干以C结尾，后缀以CC开头，那么就要把后缀的第一个C删除（“+”表示语素边界）：

$$\text{C-删除}\quad C \rightarrow \varepsilon / C + __C$$

这里是一个例子，其中语音结构为CCVC“被动结果补足语”的语素hne:l（实际上的底层形式为/hnil/），如果前面的语素以两个辅音结尾，那么它就要丢失开头的辅音C（为了对比，另一个例子中不丢失开头的辅音）：

底层语素	注解
diyel-ne:l-aw	“警戒-被动结果补足语-方位”
cawa-hne:l-aw	“呼唤-被动结果补足语-方位”

如果词干以CC结尾，后缀以C开头，那么语言中不是去掉C，而是插入一个元音来切断前面两个辅音的连接：

$$\text{V-插入}\quad \varepsilon \rightarrow V / C __ C + C$$

下面是一些例子，其中，当词根?ilk-“唱歌”和词根logw-“粉碎”后面接一个以C开头的后缀（如-hin“过去”），而不是以V开头的后缀时（如-en“将来”），在词根CC结尾的两个辅音之间要插入一个元音i。

表层形式	注解
?ilik-hin	“唱过歌了”
?ilken	“将要唱歌”
logiwhin	“粉碎了”
logwen	“将要粉碎”

Kisseberth(1970)认为，Yawelmani语中出现这两个特殊的规则并不是偶然的巧合（这里没有介绍其他有关的删除规则）。文中指出，这些规则之间在功能上是有联系的，具体地说，这些规则都是为了避免出现连续的三个辅音CCC这样的语音结构。解释这样的一般性原理的另一种办法是从音节结构的角度来看问题。Yawelmani语的音节只允许CVC或CV这样的形式（C表示辅音，V表示元音）。我们说，像Yawelmani语这样的语言不允许复杂的音节头（complex onset）或音节尾（complex codas）。从音节结构的角度来看，插入规则和删除规则都是为了让Yawelmani语的单词能够恰当地构成音节。由于CVCC这样的音节不允许在表层中出现，所以当CVCC词根在表层中出现时必须进行再音节化（resyllabified），重新构成音节。现在我们来讨论Yawelmani语的音节化和

① 下面的以Yawelmani语为例子对优选理论的解释大部分来自Archangeli(1997)和Jennifer Cole在1999年LSA语言研究所的讲义。

其他一些问题。注意，在下面的例子中，CVCC音节在进行表层的音节化时，把最后的辅音移位到了下一个音节的开头：

底层语素	表层音节化	注解
ʔilk-en	ʔil.ken	“将要唱歌”
logw-en	log.wen	“将要粉碎”
logw-hin	lo.giw.hin	“粉碎了”
xat-en	xa.ten	“将要吃”
diyel-hnil-aw	di.yel.ne:.law	“问-过去.结果.补足语.-方位”

现在我们来讨论优选理论。优选理论的基本思想是，语言对于音节结构这样的一些事情具有不同的限制。这些限制一般应用于表层形式。一个限制称为*COMPLEX，它的意思是“没有复杂的音节头和音节尾”。另一类限制要求表层形式等同于（忠实于）底层形式。因此，FAITHV的意思是：“不要删除或插入元音”，FAITHC的意思是：“不要删除或插入辅音”。给定一个底层形式，功能GEN产生一切可能的表层形式（也就是每种可能的音节形成时所具有的一切语音片段的删除和插入的可能情况），并且它们要按照使用这些限制的功能EVAL来排等级。具体说明如图4.17所示。

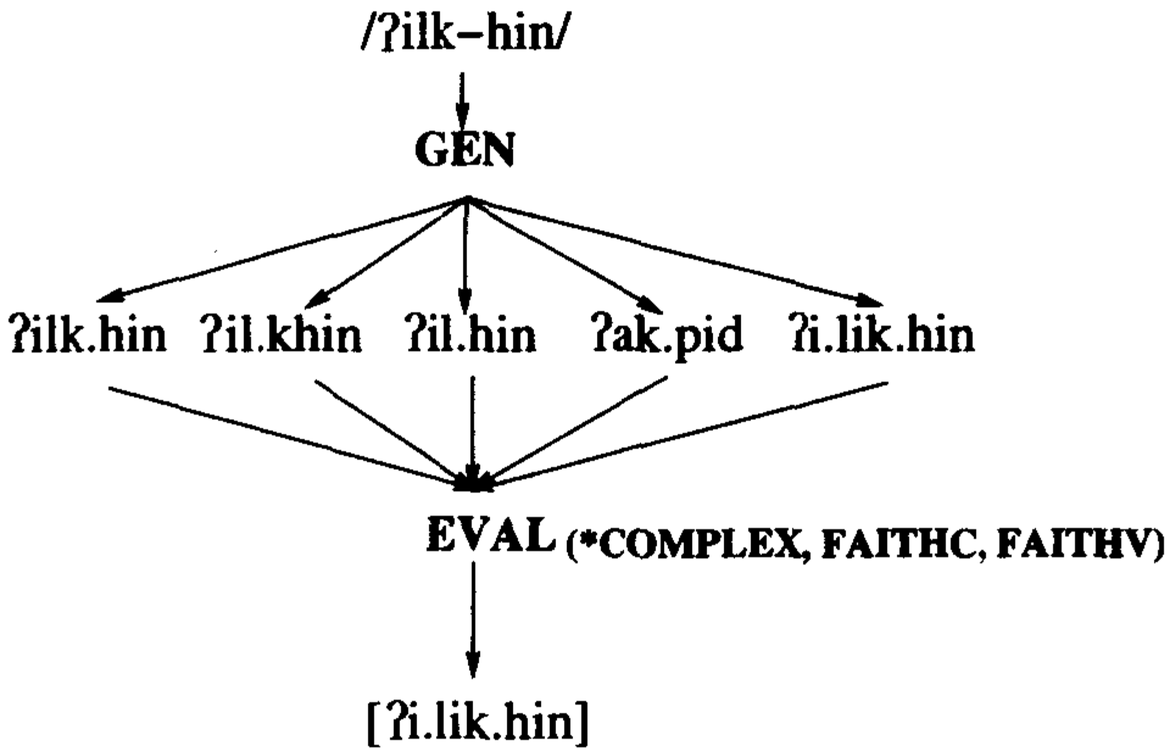


图4.17 在优选理论中一个派生的示意图(Archangeli, 1997)

功能EVAL工作时要根据等级的顺序来使用各种限制，优先的候选或者是不违背限制的候选，或者是比其他候选违背限制较少的候选。评价通常使用场景图（多元场景图）来表示。在场景图（tableau）中，左上边的项表示输入，限制按照等级的高低排在最高一行中，可能的输出写在最左一列中。尽管有无限数目的候选，传统上只显示最“接近”的候选；在场景图的下端，我们还写出了输出ʔak.pid，只是为了说明，甚至差别很大的表层形式也可以包括在我们的场景图中。如果一个形式违背了一个限制，就在相关的项中标以*号，符号!*表示致命的违背，会引起一个候选被取消。无关的限制的项（由于它们已经违背了等级最高的限制）都画了阴影。

/ʔilk-hin/	*COMPLEX	FAITHC	FAITHV
ʔilk.hin	*!		
ʔil.khin	*!		
ʔil.hin		*!	
☞ ʔi.lik.hin			*
ʔak.pid		*!	

优选理论的派生研究值得注意之处在于，这种理论假定派生中的限制具有跨语言的一般性。也就是说，一切语言中都有等同性（忠实性）的问题，都有简单符号优先性的问题，等等，只不过各种语言的表现形式不同而已。不同的语言在限制等级方面是有差异的。我们这里假定，在英语中FAITHC的等级高于*COMPLEX的等级（我们怎么知道这一点？）。

优选理论中的派生是否可以用有限状态转录机来实现呢？Frank and Satta(1999)在Ellison(1994)的奠基性研究的基础上，说明了：（1）如果GEN是一个正则关系（例如，其中不包含某种上下文无关的树），并且（2）如果在任何限制中所允许的犯规数量不能超过某个有限的数目，那么OT（优选理论）的派生就可以用有限状态的方式来计算。这里的第二条限制是相对的，因为OT理论中有一个我们尚未提到的性质：如果两个候选的犯规所涉及的限制数目恰好相同，那么就选择相关限制的犯规数目较小的候选作为取胜的候选。

用有限状态系统实现OT的一个方法是Karttunen(1998)根据前述的工作以及Hammond(1997)的工作提出来的。在Karttunen的模型中，GEN是作为有限状态转录机来实现的，它给出一个底层形式并产生一系列候选形式。例如，在上面的音节化例子中，GEN要根据辅音删除、元音插入和它们的音节化规则，生成输入变体的一切符号串。

每个限制都用一个过滤转录机（filter transducer）来实现，过滤转录机只允许通过那些满足限制条件的符号串。对于合法的符号串，转录机鉴别之后就用映射的方式来实现它。例如，*COMPLEX这个限制要通过一个转录机来实现，实现时映射任何的输入符号串到它自身，对于那些以两个辅音为音节头或音节尾的输入符号串，由于它们映射为空符号，不满足限制条件，将被转录机过滤。

这些限制可以按照层叠的方式来排列，其中级别高的限制先运行，如图4.18所示。

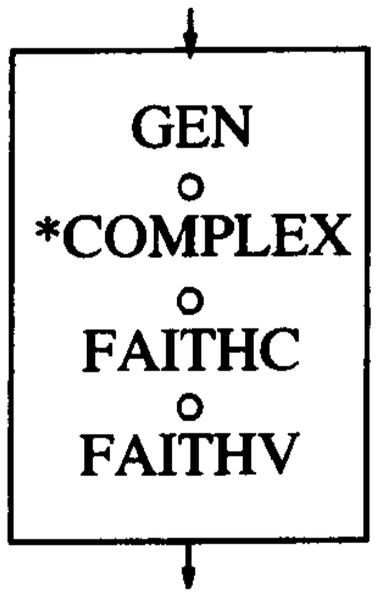


图4.18　Karttunen用有限状态层叠来实现OT的第一个版本Version #1（“严格的层叠”）

在图4.18中的层叠模型的流程是很严格的，凡是违反限制的任何候选，都要被这个限制转录机过滤掉。然而，在很多派生中，包括像?i.lik.hin这样的派生，甚至最佳的形式也会违反限制。图4.18中的层叠将会不正确地把它过滤掉，结果使得任何一个表层形式都被过滤得一干二净，什么也留不下来。Frank and Satta(1999)和Hammond(1997)都指出，有时应该强制地执行某个限制，使得它不至于把候选集减少到零。Karttunen(1998)提出了宽容组合算符（lenient composition operator）的概念，把这样的直觉形式化了。宽容组合把正则组合与一种称为优先合并（priority union）的运算结合起来。其基本思想在于，只要有任何候选满足限制，那么这些候选就可以像通常的情况那样通过过滤转录机。如果没有输出满足这样的限制，那么宽容组合将保持所有的候选。图4.19说明了这种思想的一般情况。有兴趣的读者可以参阅Karttunen(1998)，从中可以了解到这种思想的细节。关于OT的其他计算问题可以参阅Tesar(1995, 1996)，Fosler(1996)和Eisner(1997)。

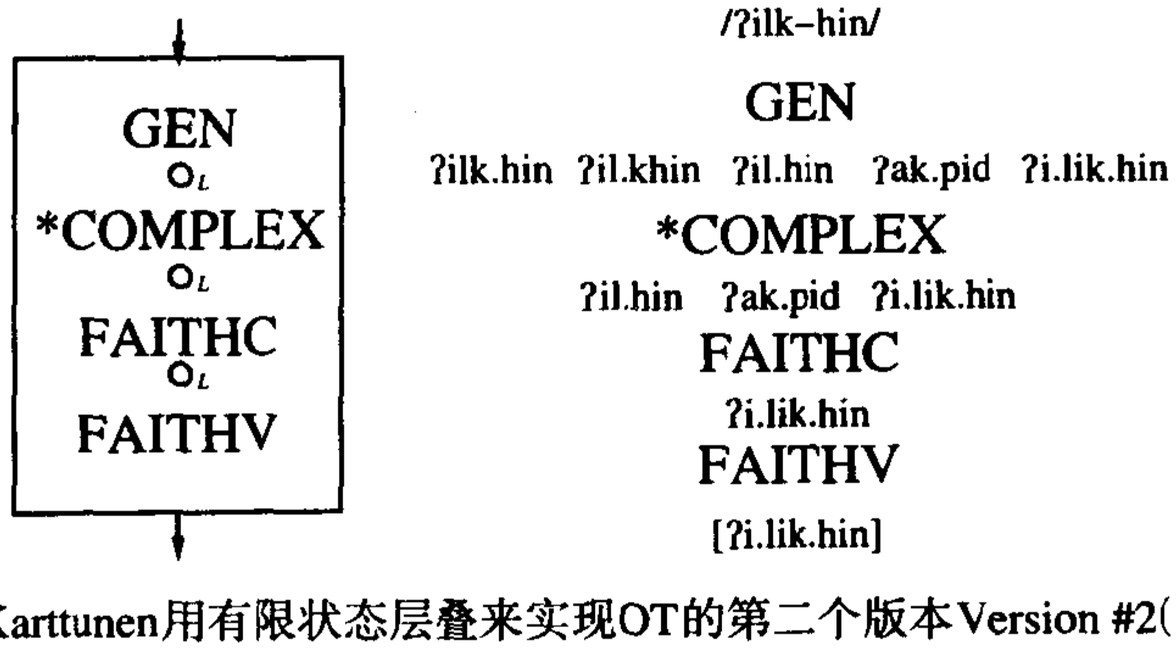


图 4.19 Karttunen用有限状态层叠来实现OT的第二个版本Version #2(“宽容的层叠”)，其中可以看到有一个候选通过了 FST 的每个限制

4.5 音位规则的机器学习

机器学习(machine learning)系统的任务在于，根据从某一领域中给定的某些数据以及其他信息，自动推导出该领域的一个模型。所以，一个能够学习音位规则的系统，至少要给出一系列单词(的表层形式)，以便从中进行推导。在有指导的算法(supervised algorithm)中，可以对其中的某些数据给出正确的答案,并使用这些答案推导出模型,然后这样的模型能够生成原先没有的新数据。在无指导的算法(unsupervised algorithm)中，只是纯粹通过数据来进行这样的推导。当无指导的算法找不到正确的分类标记时,可以给出关于规则的性质或者它们所要形成的模型的某些提示。例如，关于模式的形式可能是自动机这样的提示，这种提示称为学习偏爱(learning bias)。

本节将简短地介绍音位规则的无指导机器学习的一些模型。关于机器学习算法的更详细的描述是贯穿全书的。

Ellison(1992)指出，诸如辅音和元音的区别、语言的音节结构、元音和谐关系等概念都可以从一系列潜在的模型中选择最简单模型的系统进行学习而得到。系统的简单程度可以用选择最小编码长度或最大概率模型（将在第 6 章中定义这些术语）的方法来度量。Daelemans et al.(1994)使用基于实例的一般化算法(Aha et al., 1991)来学习荷兰语的重音；这种算法是有指导的算法，要求给出一定数量的单词以及这些单词的重音模式,从而推导出关于从单词中的轻音和重音音节的类型(轻音音节没有音节尾辅音，重音音节有一个音节尾辅音）映射到重音模式的一般性规则。Tesar and Smolensky(1993)指出，一个系统只需要给出优选理论中的辅音，但是不用给出这些辅音的等级，该系统可以使用简单的贪心算法（ greedy algorithm ）从数据中学习到辅音的等级。

Johnson(1984)给出了一个音位规则推导算法,这个算法是最早的音位推导算法之一。算法工作的规则形式是：

$$a \rightarrow b/C \tag{4.10}$$

这里的 C 是 a 周围的语音片断的特征矩阵。Johnson 的算法可建立限制性等式的系统，其中 C 必须同时满足正上下文 C_i 和负上下文 C_j。正上下文 C_i 是指在表层中 b 出现的上下文，负上下文 C_j 是指在表层中 a 出现的上下文。Touretzky et al.(1990)扩充了 Johnson 的方法，使用 Mitchell(1981)的版本空间(version space)算法来推导多映射结构中的音位规则，该方法与双层音系学很相似。和Johnson的方法一样，系统要观察单个语音片断的底层实现和表层实现。对于每个语音片断，系统都要用版本空间算法来查找对上下文的恰当说明。这个模型还有一个分离的算法，使用查找同一个词中的多个片断变化的方法来处理和谐效应，它在处理插入规则和删除规则方面比Johnson的算法更具有一般性。

Gildea and Jurafsky(1996)的算法试图推导出前面讨论过的表示双层规则的转录机。与Touretzky et al.(1990)的算法类似，Glidea 和 Jurafsky 的算法给出了底层形式和表层形式偶对的集合。该算法是根据 OSTIA(Oncina et al., 1993)算法提出来的，OSTIA 算法是有限状态转录机某个次类（称为 subsequential transducer，即亚序列转录机）的一种通用的学习算法。OSTIA 算法的通用性非常强，给出含有底层形式和表层形式偶对的大规模语料库，OSTIA算法自己就可以通过学习形成音位转录机。Gildea 和 Jurafsky 后来进一步提出了不依赖于领域的 OSTIA 系统，他们使用自然语言音系学中特有的三种类型的学习偏爱（learning biase），其中两个主要的学习偏爱是忠实性（底层片断实现时倾向于和表层尽量相似）和共同性（相似的片断具有相似性）。这样做出的系统能够通过学习形成转录机来处理美国英语颤音化或者德语辅音清音化问题。

最后，音系学的很多学习算法是概率算法。例如，Rilley(1991)和 Withgott and Chen(1993)提出决策树（decision-tree）的方法来处理片断的映射。每个片断都可以推算出一个决策树，这样的决策树可以根据上下文因素（诸如重音和前后包围的片断）对可能形成的片断进行分类。关于决策树和概率算法的一般原理将在第 5 章和第 6 章中介绍。

4.6　TTS 中从文本映射到语音

Dearest creature in Creation
Studying English pronunciation
I will teach you in my verse
Sounds like corpse, corps, horse and worse.
It will keep you, Susy, busy,
Make your head with head grow dizzy
...
River, rival; tomb, bomb, comb;
Doll and roll, and some and home.
Stranger does not rime with anger
Neither does devour with clangour.
...

G.N. Trenite (1870-1946), *The Chaos*,
Reprinted in Witten (1982)①

我们已经知道了英语语音的基本情况，并且知道了怎样给音位规则建立模型，现在来研究怎样把符合正词法的或书面的单词映射为它们的发音的问题。

4.6.1　发音词典

在这样的映射中，发音词典（pronunciation dictionary）是一个重要的组成部分。发音词典实际上既可以用于 ASR（自动语音识别）系统，也可以用于 TTS（文本-语音转换）系统，尽管由于这两种系统的需要不同，发音词典的内容可能会有所不同。

最简单的发音词典只包含由单词及其发音组成的一个词表。

单词	发音	单词	发音
cat	[kæt]	goose	[gus]
cats	[kæts]	geese	[gis]
pig	[pɪg]	hedgehog	['hɛdʒ.hɔg]
pigs	[pɪgz]	hedgehogs	['hɛdʒ.hɔgz]
fox	[fɑx]		
foxes	['fɑk.sɪz]		

这样的大规模的在线发音词典有三部，分别是 PRONLEX，CMUdict 和 CELEX。它们可用于语音识别，也可以用于语音合成。PRONLEX 发音词典(LDC, 1995)是为语音识别而设计的，包含

① 这是英语的顺口溜，翻译为汉语后将失去韵味，故原文照录，以便读者玩味。——译者注

90 694 个词形的发音，可覆盖多年来在华尔街日报语料库和 Switchboard 语料库（Switchboard Corpus）中使用的单词。CMUdict发音词典也是为自动语音识别（ASR）的目的而开发的，包含大约100 000个词形的发音。CELEX发音词典 (Celex, 1993)包括《牛津高级英语学习词典》(1974)（41 000个原形词）和《朗文现代英语词典》(1978)（53 000个原形词）的全部单词，总共包含160595个词形的发音。CELEX 发音词典是英国英语发音，其他两部发音词典是美国英语发音。每部发音词典所用的音子集(phone set)不一样，CMU和PRONLEX的音子集是从ARPAbet推出来的，CELEX的音子集是从IPA推出来的。三部发音词典都把重音表示为三层：主重音、次重音和无重音。图4.20说明了 *armadillo* 这个词在三部发音词典中的发音情况。

词典	发音	IPA标音
Pronlex	+arm.xd'Il.o	[ˌɑrmə'dɪlou]
CMU	AA2 R M AH0 D IH1 L OW0	[ˌɑrmʌ'dɪlo̲u̲]
CELEX	"#-m@-'dI-l5	[ɑ:.mə.'dɪ.ləu]

图 4.20　在三部词典中单词 *armadillo* 的发音。我们不解释这些特殊的发音符号，对于每种发音都给出了等价的IPA音标。在CMU发音词典中，对于非重读元音（[ə], [ɨ]等），用0级重音来表示。在IPA标音中，在元音下面加下划线来表示。注意，在英国英语的CELEX的发音中r-脱落的现象以及元音使用[əu]而不使用[ou]的表示方法

经常会出现两个不同的单词，其拼写形式相同而发音不同（它们称为同形词[homograph]）。例如，动词*wind*（"You need to wind this up more neatly"）的发音为[waɪnd]，而名词*wind*（"blow, blow, thou winter wind"）的发音为[wɪnd]。这个问题在 TTS 的应用中是非常关键的（因为在一定的上下文中，系统必须判断究竟应该读这两个不同的发音中的哪一个），不过，在当前的语音识别系统中，由于某些原因一般会忽略同形词的问题。书面印刷的发音词典对于不同的词类都要给出不同的发音，CELEX 在这方面做得很好，因为 CELEX 的发音词典是为 ASR 设计的。在 Pronlex 和 CMU 的发音词典中，对于 *wind* 给出了两种不同的发音，但没有说明什么词类用什么样的发音。

发音词典中通常没有收录很多专有名词。在很多实际应用中这是一个很严重的问题。Liberman and Church(1992)曾经报告过，在3300万词的 AP 新闻语料库中，有 21% 的词是专有名词。接着，Liberman 和 Church 公布了一个专有名词的词表，包含 1987 年从 Donnelly 市场组织收集的 150 万个专有名词（覆盖了美国的 7200 万个家庭）。但是，在 CELEX（它是根据传统词典编纂的）发音词典的 52 477 个原形词中，只有 1000 个专有名词。与之对比的是 Pronlex，Pronlex 包含两万个专有名词，这只是150万个词中的一小部分。只有很少的词典给出 *Dr.* 这样的词条的发音，Liberman and Church(1992)指出，*Dr.* 的发音可以为 doctor 或 drive，2/3 的发音可以为 two thirds，February third 或 two slash three。

现有的发音词典还没有很好的模型来处理虚词（*and*, *I*, *a* 和 *of* 等）的发音。这是因为这些虚词的发音变化的语音上下文环境很复杂。一般在发音词典中只包含某些最基本的形式（例如 *the* 的发音为[ðiˌ]），再使用其他算法根据上下文推出它的发音变体。第5章中将讨论如何建立这一类的词的上下文发音变体模型的问题。

TTS 和 ASR 发音词典的一个最重要的区别在于，TTS 的发音词典不需要表示方言的变体；例如，在非常精确的 ASR 的发音词典中，需要表示 *either* 和 *tomato* 的不同发音，而在 TTS 的发音词典中，只要选择不同发音中的一个发音就可以了。

4.6.2　词典之外的查找：文本分析

从文本到语音的映射依赖于前面讨论过的发音词典。前面说过，从文本到语音映射的一个办法是在发音词典中查找每个词，并读出词典之外的语音符号串。如果事先把每个词都放到发音词典中，这个方法是很奏效的。但是，正如在第3章中看到的，在英语中（或者在任何其他语言中），我们不可能把所有的词都事先表示出来。在语音合成和语音识别中，都需要猜测某些在发音词典中没有的单词的发音。本节首先研究哪些类别的单词在发音词典中容易被忽略，然后讨论怎样使用第3章中的有限状态转录机模拟文本语音转换的基本问题。第5章将介绍发音变体以及模拟它们的概率技术。

有三种重要的语言问题是不能依靠词典中的词来处理的，这三种情况是：专有名词（name），形态的能产性（morphological productivity）和数字（number）。作为简单的例子，我们这里随便选用了1998年7月17日发表在纽约时报的一篇电影短评（561个词）。关于Vincent Gallo的电影Buffalo'66的这篇短评是Janet Maslin写的。下面是文章的开头部分：

> In Vincent Gallo's "Buffalo'66", Billy Brown (Gallo) steals a blond kewpie doll named Layla (Christina Ricci) out of her tap dancing class and browbeats her into masquerading as his wife at a dinner with his parents. Billy hectors, cajoles and tries to bribe Layla.("You can eat all the food you want. Just make me look good.") He threatens both that he will kill her and that he won't be her best friend. He bullies her outrageously but with such crazy brio and jittery persistence that Layla falls for him. Gallo's film, a deadpan original mixing pathos with bravado, works on its audience in much the same way[①].

> 在Vincent Gallo的法国电影《美洲野牛66》中，Billy Brown（法国人）从踢踏舞班上偷了一个称为Layla的金发碧眼的丘比特娃娃，恫吓她并让她以他妻子的身份参加化装舞会，并陪他的父母共进晚餐。Billy虚张声势地威吓她，甜言蜜语地哄骗她，并试图收买Layla（"你想吃什么就吃什么，只要让我看起来好就行。"）他恐吓她，说他要杀她，又说他不愿做她的好朋友。他威逼她，神气活现地疯狂，战战兢兢地坚持说，Layla爱上了他。这部法国电影把毫无表情的悲伤和虚张声势混合在一起，很多地方都用相似的手法来对付观众。

我们查两部最通用的大型在线发音词典PRONLEX和CELEX。PRONLEX发音词典包含90 694个词形的发音，这些词形可覆盖多年的华尔街杂志和Switchboard语料库。CELEX大型发音词典包含160 595个词形的发音。把两部词典结合起来，总共有大约194 000个词形的发音。在这个电影短评的561个词中，有16个（3%）词的发音在这两部发音词典中是没有的（没有计算带连字符的词，如 *baby-blue* 和 *hollow-eyed*）。这些词列举如下：

专有名词		有屈折变化的名词	数字	其他
Aki	Gazzara	Gallo's	'66	c'mere
Anjelica	Kaurismaki			indie
Arquette	Kusturica			kewpie
Buscemi	Layla			sexpot
Gallo	Rosanna			

某些查不到的词可以使用增加词典容量的办法来解决，例如，在Well's(1990)的准确发音词典中（非在线的），就可以查到 *sexpot* 和 *kewpie*。但是，其他查不到的单词就需要联机进行在线生成。

---

① 这段英文是作者要分析的材料，故原文照录。为便于读者理解，下面给出了汉语译文。——译者注

专有名词是发音词典中的一个大问题；词典中不可能事先列举出英语中的一切专有名词；专有名词还可能来自其他任何语言，而且还可能有不同的拼写方法。TTS 和 ASR 的很多应用都是与专有名词分不开的；例如，在与电话有关的应用中，电话簿和打电话都离不开人名和地名。在很多应用系统中都已经注意到专有名词，并且不断地把专有名词（如 CoComp, Intel, Cisco）导入系统中。把语音技术应用于医学（例如，转写医生和病人的对话）要求处理药品名称的发音。现在已经有一些脱机的医学发音词典，但它们都很不精确(Markey and Ward, 1997)。前面说过，有 150 万个专有名词，Liberman and Church(1992)发现，在 3300 万词的 1988 AP 新闻语料库中，有 21% 是专有名词。

与英语不同，形态分析是其他很多语言的特殊问题。对于那些形态能产性很高的语言，要表示它们的每个可能的词，在计算上几乎是不可能的。例如，前面我们提到的土耳其语的例子：

uygar *+laş* *+tır* *+ama* *+dık* *+lar* *+ımız*
civilized +BEC +CAUS +NEGABLE +PPART +PL +P1PL
+dan *+mış* *+sınız* *+casına*
+ABL +PAST +2PL +AsIf (4.11)
"(behaving) as if you are among those whom we could not
civilize/cause to become civilized"

这个土耳其语单词的英语解释是："(behaving) as if you are among those whom we could not civilize/cause to become civilized"。它的中文意思是："举止不文明"。

即使像德语这样的与英语很相近的语言，创造新词的能力也比英语强得多；Sproat et al.(1998)曾举出德语中创造新词的一个例子：*Unerfindlichkeitsunterstellung*（allegation of incomprehensibility，"不可思议的主张"）。

尽管英语的形态比较简单，但是在语音模型的建模中形态知识仍然非常必要。例如，专有名词和首字母缩写词（acronyms）经常会有屈折变化（*Gallo's, IBM's, DATs, Syntex's*），就像新词会有屈折变化一样（*faxes, indies*）。另外，我们还不能简单地把 s 加在非屈折形式的发音中，因为在前一节中说过，英语中物主代词后缀-*'s* 和复数后缀-*s* 在不同的上下文中的发音是不同的；*Syntex's* 的发音为[sɪntɛksɪz]，*faxes* 的发音为[fæksɪz]，*IBM 's* 的发音为[aɪbijɛmz]，*DATs* 的发音为[dæts]。

最后，数字的发音是一个特别复杂的问题。在 *'66 in Buffalo* 中，*66* 的发音应该是[sɪkstɪsɪks]，而不是[sɪkssɪks]。电话号码 "947-2020" 的最自然的读音大概应该是 "nine" - "four" - "seven" - "twenty" - "twenty"，而不是 "nine" - "four" - "seven" - "two" - "zero" - "two" - "zero"。Liberman and Church(1992)指出，数字串的读音主要有五种方法：

- **顺序式读法（Serial）** 每个数字单独读音。例如，*8765* 的读音为 "eight seven six five"。
- **组合式读法（Combined）** 数字串按照一个整数来读音，每个数字根据它所在的位置分别加读 "thousand、hundred" 等进位数。例如，*8765* 的读音为 "eight thousand seven hundred sixty five"。
- **双对式读法（Paired）** 数字一对一对地按一个整数来读音；如果数字有奇数个，则第一个数字单独读音。例如，*8765* 的读音为 "eighty-seven sixty-five"。
- **百位式读法（Hundreds）** 四位的数字串可以按百位记数方式来读音。例如，*8765* 的读音为 "eighty-seven hundred (and) sixty-five"。
- **跟踪单位读法（Trailing Unit）** 以若干个零为结尾的数字，非零的数字部分按顺序式读法来读音，零的部分按适当的进位制来读音。例如，8 765 000 的读音为 "eight seven six five thousand"。

读者可进一步做些练习来理解这五种读音方法。

4.6.3　基于有限状态转录机（FST）的发音词典

文本－语音转换系统的发音模拟的早期工作［例如，Allen et al.(1987)的基础 MITalk 系统］主要依靠字母－语音转换规则。每条规则描述字母或字母组合是如何映射为语音的。下面是 Witten (1982)规则库中的一个片断：

字母片断	读音
-p-	[p]
-ph-	[f]
-phe-	[fi]
-phes-	[fiz]
-place-	[pleɪs]
-placi-	[pleɪsi]
-plement-	[plɪmɛnt]

这个系统有一个很长的表来记录这样的规则，并且有一部很小的词典来记录例外（这些例外常常是虚词，如 *a*, *are*, *as*, *both*, *do* 和 *does* 等）。最新的一些系统依赖于很大的词典，完全转向了算法，字母－语音转换规则只用于那些在词典中没有的单词或单词的形态变体。如何表示这些大词典才能使它们能够适应形态能产性的要求呢？非常幸运的是，发音中的这些形态学问题（例如，增加一个屈折后缀，在两个语素的连接处出现微小的发音变化等）与我们在第3章中讨论过的拼写中的形态学问题是完全等同的。因此，Sproat(1998b)在文本－语音转换中使用了转录机，我们可以把所用的转录机方法分解为五部分：

1. 一个表示词典中的每个单词和语素发音的有限状态转录机 FST；
2. 表示可能的语素序列的有限状态自动机 FSA；
3. 表示每个发音规则的单独的有限状态转录机 FST（例如，表示在不同上下文中 *-s* 的读音规则）；
4. 用于模拟专有名词和首字母缩写词的试探性规则和转录机以及字母－语音转换（letter-to-sound，LTS）规则和转录机；
5. 具有默认值的文字－语音转换的规则和转录机，以处理其他任何未知词。

这里我们仅限于讨论前面四个部分；这些有趣的字母–语音转换规则可参见(Allen et al., 1987)。前面这些部分的运行情况是第3章中讲过的有限状态转录机FST的简单扩充。第一部分用于表示每个单词的基本形式。我们知道，所谓基本形式就是单词未经屈折变化的形式。这些基本形式是按照它们符合正词法的形式存储在词典中的；我们将根据正确的词汇音位表示方法来扩充这些基本形式。图 4.21 说明了原来的条目和更新后的条目。

我们的 FST 的第二部分是模拟单词形态变化的有限状态机。这里只举一个例子：名词的复数后缀-*s*。第3章中讲过的处理英语复数的自动机加以修改之后就可以用于处理语音，如图4.22所示。惟一的修改就是增加了[s]作为后缀的发音规则以及表示所有形态特征的 ε 发音规则。

我们可以把图 4.22 中的处理屈折变化的有限状态自动机 FSA 和图 4.21 中处理词的基本形式词典的转录机结合起来，产生一部增加了处理屈折功能的、既包括名词单数也包括名词复数的词典。这样形成的一个微型词典（mini-lexicon），如图 4.23 所示。

图 4.23 中的微型词典有两个层：底层（或词汇层）和中间层。剩下的惟一事情就是增加转录机，应用拼写规则和发音规则把中间层映射到表层。拼写规则和发音规则在前面已经讲过。

符合正词法的词典	词典
规则名词	
cat	c|k a|æ t|t
fox	f|f o|ɑ x|ks
dog	d|d o|ɑ g|g
不规则单数名词	
goose	g|g oo|u s|s e|ɛ
不规则复数名词	
g o:e o:e s e	g|g oo|u:ee|i s|s e|ɛ

图 4.21 基于FST的词典，它是第3章中讲过的那个表的扩充。词典中的每个符号都表示为符号对，中间用“|”隔开：一侧表示符合正词法规则的词典条目，另一侧表示音位的词典条目。不规则的复数名词 *geese* 也说明了它在中间带子上的表示方法

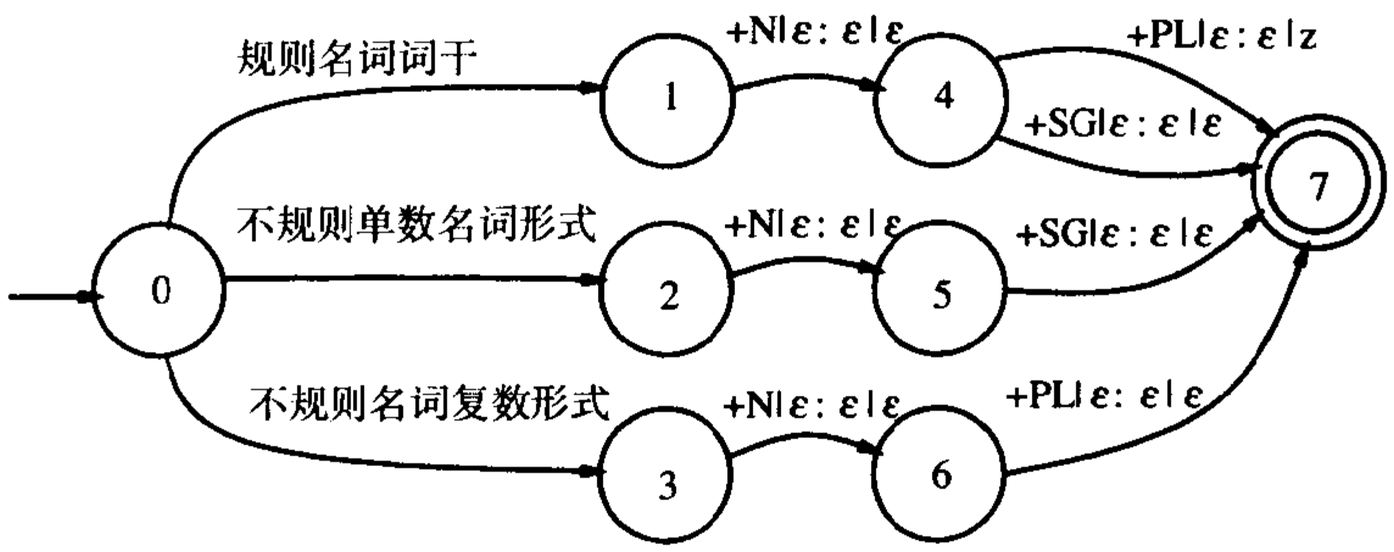


图 4.22 用于处理名词单数和复数屈折变化的有限状态转录机FST。这个自动机把形态特征[+N],[+PL]和[+SG]加到有关的词汇层上，并且把复数后缀*s*|z加到中间层上。下面将讨论为什么要把 *-s* 的发音表示为 z 而不表示为 s

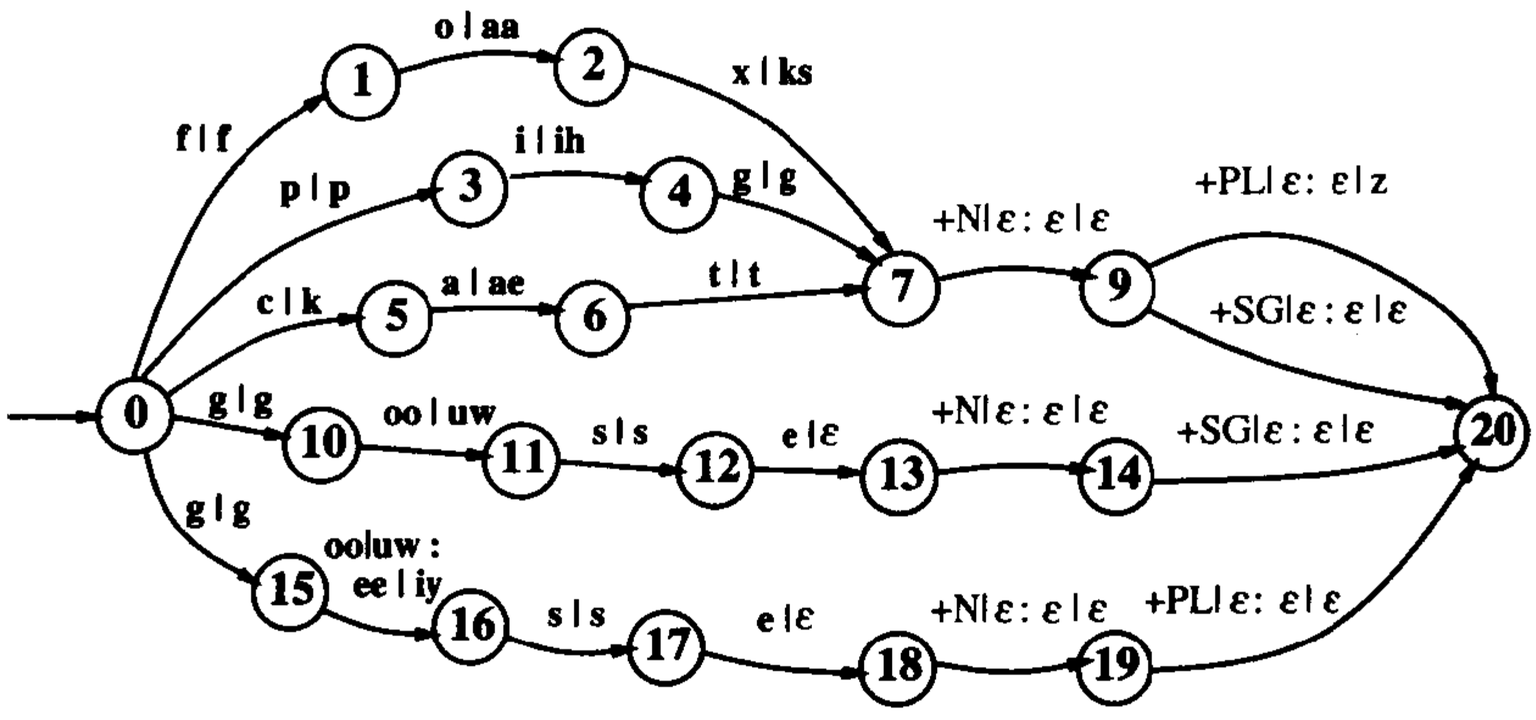


图 4.23 把图 4.21 中的处理基本形式词典的转录机与图 4.22 中的处理屈折变化的转录机结合起来的微型词典

现在可以利用第3章中讲过的词典、音位规则和正词法规则来进行词汇表示（包括符合正词法的符号串和音位符号串）与表层表示（包括符合正词法的符号串和音位符号串）之间的映射。正如在第3章中所述，这样的映射能够从表层到词汇层，也能够从词汇层到表层。图4.24说明了这样的结构方式。词典FST在词汇层（包括单词的词干及其形态特征）和中间层（表示语素的简单毗连）之间进行映射。每个FST或者表示一个简单的拼写规则限制，或者表示一个简单的音位限制，这些FST并行地运行，在中间层和表层之间进行映射。每层都有正词法表示和音位表示。在输入为词汇形式的文本－语音转换的应用中（例如，文本生成时，系统知道单词的词汇形式、它的词类、它的屈折变化等），层叠式的FST能够从词汇形式映射到表层的发音。在输入为表层拼写的文本－语音

转换的应用中（例如，文本朗读系统），层叠式的FST能够从表层的符合正词法的形式通过底层词汇形式映射到表层的发音。

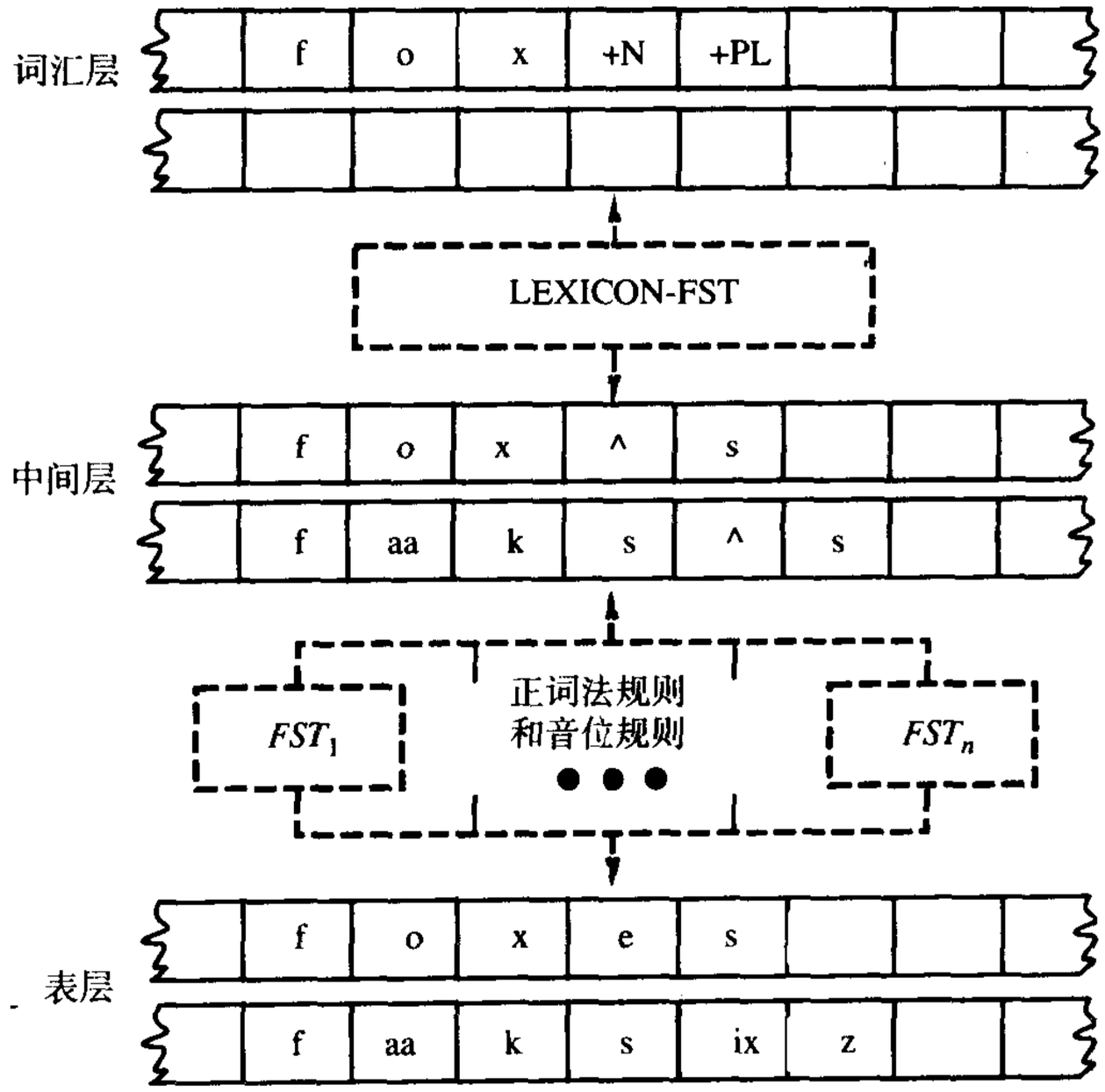


图4.24 同时根据正词法规则和音位规则在词汇形式和表层形式之间进行映射。这个系统可以用于把词汇条目映射到它的表层发音，或通过词汇条目从它的表层正词法形式映射到它的表层发音

最后，我们来谈一下关于专有名词和首字母缩写词的问题。首字母缩写词有带分割圆点的（如*I.R.S.*），也有不带分割圆点的（如*IRS*）。带分割圆点的缩写词通常按字母逐个拼读（[aɪɑrɛs]）。首字母缩写词出现时通常不带分割圆点（如AIDS，ANSI和ASCAP），或者按字母逐个拼读，或者按一个单词来发音；所以，AIDS通常的读音与动词*aid*的第三人称形式相同。Liberman and Church (1992)建议编写一本首字母缩写词小词典，收录那些按单词来读音的首字母缩写词，其他首字母缩写词都按字母逐个拼读。根据这样的方法编写了一部包含50 000个专有名词发音的词典，然后应用少量的词缀剥离规则（大致相当于第3章中的Porter Stemmer）、韵基试探（rhyming heuristics）规则、字母－语音转换规则来增加词典的覆盖面。Liberman and Church(1992)从Donnelly词表中选取出现频度最高的25万个专有名词，发现包含50 000个单词的词典就可以覆盖这25万个专有名词的59%。再增加诸如*-s*, *-ville*和*-son*等中性重音后缀（*Walters* = *Walter* + *s*, *Abelson* = *Abel* + *son*, *Lucasville* = *Lucas* + *ville*），就可以把覆盖率提高到84%。再增加“名称＋名称”的复合专有名词（例如*Abdulhussein*和*Baumgaertner*）和韵基试探，就可以把覆盖率提高到89%。韵基试探利用处于词首的字母－语音转换规则来发现韵基词，以帮助确定词尾部分的发音。例如，在发Plotsky的音时，利用*Pl-*的LTS规则并从*Trotsky*的发音来猜测*otsky*的发音。然后，再增加一定数量的更复杂的形态规则（例如O'Brien的前缀的发音规则）、重音变化后缀（*Adamovich*）、后缀变换（*Bierstadt* = *Bierbaum* - *baum* + *stadt*），并且对剩下的部分使用字母－语音转换规则系统来确定它们的发音。这个系统不是作为FST实现的。作为练习，读者可以把这一类规则改写成一个FST。对于专有名词、首字母缩写词和其他未知词的细节问题有兴趣的读者可参阅Liberman and Church(1992)，Vitale (1991)和Allen et al.(1987)。

4.7 文本－语音转换中的韵律

从符合正词法的文本到语音的转录过程只是描述了在实际生成语音的文本-语音转换系统TTS中输入的主要部分的产生过程。输入的另一个重要部分是韵律（prosody）。韵律这个术语一般用于表示句子发音中没有用从词典中推导出的语音序列来描写的那些方面。韵律是在比语音更长的语言单位上起作用的，因此，韵律的研究有时又称为超音段现象（suprasegmental phenomena）的研究。

4.7.1 韵律的音系学性质

韵律有三个主要的音系学性质：突显度（prominence）、结构（structure）和调（tune）。

前面曾经讨论过突显度，突显度是用于覆盖重音和重读的一个广义的术语。突显度是音节的一个性质，而且它的描述通常都是相对的，只是说明一个音节比另一个音节显得更突出。发音词典要标出词重音，例如table的重音在第一个音节，而*machine*的重音在第二个音节。诸如*there*, *the*和*a*这样的虚词，通常是完全不重读的。当若干个词结合在一起时，它们的重音模式也会结合起来，在更大的组合体中形成一个更大的重音模型。重音结合时要遵从一些规则。例如，诸如*new truck*（新卡车）这样的形容词-名词组合，其重音在右侧的词（*new *truck*），而诸如**tree surgeon*（树木医生）这样的名词－名词组合，其重音则在左侧。不过，一般地说，这样的规则都有例外，所以重音的预测就是一个非常复杂的问题。例如，名词-名词组合**apple cake*（苹果蛋糕）的重音在第一个单词，而名词-名词组合*apple *pie*（苹果馅饼）的重音则在第二个单词(Liberman and Sproat, 1992; Sproat, 1994, 1998a)。还有，韵基（rhythm）的作用可以使重读音节稍微有所扩展，例如，*city *hall*（市政厅）和**parking lot*（停车区）结合成**city hall* **parking lot*(Liberman and Prince, 1977)。最后，重音的位置受到话语因素的强烈影响，例如，新词语或做焦点的词语经常重读，我们将在第18章和第19章中讲解这个问题。

句子的韵律结构是指某些词似乎自然地结合在一起，而某些词似乎有明显的间隔或者彼此分开。通常用韵律短语（prosodic phrasing）来描述韵律结构。具有同样的韵律短语结构的一段话语应该具有同样的句法结构。例如，句子"*I wanted to go to London, but could only get tickets for France*"似乎包含两个主要的韵律短语，它们的边界就在逗号处。这些较大的韵律单位通常使用的术语有：语调短语（intonational phrase或IP）(Beckman and Pierrehumbert, 1986)、语调单位（intonation unit）(Du Bois et al., 1983)、调单位（tone unit）(Crystal, 1969)。另外，在第一个短语中，似乎还有更小的韵律短语边界[通常称为中间短语（intermediate phrase）]分割单词："*I wanted* | *to go* | *to London*"。韵律短语和次短语的精确定义以及它们与诸如子句和名词短语之类的句法短语及语义短语之间的关系，过去一直是而且现在仍然是很多争论的主要内容(Chomsky and Halle, 1968; Langendoen, 1975; Streeter, 1978; Hirschberg and Pierrehumbert, 1986; Selkirk, 1986; Nespor and Vogel, 1986; Croft, 1995; Ladd, 1996; Ford and Thompson, 1996; Ford et al., 1996)。尽管这个问题十分复杂，学者们还是提出了一些算法，试图把输入文本的句子自动分割为一些语调短语。例如，Wang and Hirschberg(1992)，Ostendorf and Veilleux(1994)，Tayler and Black(1998)以及其他一些学者曾经根据周围单词的词类特征、在当前单词和下一个单词中语段的长度特征、从语段的开始算起或从语段的结尾算起的潜在边界的距离特征，以及周围的单词是否有重音等特征，建立统计模型（导入概率预测参数，例如第5章中定义的CART-风格判定树等）来预测语调短语的边界。

具有同样的突显度和短语模型的两段话语可能由于具有不同的调（tune）而在韵律上有所不同。调就是话语的语调节律。我们来考虑语段"*oh, really*"。不用改变短语和重音，只要改变语调，这

个语段也会有很多变体。例如，我们可以激动地说“*oh, really*!”（当有人告诉你中了彩票，你激动地做出的回应）；也可以怀疑地说“*oh, really*?”（当你不相信说话人所说的是真的时），也可以愤怒地说“*oh, really*!”来表达你的不悦。语调可以分解成一些组成部分，其中最重要的组成部分是音高重音（pitch accent）。音高重音出现在重读音节中，形成F0曲拱的一个特殊模式（请看下面的解释）。根据模式的类型，可以产生不同的效应。音高重音分类的最流行模式是Pierrehumbert模式或ToBI模式(Pierrehumbert, 1980; Silverman et al., 1992)。这个模式指出，英语中共有5种音高重音，是由高调**H**和低调**L**两个简单的调按不同的方式组合而成的。**H+L**模式形成降调，**L+H**模式形成升调。星号（*）用于表示在重音音节上哪个调是降调。这样，可形成的音高重音模式有**H***，**L***，**L+H***，**L*+H**，**H+L***（第6个音高重音模式**H*+L**在该模式的早期版本中出现过，但后来被放弃了）。关于所举的“oh, really”的3个例子，其重音可以分别标为**L+H***，**L*+H**和**L***。除了音高重音之外，该模式还有两个短语重音**L-**和**H-**，两个边界调**L%**和**H%**，用于短语的结尾以控制语调的升或降。

其他语调模型与ToBI模型的不同之处在于，它们不使用离散的音位类别来表示语调重音。例如，Tilt(Taylor, 2000)和Fujisaki模型(Fujisaki and Ohno, 1997)使用连续的参数而不使用离散的范畴来模拟音高重音。这些研究者证明，离散模型通常比较直观，便于掌握，而连续模型则可能具有更高的鲁棒性和精确性，更便于计算使用。

4.7.2　韵律的语音和声学性质

上述三个音位因素相互作用，并在各种不同的语音和声学现象中实现。突显的音节一般比非突显的音节读得重一些、长一些。韵律的短语边界通常有停顿，边界之前的音节变长，有时边界处的音高变低。语调则表现为基频（F0）曲拱。

4.7.3　语音合成中的韵律

TTS的主要任务是生成韵律的适当语言表示，并且从这样的语言表示出发，生成适当的声学模式，而这样的声学模式将表现为输出语音的波形。这样一个韵律成分在TTS系统中的输出就是音子的一个序列，每个音子都有一个音延（duration）值和一个音高（pitch）值。每个音子的音延与语音上下文有关（见第7章）。F0的值受到前面讨论过的各种因素的影响，包括词重音、句子的重读或焦点成分以及话语的语调（例如，疑问句中后面部分的语调要升高）。图4.25是FESTIVAL(Black et al., 1999)语音合成系统对于句子 *Do you really want to see all of it*?的TTS输出的一个样本。这个输出以及图4.26所示的F0的值成了第7章中所描述的波形合成（waveform synthesis）的输入。其中的音延是用CART风格判定树(Riley, 1992)来计算的。

				H*												L*		L- H%			
do		you		really				want				to		see		all		of		it	
d	uw	y	uw	r	ih	l	iy	w	aa	n	t	t	ax	s	iy	ao	l	ah	v	ih	t
110	110	50	50	75	64	57	82	57	50	72	41	43	47	54	130	76	90	44	62	46	220

图4.25　句子 *Do you really want to see all of it*?在FESTIVAL语音合成器(Black et al., 1999)中的输出。精确的语调曲拱如图4.26所示。图片由Paul Taylor提供

如上所述，确定一个句子的韵律模式是很困难的，因为需要有真实世界的知识和语义学的信息来判别要重读什么音节，要应用什么语调。此类信息很难从文本中抽取出来，因此韵律模式通常只

是产生输入文本的“中性的陈述句”，并且假定说这样的句子时，不需要参照话语的历史或现实世界的事件，它是一个默认值。这是在TTS中语调总是显得有些“呆板”的一个主要原因。

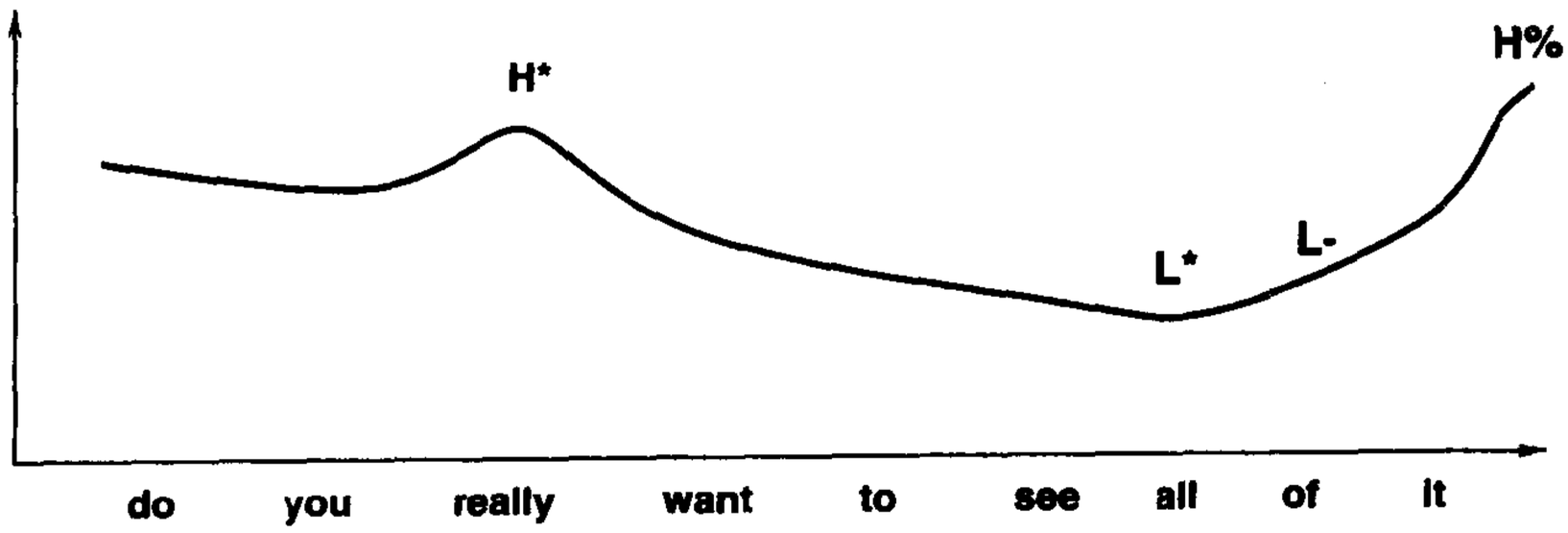


图4.26 FESTIVAL语音合成系统生成的图4.25中的示例句子的F0曲拱。图片由Paul Taylor提供

4.8 人处理音位和形态的过程

在第3章中我们曾经提过，在人的词表中能产性的形态规则起着心理现实的作用，但未详细展开讨论就转去研究如何建立表示形态规则的模型问题。现在，我们研究了音位结构和音位学习，再回过头来讨论形态知识和音位知识表示的心理学问题。

关于人的形态处理和音位处理的一种观点是，把能产性的、规则的形态与不规则的、有例外的形态区分开来。根据这样的观点，例如，规则的过去时态语素-ed可以在人的思想里表示为一条规则，这条规则还可以应用于像*walk*这样的动词，产生出它的过去时态*walked*。而像*broke*，*sang*或*brought*这样的不规则过去时态动词，则要作为词汇表示的一个部分逐个存储起来，对它们不能使用规则。在这种情况下，这样的看法严格地把使用规则的表示与使用词汇列表的表示区分开来。

这样的看法似乎是明智的，它与前两章介绍过的基于转录机的模式是等同的。可是，这样的模式似乎是错的。第一个问题是，不规则动词本身在音位上明显地具有亚规则性（subregularity）。例如，联系*ring*和*rang*的ɪ/æ交替,也出现在*sing*和*sang*的联系和*swim*和*swam*的联系中(Bybee and Slobin, 1982)。儿童在学习语言的过程中经常会把这个模式不正确地扩充到其他词,产生*bring-brang*这样的错误联系。成年人经常会犯发音的错误，也是这种“亚规则性”影响的结果。第二个问题是，心理学已经证明，高频度的屈折规则形式（*needed*和*covered*）也像词干（*need*和*cover*）一样存储在词表中(Losiewicz, 1992)。最后一个问题是，在人的信息处理中，词的频度和语素的频度一般都起着很重要的作用。

诸如此类的论据导致了形态学习和形态表示的“数据驱动”（data-driven）模型,也就是说，所有的屈折形式都原封不动地存储在词表中。这个模型可以更一般地通过类推表示为新的形式：规则的形态变化与亚规则的形态变化都是一样的，规则的形态变化外部看来貌似规则，只不过是由于它们发生得比较频繁而已。这样的模型包括Rumelhart and McClelland(1986)的计算连接主义（connectionist）或并行分布式处理模型（Parallel Distributed Processing），之后对这个模型的各种改进(Plunkett and Marchman, 1991; MacWhinney and Leinbach, 1991)以及Bybee(1985, 1995)的相似网络模型（network）。在这些模型中，像*-ed*这样的规则语素的行为由于它与其他形式的频繁交互而暴露出来了。倡导形态变化基于规则的研究，如Pinker and Prince(1988)，Marcus et al.(1995)以及其他一些研究，都批评连接主义模型，提出了妥协性的双处理模型（dual processing），认为像-ed这样的规则形式可表示为符号规则，而亚规则的例子（*broke*, *brought*）则可以表示为具有连接主义风格的模式联想机制。连接主义和双处理模型之间的争论对于各种正规的基于规则的行为的心理解

释有着深刻的意义，这种争论是人类语言处理的一个最有意思的公开问题。第7章将简短地讨论人类语言处理的连接主义模型，对于连接主义有进一步探讨兴趣的读者可参阅 Anderson(1995)。

4.9　小结

本章介绍了理解口语处理时需要的一些重要概念。本章要点如下：

- 我们可以用称为音子（phone）的单位来表示词的发音。表示音子的标准系统是国际音标或IPA。使用 ASCII 字符的另一个只适用于英语的转写系统是 ARPAbet。
- 根据发音器官如何产生音子可以对音子进行描述，根据辅音的发音部位和发音方法以及浊音化（voicing）的情况可以对辅音进行描述，根据舌位的高低和舌位的前后则可以对元音进行描述。
- 音位是对不同语音情况的一般化和抽象。音位变体规则描述一个音位在给定的上下文环境中的实现情况。
- 转录机可以用来模拟音位规则，其方法与在第3章中模拟拼写规则的方法一样。双层形态学是一种形态学/音系学的理论。这种理论把形态规则模拟为对于词汇形式和表层形式之间映射的一种有限状态的、良构的限制（well-formedness constraint）。
- 发音词典既可以用于文本-语音转换，也可以用于语音自动识别。发音词典要给出单词的发音，这种发音是音子的序列，有时也包括音节划分（syllabification）和重音。大多数联机发音词典的规模为10万词左右，但是还缺少专有名词、首字母缩写词和屈折变化形式。
- 文本－语音转换系统中的文本分析（text-analysis）部分把符合正词法的文字串映射为音子串。这通常要使用大型的发音词典来进行。大型的发音词典是在发音词典的基础上提升而成的，它应该带有一个用于处理能产性的形态规则、发音变化、专有名词、数字以及首字母缩写词的系统（这样的系统犹如一个转录机）。

4.10　文献和历史说明

发音语音学的主要研究工作可以追溯到公元前800年至150年的印度语言学家。他们提出了发音部位和发音方法的概念，揭示了区分浊音和清音的声门机制，探索了语音同化的概念。直到2000多年以后，也就是19世纪后期，欧洲的语音学家在语音科学研究方面还没有赶上印度的语音学家。希腊人只具有一些最起码的语音学知识，例如，在柏拉图的 Theaetetus 和 Cratylus 时代，他们能区分元音和辅音，停顿辅音和连续辅音。斯多葛学派发展了关于音节的思想，认识到了语音对于可能单词的制约关系。12世纪的一个不知名的冰岛学者探索了音位的概念，提出了冰岛语的一个书写系统，其中包括用于表示音长和鼻音化的变音符号。但是他的文章直到1818年还没有发表，以至于到现在，在斯堪的那维亚半岛之外的大多数学者对此事还一无所知（Robins, 1967)。大家普遍认为现代语音学是从 Sweet 开始的。他在 *Handbook of phonetics*(1877)一书中，已经从本质上说明了什么是音位。他还设计了音标字母，区分了宽式音标和严式音标，他提出的很多建议现在都纳入了IPA。在 Sweet 的那个时代，我们可以说 Sweet 是最优秀的语音学实践家；他第一次把语言进行了科学的录音，以便于语音学的研究，他还推进了语音发音描写的技术。在戏剧家肖伯纳（George Bernard Shaw）的剧本中，有一个称为 Henry Higgins 的角色就是按照 Sweet 的形象塑造的，可是在戏剧家笔下他却被描写成一个性格古怪、难于接近并且名声不太好的家伙。“音位”（phoneme）这个术语是在波兰学者 Baudouin de Courtenay 于1894年提出的理论中首先命名的。

Johnson(1972)提出了用正则关系来模拟音位规则的思想，他说明了不允许规则用于其本身输出的任何音位系统（也就是没有递归规则的音位系统）都能用正则关系（或有限状态转录机）来模拟。从本质上讲，至今所有被形式化的音位规则都具有这样的性质（除了某些具有诸如前重音和语调规则的整合值特征的规则之外）。可惜 Johnson 的思想没有引起学术界的注意，而在后来被 Roland Kaplan 和 Martin Kay 独立地发现了。关于双层形态学的其他历史问题可参见第3章。Karttunen(1993)写了一个关于双层形态学的教材性质的导论，其中包括的一些最新研究进展和详细说明资料是我们在这里没有介绍的。

对于音系学有兴趣的读者可以阅读Goldsmith(1995)，此书是音系学理论的通论性的参考；关于优选理论可参阅 Archangeli and Langendoen(1997)。

有两个经典的文本-语音合成系统分别在 Allen et al.(1987)（MITalk 系统）和 Sproat(1998b)（Bell Labs 系统）中做了介绍。在文本-语音合成系统中的发音问题现在还是一个正在进行的研究领域；很多研究集中在韵律问题上。有兴趣的读者可以参阅主要语音工程会议的论文集，它们是：ICSLP（the International Conference on Spoken Language Processing，口语处理国际会议），IEE ICASSP（the International Conference on Acoustics, Speech, and Signal Processing，声学、语音和信号处理国际会议）和 EUROSPEECH。

对于音标和发音语音学有兴趣的读者可参阅Ladefoged(1993)，这是一本有关语音学的导论性教材。Pullum and Ladusaw(1996)是关于 IPA 的所有符号和变音符的非常全面的指导读物。与计算有关的很多语音学文章可以在*Journal of the Acoustical Society of America*（美国声学学会杂志）和 *Computer Speech and Language*（计算语音和语言）以及 *Speech Communication*（语音通信）中找到。

第5章　发音与拼写的概率模型

ALGERNON: *But my own sweet Cecily, I have never written you any letters.*
CECILY: *You need hardly remind me of that, Ernest. I remember only too well that I was forced to write your letters for you. I wrote always three times a week, and sometimes oftener.*
ALGERNON: *Oh, do let me read them, Cecily?*
CECILY: *Oh, I couldn't possibly. They would make you far too conceited. The three you wrote me after I had broken off the engagement are so beautiful, and so badly spelled, that even now I can hardly read them without crying a little.*

Oscar Wilde, *The Importance of being Ernest*①

就像 Oscar Wilde 寓言中的那个 Cecily 一样，在 19 世纪和 20 世纪之交，很多人都在考虑关于拼写的问题。Gilbert 和 Sullivan 提供了很多例子。例如，*Gondoliers' Giuseppe* 担心，如果 *Iolanthe's* Phyllis 把他的私人秘书“所用的每个词都拼写出来”，那么，他的私人秘书“将会在他的拼写面前吓得发抖”。Thorstein Veblen 在 1899 年的经典著作 *The Theory of the Leisure Class*（有闲阶级的理论）中曾经解释道，“深奥难懂的，累赘的，低效率的”英语拼写系统的主要目的在于提供一种十分困难的手段来验证有闲阶级的身份。无论拼写的社会作用怎样，我们认为大多数人都会喜欢 Cecily，而不会喜欢 Phyllis。据估计，人们键入文本的拼写错误大约在 0.05%（对于仔细编辑的新闻文本）到 38%（对于电话簿查询等拼写困难的应用场合）之间变动(Kukich, 1992)。

在本章中，我们将讨论拼写错误的检查和更正问题以及与语音自动识别和文本-语音转换系统中模拟发音变化有关的问题。从表面上看来，发现文本中的拼写错误以及模拟口语中单词的发音变化的问题似乎没有什么共同点。但是，我们可以使用一种重要方法把这些问题转化成同构（isomorphic）问题：它们两者都可以看成概率转移（probabilistic transduction）问题。在语音识别中，给出表示在上下文中单词发音的一串符号，我们需要把这一串符号转化为表示该单词的确切的词汇发音或在词典中所标注的发音的符号串，因此，我们必须查找词典。然而，任何给定的表层发音都是有歧义的；一个表层发音对应于若干个可能的不同单词。例如，ARPAbet 发音[er]可能对应于单词 *her*, *were*, *are*, *their* 或 *your* 等单词中的某部分形式。这样的歧义问题可以概括为“发音变异”（pronunciation variation）。例如，单词 the 有时发音为 THEE，有时发音为 THUH；单词 *because* 有时的出现形式为 *because*，有时的出现形式为 *'cause*。这种变异的某些方面是有系统性的；在 5.7 节中将考察语音识别和文本-语音转换中某些重要的发音变异类别，并介绍描写发音变异的某些初步规则。高质量的语音合成算法需要知道，什么时候使用什么样的特定发音变异。要完成语音合成和

---

① 译文如下，供读者参考：
ALGERNON：不过，我亲爱的 Cecily，我还没有给你写信呢。
CECILY：你别提写信这件事了，Ernest。我只是想起当我不得不给你写信的时候，我是写得多么地工整啊。我总是每星期给你写三封信，有时还更多。
ALGERNON：哦，你能让我读一读它们吗，Cecily?
CECILY：哦，我大概不能这样做。它们也许会使你变得盲目地自以为是吧。在我解除婚约之后，你写给我的那三封信是多么地漂亮啊，可是你的拼写却糟糕透了，以至于现在如果我不悄声哭出来就难以卒读。

语音识别这两个方面的任务，要求我们使用概率变异来扩充第4章中讨论过的表层音子和词汇音子之间的转化方法。

同样地，给定对应于拼写错误的单词的字母序列，我们需要产生一个顺序表，把可能的正确单词列举出来。例如，*acress* 这个序列可能是 *actress*（女演员），或 *cress*（水芹），或 *acres*（英亩）的错误拼写。我们要把 *acress* 的表层形式转换为它的各种可能的“词汇”形式，并标上它们的相应概率，然后，我们从中选择概率最高的词为正确的词。

在本章中，我们首先介绍检查和更正拼写错误的问题，总结人们的典型拼写错误模式。然后，我们介绍用于解决拼写和发音问题的主要概率技术：贝叶斯规则和噪声信道模型。在本书的很多问题中，特别是在语音识别（speech recognition）（见第 7 章）、词类标注（part-of-speech tagging）和概率剖析（probabilistic parsing）中，贝叶斯规则及其在噪声信道模型中的应用，都起着十分重要的作用（见第 12 章）。

贝叶斯规则和噪声信道模型提供了解决这些问题的概率框架。但是，要实际地解决这些问题还需要算法。本章将介绍一种称为动态规划（dynamic programming）的主要算法，并介绍这种算法的一些实际应用，包括 Viterbi 算法（韦特比算法）、最小编辑距离算法（minimum edit distance algorithm）和向前算法（forward algorithm）。我们还要介绍有限状态自动机的一个概率版本，称为加权自动机（weighted automaton）。

5.1 关于拼写错误

拼写错误的检查和更正是现代词语处理器的一个组成部分。同样的算法对于光学字符识别（optical character recognition, OCR）和联机手写体识别（on-line handwriting recognition）也是很重要的，尽管个别字母还不能保证被精确地识别出来。“光学字符识别”是机器印刷或手写字符自动识别中所用的一个术语。在光学字符识别中，光学扫描仪把机器印刷或手写的页面转换成位图，然后再把位图传送给 OCR 算法进行处理。

联机手写体识别是在用户联机书写的同时计算机来识别人手写的字符或者曲线手写体字符。与 OCR 对于手写体的分析不同，联机手写体识别的算法可以使用在手写输入时的一些动态信息，例如字符笔画的数量和顺序，每个笔画书写时的速度和方向等。联机手写体识别在不便使用键盘的时候是很重要的，例如，在小规模的计算环境中（如Palm-Pilot掌上定位应用等），或者在中文输入中（这时，由于汉字书写符号的数目很多使得键盘难以应付）。

在本章中，我们将集中研究拼写的检查和更正，主要研究打字者键入的文本，但这样的算法也可以应用于 OCR 和手写体识别。尽管 OCR 的错误类型与打字者的错误类型不同，但 OCR 系统会比打字者的错误率更高。例如，OCR 系统经常会错误地混淆“*D*”和“*O*”。“*ri*”和“*n*”，形成错误拼写的单词，把 *derision* 判断为 *dension*，把 *PDQ Bach* 判断为 *POQ Bach*。对于手写体识别有兴趣的读者可以阅读 Tappert et al.(1990)，Hu et al.(1996)和 Casey and Lecolinet(1996)。

Kukich(1992)在关于拼写更正的调查文章中，把这个领域分解为 3 个大问题，这些问题的范围正在扩大：

1. **非词错误检查** 检查会导致非词的拼写错误，例如把 *giraffe* 拼写成 *graffe*。
2. **孤立词错误更正** 更正会导致非词的拼写错误，例如把 *graffe* 更正为 *giraffe*，但只在孤立的环境中查找这个词。
3. **依赖于上下文的错误检查和更正** 如果错误的拼写恰好是英语中真实存在的一个单词（真词错误，*real-word error*），就需要使用上下文来检查和更正这样的拼写错误。这种情况常常

来自打字操作时的错误（例如，插入、脱落或改变位置），或者由于书写者错误地拼写同音词和准同音词，并用这些词来彼此替换（例如，用*dessert*替换*desert*，或者用*piece*替换*peace*）。

下一节将讨论发生在打字文本、OCR和手写体识别输入中的拼写错误模式的类别。

5.2　拼写错误模式

人的打字文本中出现的拼写错误，在数量和性质方面，不同于在OCR和手写体识别器这样的模式识别设备中引起的拼写错误。Grudin(1983)发现，在人的打字文本中的拼写错误率在1%~3%（包括非词错误和真词错误）。这个错误率对于经过复制编辑后的文本将会明显地下降。手写体文本的拼写错误率与此相似，根据Kukich(1992)的报告，词错误率在1.5%~2.5%。

OCR和联机手写体识别系统的错误是不同的。Yaeger et al.(1998)根据研究（文中告诫说，这项研究的结果还没有确定）指出，对于Apple计算机的NEWTON MESSAGEPAD上的联机印刷体字符识别，单词的识别准确率为97%~98%，这就是说其错误率为2%~3%，不过变化幅度比较大（这依赖于对写作人的训练等）。现在还不清楚，NEWTON失败的原因究竟是由于原来对这个错误率的估计过于乐观，还是2%~3%的错误率不可接受。一些更现代的设备，例如3COM's Palm Pilot，经常使用一种特殊的输入字符（如Palm Pilot的“Graffiti”）来替代随意的手写输入。OCR的错误率，由于输入质量的不同，变化幅度也很大；Lopresti and Zhou(1997)建议，在典型情况下，OCR的字母错误率的范围应在0.2%（对于第一次复印的干净文本）到20%之间，或者更差（对于多次复印的文本或者传真文本）。

在早期的研究中，Damerau(1964)发现，在人穿孔的文本中，80%的错误是由“单个错误的错拼”（single-error misspelling）引起的。所谓“单个错误”，就是下列错误中的某一个[①]：

- **插入**（insertion）　把*the*错误地打成*ther*
- **脱落**（deletion）　把*the*错误地打成*th*
- **替代**（substitution）　把*the*错误地打成*thw*
- **换位**（transposition）　把*the*错误地打成*hte*

在这项研究的影响下，后来很多的研究工作都集中在更正“单个错误的错拼”方面。根据单个错误的错拼的大量资料，我们将在本章后面介绍第一个算法。

Kukich(1992)把人的打字错误分为两大类：打字操作错误（typographic error）和认知错误（cognitive error）。打字操作错误（例如，把*spell*错误地拼写为*speel*）一般与键盘有关。认知错误（例如，把*separate*错误地拼写为*seperate*）是由于写文章的人不知道如何拼写某个单词造成的。Grudin(1983)发现，键盘对于错误的产生有强烈的影响；打字操作错误占了所有错误类型的大多数。例如，我们来考虑替代错误，这种错误是打字新手最常见的出现频度最高的错误，又是专业打字员最常见的出现频度次高的错误。Grudin发现，键盘中同一行的相邻键（例如，*a*和*s*）的替代错误为打字新手替代错误的59%，为专业打字员替代错误的31%（例如，把*small*错误地拼写为*smsll*）。再加上同一列键的替代错误和对应错误（homologous errors，错误地用另一只手打键盘对侧相应的键而造成的错误），打字新手的替代错误的83%和专业打字员的替代错误的51%都可以看成是基于

① 根据另外的语料库的调查，Peterson(1986)发现，单个错误的错拼在所有错拼的单词中所占的百分比更高（93%~95%）。80%的单个错误的错拼和更高的错拼比例之间的差别可能是由于如下事实造成的：Damerau的文本已经包含了转写到穿孔卡片时引起的错误、键盘穿孔的错误、纸带设备引起的错误，再加上纯粹由人造成的拼写错误。

键盘的错误。认知错误包括语音错误（用语音上等价的字母序列来替代，例如用*seperate*来替代*separate*）和同音词错误（例如用*piece*来替代*peace*）。我们将在第7章讨论真词错误更正时进一步讨论同音词错误。

打字错误一般可以用上述的替代、插入、脱落和换位来描述。OCR错误一般可以分为五类：替代、多重替代、空白脱落、空白插入和识别失败。Lopresti and Zhou(1997)给出了下列常见OCR错误的例子：

正确输入的文本：

The quick brown fox jumps over the lazy dog.

识别输出文本：

'lhe q~ick brown foxjurnps ovcr tb l azy dog.

替代错误（$e \rightarrow c$）通常是由视觉的相似性造成的（而不是由键盘键位的距离造成的），多重替代（$T \rightarrow$ *'l*, $m \rightarrow$ *rn*, *he* $\rightarrow$ *b*）也是由这样的原因造成的。识别失败（用代字符号"~"表示：$u \rightarrow$ ~）则是由于OCR的算法没能精确地选到任何字符造成的。

5.3 非词错误的检查

检查文本中的非词错误，无论这个文本是打字的还是扫描的，最通常的做法是使用词典。例如，在上面的OCR错误文本中*foxjurnp*是不可能在词典中出现的。早期的一些研究(Peterson, 1986)曾经建议，这样的拼写词典的规模应该比较小，因为规模太大的词典可能包含一些罕用词，它们容易与其他词的错误拼写形式相混淆。例如，*wont*（习惯）是一个合法的罕用词，但它又是*won't*的一个常见的错误拼写形式。同样，*veery*（一种画眉鸟）也是*very*的一个常见的错误拼写形式。在单个错误的错拼简单模式的基础上，Peterson指出，在50 000个词的词典中，可能有10%的错拼被真词所"隐藏"，在350 000个词的词典中，可能有15%的错拼被真词所"隐藏"。Damerau and Mays (1989)发现，实际上并不是这样的，在大规模的词典中，某些错拼并不会被真词"隐藏"，事实证明使用大规模的词典的利大于弊。

为了表示能产性的屈折变化（后缀*-s*和*-ed*）和派生，用来做拼写错误检查的词典一般还要包括形态分析模式，正如在第3章和第4章中用于文本－语音转换的词典一样。早期的拼写错误检查系统允许任何词具有任何后缀，Unix SPELL系统允许接受一些带奇怪前缀的单词，如*misclam*, *antiundoggingly*和一些由*the*派生的后缀，如*thehood*和*theness*等。现代的拼写检查系统都使用了更多的、充分考虑语言学规则的形态表示（见第3章）。

5.4 概率模型

本节介绍发音和拼写变异中的概率模型。这些模型，特别是贝叶斯推理（Bayesian inference）和噪声信道模型（noisy channel model），同时也应用于本书中的其他很多问题。

我们在前面说过，ASR发音模型的问题和打字文本、OCR文本的拼写更正问题都可以模拟为从一个符号串到另一个符号串的映像问题。在语音识别中，对于给定的表示在上下文中某个单词的发音的一个符号串，我们需要计算并输出表示其词汇发音或词典发音的一个符号串，因此可以在词典中查找这个单词。同样地，对于给定的在拼写错误的单词中的不正确的字母序列，需要计算并输出在拼写正确的词中的一个正确的字母序列。

从直觉来看，噪声信道模型可以这样来理解：表层形式（“推导出”的发音或错拼单词）可以看成是词汇形式（“词汇”的发音或正确拼写的单词）在通过了噪声信道之后得到的一个实例。由于在信道中有“噪声”，使得我们难以辨认出词汇形式的“真实”单词的面目。我们的目的就是建立一个信道模型，使得能够计算出这个“真实”单词是如何被“噪声”改变面目的，从而恢复它的本来面目。对于完全的语音识别研究，它的噪声有很多来源：发音变异、音子实现时的变异以及来自信道的声学方面的变异（扩音器、电话网络等）。由于本章重点讨论发音的问题，因此这里的所谓“噪声”就是指发音变异，也就是那些给词汇发音或“规范”发音戴上假面具的发音变异；在语音识别系统中的其他噪声源将在第7章中讨论。对于拼写错误检查，所谓“噪声”就是那些给正确的拼写戴上假面具的拼写错误。“噪声信道”这个比喻来自20世纪70年代IBM实验室应用于语音识别的模型，这个模型称为“噪声信道模型”(Jelinek, 1976)，如图5.1所示。不过，这个模型本身却是贝叶斯推理的一个特殊情况，自从Bayes(1763)的研究工作以来，这个模型就为世人所知了。早在20世纪50年代初，贝叶斯推理和贝叶斯分类就已经在语言研究中得到了成功的应用，其中包括1959年Bledsoe在OCR方面的研究以及Mosteller and Wallace(1964)应用贝叶斯推理来确定the Federalist（联邦主义者）文章的作者分布的创新性研究。

图5.1　噪声信道模型

在贝叶斯分类中，与在其他分类中一样，当我们得到某个“观察”（observation）时，我们的任务是确定这个观察属于哪个类别的集合。对于语音识别，我们设想，现在的观察就是构成我们所听到的单词的音子串。对于拼写错误检查，观察就是构成可能的错拼单词的字母串。在这两种场合，我们都想把观察分摊到单词。在语音识别的场合，无论单词*about*的发音有多少种可能的方式（见第4章），我们只想把它分摊到*about*。在拼写检查的场合，无论单词*separate*是怎样错误拼写了，我们只想把它识别为*separate*。

让我们从发音的例子开始。我们得到一个音子串（例如，[ni]）。我们想知道，哪个单词对应于这个音子串。这个问题的贝叶斯解释要从考虑一切可能的类开始，所谓“一切可能的类”，在我们的场合，也就是“一切可能的单词”。在所有这些包罗万象的单词中，我们只想选择那些最可能给出已有的观察[ni]的单词。换句话说，在词汇V的所有单词中，我们只想使得P（单词|观察）为最大的那个单词。我们用$\hat{w}$来表示“我们对正确单词w的估计”，用$\mathbf{O}$来表示“观察序列[ni]”（我们之所以说“序列”是因为我们把每个字母看成一个观察）。因此，从其中挑选出最优单词的等式为：

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} P(w|O) \tag{5.1}$$

函数$\operatorname{argmax}_x f(x)$的意思是“使得$f(x)$为最大值的$x$”。式（5.1）能保证给我们最优的单词$w$，但是，我们不清楚怎样才能使这个等式运行起来；这就是说，对于给定的单词w和观察序列O，我们还不知道怎样才能计算出$P(w|O)$。贝叶斯分类的直觉告诉我们，如何利用贝叶斯公式，把式（5.1）转换成两个概率的乘积，而其中的每个概率都比$P(w|O)$容易计算。贝叶斯公式如式（5.2）所示；这个规则给了我们把$P(w|O)$分解成其他三个概率的方法：

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \tag{5.2}$$

把式（5.2）代入式（5.1），得到式（5.3），我们就把$P(w|O)$分解成了三个概率：

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} \frac{P(O|w)P(w)}{P(O)} \tag{5.3}$$

式（5.3）中右侧的大部分概率与原来试图求最大值的式（5.1）中的$P(w|O)$相比，更容易进行计算。例如，$P(w)$是单词的概率，可以根据单词的频度来估计。下面将会看到，$P(O|w)$也很容易估计出来。但是，观察的概率$P(O)$却很难估计。幸运的是，我们可以忽略$P(O)$。为什么？因为我们现在要对所有的词求最大值，我们将对每个词计算$\frac{P(O|w)P(w)}{P(O)}$。但是每个单词的$P(O)$是不会改变的，因为我们总是要对同样的观察O问最大可能的词串是什么，而观察都有同样的概率$P(O)$。因此，我们有：

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} \frac{P(O|w)P(w)}{P(O)} = \underset{w \in V}{\operatorname{argmax}} P(O|w)\,P(w) \tag{5.4}$$

总之，给出某个观察的具有最大概率的单词w可以用每个单词的两个概率的乘积来计算，并且选乘积最大的单词为所求的单词。这两个术语的名称如下：$P(w)$称为先验概率（Prior probability），$P(O|w)$称为似然度（likelihood）。

关键概念#3

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} \overbrace{P(O|w)}^{\text{似然度}} \overbrace{P(w)}^{\text{先验概率}} \tag{5.5}$$

下一节将说明如何对发音和拼写进行概率计算，求出它们的先验概率和似然度这两个概率。

5.5 把贝叶斯方法应用于拼写

拼写更正有很多算法，我们将重点研究贝叶斯算法，因为这种算法具有普遍性。

第6章将说明如何扩充这个算法以模拟真词拼写错误，本节只重点讨论非词拼写错误。Kernighan et al.(1990)首先提出把噪声信道方法应用于拼写更正，所用的*correct*程序把UNIX spell程序拒绝的单词生成潜在地可能正确的单词表，并按式（5.5）排序，从中挑选出序号最高的单词作为真正正确的单词。

让我们以Kernighan et al.(1990)的错拼单词acress为例，将其使用的算法走一遍。这个算法分为两步：第一步是提出候选更正表（proposing candidate correction），第二步是对候选进行打分（scoring the candidate）。

为了提出候选更正表，Kernighan等简单地假设：正确单词与错拼单词的差别只表现为插入、脱落、替代和换位这四种方式中的一种。Damerau(1964)的结果说明，尽管这个假设可能漏掉某些更正，但它可以对付在人的打字文本中的大多数拼写错误。这个候选单词表可以从错拼单词生成，而把在单词中引起的任何一个单独的转换方式应用于一个大型的联机词典就可以得到错拼单词（typo）。把所有的转换方式应用于*acress*，就可以得到图5.2中的候选单词表。

算法的第二步是使用式（5.4）来给候选更正打分。令t表示错拼单词（typo），c表示候选更正集合C上的元素c。这样，最佳的更正为

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \overbrace{P(t|c)}^{\text{似然度}} \overbrace{P(c)}^{\text{先验概率}} \tag{5.6}$$

错误	更正	转换			
		正确字母	错误字母	位置（字母#）	类型
acress	actress	t	–	2	脱落
acress	cress	–	a	0	插入
acress	caress	ca	ac	0	换位
acress	access	c	r	2	替代
acress	across	o	e	3	替代
acress	acres	–	2	5	插入
acress	acres	–	2	4	插入

图5.2　错拼词acress的候选更正表以及产生错误的转换方式(Kernighan et al., 1990)，“–”表示零字母

正如在式（5.4）中一样，我们在式（5.6）中省略了分母，因为错拼词为t，所以对于所有的c，t的概率$P(t)$是一个常数。每个更正的先验概率$P(c)$，可以根据单词c在某个语料库中出现的频繁程度次数来计算。然后，用所有单词的总出现次数来归一化（normalizing）这个频繁程度次数[①]。所以，一个更正单词c的概率就等于c的出现次数除以语料库中的词数N。零次数可能会引起问题，所以我们给所有出现次数都加0.5。这种做法称为“平滑”（smoothing），将在第6章中进行讨论。注意，在式（5.7）中不能只用单词总数N来除，因为我们对每个更正单词的出现次数都加了0.5，所以也要给词汇中的单词V加0.5。

$$P(c)=\frac{C(c)+0.5}{N+0.5V} \tag{5.7}$$

在第6章中，我们将更多地讨论语料库在计算先验概率中的作用；现在，我们使用Kernighan et al.(1990)的语料库来进行计算，这是AP新闻语料库(1988)，规模为4400万词。所以，N为4400万。在这个语料库中，单词*actress*出现了1343次，单词*acres*出现了2879次，等等。我们计算得到如下的先验概率：

c	freq(c)	p(c)
actress	1343	0.000 031 5
cress	0	0.000 000 014
caress	4	0.000 000 1
access	2280	0.000 058
across	8436	0.000 19
acres	2879	0.000 065

其中，c表示候选更正单词，freq(c)表示c的出现次数（频度），p(c)表示c的概率。

似然度($p(t|c)$)的精确计算至今还是一个没有解决的（或不可解决的）研究题目；一个单词被错误拼写的精确概率与打字者是谁、打字者是否熟悉他所用的键盘、打字者的一只手是否比另一只手更疲倦等因素都有关系。幸运的是，尽管我们不能精确地计算$p(t|c)$，但是可以相当好地来估算它，因为预示插入、脱落、换位等错拼的大多数重要因素都是一些局部性的因素，诸如正确字母本身是否等同，字母如何被错拼以及错拼时周围的上下文等。例如，字母m和n经常彼此替代而发生错拼，其部分原因是由于这两个字母的等同性（这两个字母发音相近，在键盘的位置彼此相邻），部分原因是由于上下文（这两个字母不仅发音相近，而且它们往往出现在相似的上下文中）。

Kerninghan et al.(1990)使用了一种估算这种概率的简单方法，不考虑大多数可能因素对错误概率的影响，而只是进行估算。例如，考虑在某个有错误的大语料库中e替代o的次数来估算p

① 所谓“归一化”，就是用某个总次数来除，使得其概率处于0和1之间，使之符合规格。

(acresslacross)。这可以用含混矩阵（confusion matrix）来表示，含混矩阵是26 × 26的方框表，表示一个字母被另一个字母错误替代的次数。例如，在替代含混矩阵中，标记为[*o*, *e*]的单元将给出*e*替代*o*的次数；在插入含混矩阵中，标记为[*t*, *s*]的单元将给出*t*插入到*s*后面的次数。计算含混矩阵时，需要手工收集拼写错误及其相应的正确拼写，然后计算不同错误发生的次数，Grudin (1983)就是这样做的。Kernighan et al.(1990)使用了四个含混矩阵，每个含混矩阵代表一类单独错误。

- del[*x*, *y*]　训练集中的字符*xy*在正确单词中应打字为*x*的次数。
- ins[*x*, *y*]　训练集中的字符*x*在正确单词中应打字为*xy*的次数。
- sub[*x*, *y*]　*x*被打字为*y*的次数。
- trans[*x*, *y*]　*xy*被打字为*yx*的次数。

注意，在这里选择插入和脱落的条件是前面一个字符，也可以选择后面一个字符为条件。使用这些含混矩阵，估算$p(t|c)$如下（其中，c_p表示单词c中的第p个字符）：

$$P(t|c)=\begin{cases}\dfrac{\mathbf{del}[c_{p-1},c_p]}{\mathbf{count}[c_{p-1}c_p]}, & \text{当删除时}\\ \dfrac{\mathbf{ins}[c_{p-1},t_p]}{\mathbf{count}[c_{p-1}]}, & \text{当插入时}\\ \dfrac{\mathbf{sub}[t_p,c_p]}{\mathbf{count}[c_p]}, & \text{当替换时}\\ \dfrac{\mathbf{trans}[c_p,c_{p+1}]}{\mathbf{count}[c_pc_{p+1}]}, & \text{当颠倒时}\end{cases} \tag{5.8}$$

图5.3给出了每个潜在可能的更正的最后概率；根据式（5.7）计算先验概率，使用式（5.8）和含混矩阵计算似然度，并把先验概率与似然度相乘。最后一栏给出了“归一化后的百分比”。

c	freq(c)	p(c)	p(t|c)	p(t|c)p(c)	%
actress	1343	0.000 031 5	0.000 117	3.69×10^{-9}	**37%**
cress	0	0.000 000 014	0.000 001 44	2.02×10^{-14}	**0%**
caress	4	0.000 000 1	0.000 001 64	1.64×10^{-13}	**0%**
access	2280	0.000 058	0.000 000 209	1.21×10^{-11}	**0%**
across	8436	0.000 19	0.000 009 3	1.77×10^{-9}	**18%**
acres	2879	0.000 065	0.000 032 1	2.09×10^{-9}	**21%**
acres	2879	0.000 065	0.000 034 2	2.22×10^{-9}	**23%**

图5.3　每个候选更正等级的计算。注意，等级最高的单词不是*actress*而是*acres*（在表中最底部的两行），因为*acres*可以通过两个途径生成。在Kerninghan et al.(1990)中给出了*del*[]，*ins*[]，*sub*[]和*trans*[]的全部含混矩阵

我们使用贝叶斯算法预见到acres是正确单词（这个单词的归一化百分比共计为45%），而actress则是第二位的最可能的正确单词。遗憾的是，这个算法在这里算错了。文章作者的意图可以从如下的上下文中看得很清楚：*...was called a “stellar and versatile acress whose combination of sass and glamour has defined her ...”*。从*acress*周围的词来看，它的正确单词显然应该是*actress*（女演员）而不是acres（英亩）。在第6章中将说明如何使用周围的词来更好地进行先验概率的计算。

我们所描述的算法要求手工标注数据来训练含混矩阵。Kernighan(1990)等的另一种不同方法是迭代地使用错拼更正算法本身来计算含混矩阵。迭代算法首先用相等的值启动一个矩阵，这时任何字符都是相等的，无论它是脱落还是被另一个字符替代等。然后，在一个拼写错误词的集合上运行

错拼更正算法。给出拼写错误类型及其相对应的更正，这时再计算含混矩阵，再运行拼写算法，这样地不断进行，便可以循序渐进地得到越来越好的含混矩阵。这个聪明的方法是重要的EM算法的一个实例(Dempster et al., 1977)，我们将在第 7 章和附录 D 中讨论这种 EM 算法。Kernighan et al.(1990)的算法的评测是取一些具有两种潜在更正的拼写错误，请三个评判人从中挑一个最好的更正来投票。程序与大多数评判人的投票是一致的，占了 87%。

5.6　最小编辑距离

前一节说明，贝叶斯算法与含混矩阵结合起来使用，可以排列出候选更正的等级。但是，Kernighan et al.(1990)的工作依赖于一个简单的假设，即假设每个词只有一个单独的拼写错误。我们还需要更强的算法来处理多个错误的情况。我们可以把这种算法理解为对符号串距离（string distance）问题的一般解。“符号串距离”是某种矩阵，矩阵中要说明两个符号串彼此的相近程度。贝叶斯方法可以看成是把这种算法应用于错拼更正问题的一个途径；我们要挑出“最接近于”错误的候选词，即挑出具有最高概率的错误。

找出符号串距离的最通用算法之一是最小编辑距离（minimum edit distance）算法。这种算法是 Wagner and Fischer(1974)提出的，但很多人也独立地发现了它，详情参见 5.12 节。两个符号串之间的最小编辑距离是指把一个符号串转换为另一个符号串时所需的最小编辑操作的次数。例如，intention和execution之间的距离是5个操作，可以用3种方法来表示：踪迹（trace）、对齐（alignment）和操作表（operation list）。如图 5.4 所示。

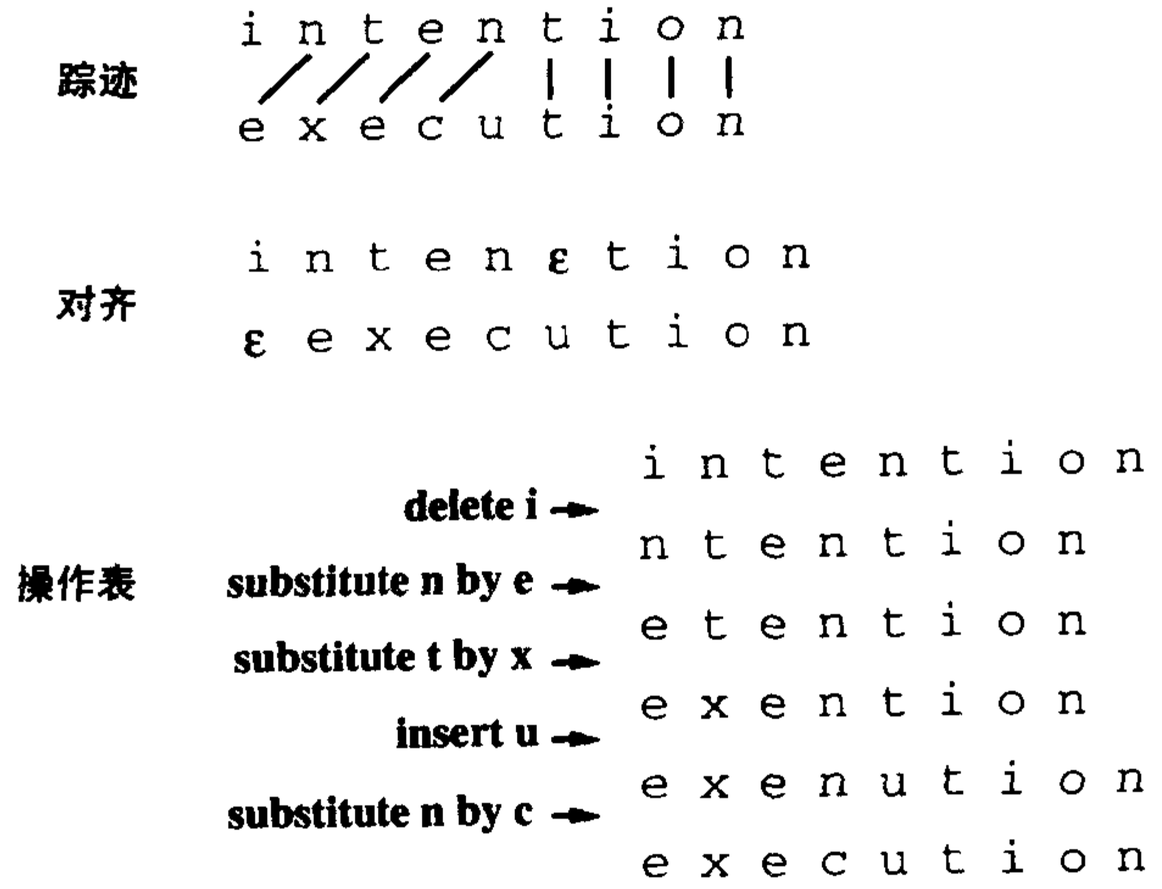


图 5.4　表示符号串差别的 3 种方法(Kruskal, 1983)

我们也可以给每个操作赋一个值（cost）或权值（weight）。两个序列之间的 Levenshtein 距离（Levenshtein distance）是最简单的加权因子，上面 3 种方法中的每个操作的值为 1(Levenshtein, 1966)。所以，在 *intention* 和 *execution* 之间的 Levenshtein 距离为 5。Levenshtein 还提出了另一种不同的度量方法，这种方法规定，插入或脱落操作的值为 1，不允许替代操作（实际上，如果把替代操作表示为一个插入操作加上一个脱落操作，那么替代操作的值为 2，这实际上也就等于允许了替代操作）。使用这样的度量方法，在 *intention* 和 *execution* 之间的 Levenshtein 距离应该是 8。我们也可以使用更复杂的函数来给操作加权。例如，使用前面讨论过的含混矩阵来给每种操作赋一个概率值。在这种情况下，我们谈的就不是两个符号串之间的“最小编辑距离”，而是一个符号串与另一

个符号串的“最大概率对齐”（maximum probability alignment）了。如果这样做，就能够通过给每个转换都乘以一个概率值的办法来增强最小编辑距离，这种增强了的最小编辑距离可用来估算给定候选更正的包含多个错误的错误类型的贝叶斯似然度。

最小编辑距离使用动态规划（dynamic programming）来计算。动态规划是一种算法的名字，首先由 Bellman(1957)提出。动态规划应用表驱动（table-driven）的方法，结合各个子问题的求解来求解整个问题。这一类算法包括了语音处理和语言处理中的大多数通用算法，例如在拼写错误更正中使用的最小编辑距离算法，在语音识别和机器翻译中使用的 Viterbi 算法（Viterbi algorithm）和向前算法（forward algorithm），在剖析中使用的 CYK 算法（CYK algorithm）和 Earley 算法（Earley algorithm）。在本章和第 7 章中，我们将介绍最小编辑距离算法、Viterbi 算法和向前算法（forward algorithm），在第 10 章中介绍 Earley 算法，在第 12 章中介绍 CYK 算法。

从直觉上来说，动态规划问题就是把一个大问题化解为不同的子问题来求解，再把这些子问题的解适当地结合起来，就可以实现对大问题的求解。例如，符号串 *intention* 和 *execution* 之间的最小编辑距离的求解，就要考虑被转换的不同单词的序列和“路径”。我们可以设想，某个符号串（例如 *exention*）处于最优的路径上（无论哪个路径）。动态规划的要求是，如果 *exention* 处于最优操作表中，那么最优的序列也应该包含从 *intention* 到 *exention* 的最优路径。为什么？如果还有从 *intention* 到 *exention* 的更短路径，我们就要用最短的路径来取代它，从而形成所有路径中的最短路径，在这种情况下这个最优序列就不是最优的了，这样就导出了矛盾。

当用于序列比较的动态规划算法工作时，要建立一个距离矩阵，目标序列的每个符号记录在矩阵的行上，源序列的每个符号记录在矩阵的列上（也就是说，目标序列的字母沿着底线排列，源序列的字母沿着侧线排列）。对于最小编辑距离来说，这个矩阵就是编辑距离矩阵。每个编辑距离单元[*i*, *j*]表示目标序列前 *i* 个字符和源序列的前 *j* 个字符之间的距离。每个单元可以作为周围单元的简单函数来计算；这样，从矩阵的开始点出发，就能够把矩阵中的所有项都填满。计算每个单元中的值时，我们取到达该单元时三个可能路径中的最小路径为其值：

$$P(t|c) = \min \begin{cases} distance[i-1, j] + ins\text{-}cost(target_i) \\ distance[i-1, j-1] + subst\text{-}cost(source_j, target_i) \\ distance[i, j-1] + del\text{-}cost(source_j)) \end{cases} \tag{5.9}$$

这个算法如图 5.5 所示。图 5.6 是应用这个算法计算 *intention* 和 *execution* 之间的距离的结果，计算时采用了 Levenshtein 距离，其中插入和脱落分别取值为 1，替代取值为 2。

function MIN-EDIT-DISTANCE(*target*, *source*) **returns** *min-distance*

$n \leftarrow$ LENGTH(*target*)
$m \leftarrow$ LENGTH(*source*)
Create a distance matrix *distance[n+1,m+1]*
distance[0,0] $\leftarrow 0$
for each column *i* **from** 0 **to** *n* **do**
  for each row *j* **from** 0 **to** *m* **do**
    distance[i, j] $\leftarrow$ MIN(*distance*[*i*−1,*j*] + *ins-cost*(*target*$_i$),
                distance[*i*−1,*j*−1] + *subst-cost*(*source*$_j$, target$_i$),
                distance[*i*,*j*−1] + *del-cost*(*source*$_j$))

图 5.5　最小编辑距离算法，动态规划算法的一个实例

n	9	10	11	10	11	12	11	10	9	**8**
o	8	9	10	9	10	11	10	9	**8**	9
i	7	8	9	8	9	10	**9**	**8**	9	10
t	6	7	8	7	8	**9**	**8**	9	10	11
n	5	6	7	6	7	**8**	9	10	11	12
e	4	5	6	**5**	**6**	7	8	9	10	11
t	3	4	**5**	6	7	8	9	10	11	12
n	2	3	**4**	5	6	7	8	8	10	11
i	1	**2**	3	4	5	6	7	8	9	10
#	**0**	1	2	3	4	5	6	7	8	9
	#	e	x	e	c	u	t	i	o	n

图5.6　应用图5.5中的算法计算 *intention* 和 *execution* 之间的最小编辑距离，计算时采用了Levenshtein距离，插入和脱落分别取值为1，替代取值为2。字符自身替代取值为0

5.7　英语的发音变异

> When any of the fugitives of Ephraim said: 'Let me go over.' the men of Gilead said unto him: 'Art thou arn Ephraimite?' If he said: 'Nay'; then said they unto him: 'Say now Shibboleth'; and he said 'Sibboleth'; for he could not frame to pronounce it right; then they laid hold on him, and slew him at the fords of the Jordan.
>
> Judges 12：5-6[①]

这是《旧约全书·士师记》中的一段故事，这段骇人听闻的故事提醒我们，发音变异在政治斗争中是多么重要。即使是在发音问题的计算机应用中（我们希望它没有很浓的政治色彩），为了正确地模拟发音变化的情况，发音问题仍然非常重要。我们已经知道，音位在不同的语音环境中可以被实现为不同的音位变体。我们也讲过，在语音合成中，如何写一些规则和使用转录机来模拟这些变化。遗憾的是，这些模型都把语音变异的本质方面大大地简单化了。具体地说，影响发音变异的因素，除了语音环境之外还有很多的因素。本节将总结这些类型的语音变异；下一节将介绍模拟这些变异的概率工具。

发音变异的涉及面很广。图5.7说明了单词 *because* 和 *about* 的最普遍的发音变异情况，这些材料来自美国英语电话对话的Switchboard语料库，是手工转写的。注意，当话语处于连续语流中时，这两个单词的发音变异的幅度是很大的。

这样的变异是怎样引起的呢？发音变异有两大类：词汇变异（lexical variation）和音位变异（allophonic variation）。我们可以把词汇变异看成是用来表示词表中的单词的语音片断的差异，而音位变异则是单独的语音片断在不同的上下文中的发音差异。在图5.7中，大多数发音变异是音位变异，是由于周围语音、音节结构等的影响而产生的差异。但是，单词 *because* 可以发音为单音节的 *'cause*，又可以发音为双音节的 *because*，这大概是一种词汇事实，必须联系语音信息的层次来处理。

词汇变异（虽然它也会影响音位变异）的一个重要来源是社会语言变异（sociolinguistic variation）。社会语言变异是由于语言外的因素引起的，例如说话者的社会等同性和社会背景等都会引起社会语言变异。方言变异（dialect variation）是一种社会语言变异。说美国英语南方方言的人在某些带元音[aɪ]的词中，用单元音[a]或准单元音[aɛ]来替代双元音。在这些方言中，*rice* 的发音为[ra:s]。非洲

① 因为这段文字中涉及到某些英文单词的读音，为了便于读者理解，我们提供译文如下，以便读者对照阅读。
以色列的逃亡者中的一个人说："让我过去。"吉利德人冲着他说："你是厄福拉衣米德人吗？"他回答："不是"，他们又冲着他说："现在说口令"，然后他就说"口令"，由于他不能把发音安排得很正确，所以，他们扣住了他，然后就把他转送到约旦的浅滩地去了。
ephraimite（厄福拉衣米德）的发音和Ephraim（以色列）的发音相近——译者注。

美国本地英语（African-American Vernacular English，简称 AAVE）与南美英语（Southern American English）一样，有的元音与一般的美国英语的发音不同，有些单词的发音很特殊，例如，*Business* 的发音为[bɪdnɪs]，ask 的发音为[æks]。有些老人或不是从美国西部和中西部来的人，对单词 *caught* 和 *cot* 中的元音的发音不同（分别为[kɔt] 和[kɑt]）。年轻的美国人或从西部来的人，对 *caught* 和 *cot* 这两个词的发音是相同的。在这些方言中，一般不区别[ɔ]和[ɑ]。来自纽约的人，如本书第一作者的父母，对 *Mary*[meɪri]，*marry*[mæri]和 *merry*[mɛri]这三个词的发音是不同的，而纽约的其他一些人，如本书的第二作者，对 *Mary* 和 *merry* 的发音完全相同，但对 *marry* 的发音不同于 *Mary* 和 *merry* 的发音。多数的美国人将这三个词都同样地发为[mɛri]。对英语方言有兴趣的读者可以参阅 Wells (1982)，这是世界上最全面的关于英语方言的研究著作。

because			about		
IPA	**ARPAbet**	**%**	**IPA**	**ARPAbet**	**%**
[bikʌz]	[b iy k ah z]	27%	[əbaʊ]	[ax b aw]	32%
[bɨkʌz]	[b ix k ah z]	14%	[əbaʊt]	[ax b aw t]	16%
[kʌz]	[k ah z]	7%	[baʊ]	[b aw]	9%
[kəz]	[k ax z]	5%	[ʌbaʊ]	[ix b aw]	8%
[bɨkəz]	[b ix k ax z]	4%	[ɨbaʊt]	[ix b aw t]	5%
[bɪkʌz]	[b ih k ah z]	3%	[ɨbæ]	[ix b ae]	4%
[bəkʌz]	[b ax k ah z]	3%	[əbæɾ]	[ax b ae dx]	3%
[kʊz]	[k uh z]	2%	[baʊɾ]	[b aw dx]	3%
[kɜ]	[k s]	2%	[bæ]	[b ae]	3%
[kɨz]	[k ix z]	2%	[baʊt]	[b aw t]	3%
[kɪz]	[k ih z]	2%	[əbaʊɾ]	[ax b aw dx]	3%
[bikʌʒ]	[b iy k ah zh]	2%	[əbæ]	[ax b ae]	3%
[bikʌs]	[b iy k ah s]	2%	[bɑ]	[b aa]	3%
[bikʌ]	[b iy k ah]	2%	[bæɾ]	[b ae dx]	3%
[bikɑz]	[b iy k aa z]	2%	[ɨbaʊɾ]	[ix b aw dx]	2%
[əz]	[ax z]	2%	[ɨbɑt]	[ix b aa t]	2%

图 5.7　because 和 about 的 16 种最常见的发音，材料来自美国英语电话会话口语的手工转写语料库(Godfrey et al., 1992; Greenberg et al., 1996)

除了方言的因素之外，造成社会语言差异的另一个原因是语域（register）或风格（style）。由于风格造成的发音差异表现在，由于说话者谈话的对象或社会地位的不同，同样一个说话者对同一个单词的发音可能也有所不同；上面的例子中的 *because* 和 *'cause* 的区别，大概就是由于这样的原因造成的。关于风格变异研究得最好的一个例子是后缀 *-ing*（例如在 *something* 中），这个后缀的发音可以发音为[ɪŋ]，也可以发音为/ɪn/（这时通常写为 *somethin'*）。大多数说话者都使用两种形式；Labov(1966)指出，当人们在比较正式的场合就使用[ɪŋ]，当人们在比较随便的场合就使用[ɪn]。事实上，一个说话者是使用[ɪŋ]还是使用[ɪn]明显地随着社会背景的不同，随着说话者的性别的不同，随着另一个说话者的性别的不同而变化。Wald and Shopen(1981)发现，男性比女性更喜欢使用非标准的形式[ɪn]。当听话者是一个女性时，无论是男性还是女性都更喜欢使用标准形式[ɪŋ]；当男性（而不是女性）与朋友谈话时，则倾向于使用[ɪn]。

词汇变异发生在词汇层面，而音位变异则发生在表层形式，它反映了语音因素和发音因素[①]。例如，图 5.7 中单词 *about* 的变异是由于两个元音变为一个元音而引起的，或者是由于词末[t]的改变而引起的。这种变异是由于我们在讨论音位 /t/ 的实现时的那些音位变体规则而引起的。例如，*about* 发音为[əbaʊɾ]/[ax b aw dx]在词末有一个闪音是由于这个单词的下一个词是以元音开头的 *it*;

① 为了某些目的，我们有时要区分音位变异和所谓的“随选的音位规则”，但在本书里我们把两者统称为“音位变异”。

序列 *about it* 的发音为[əbaʊɾɨ]/[ax b aw dx ix]。与此相似，我们注意到词末[t]经常脱落（*about* 发音为[baʊ]/[b aw]）。对于在这些场合的脱落，我们也可以把它看成一种简化；[t]脱落的很多情况事实上是通过稍微改变元音的音质就可以实现的，这种稍微的改变称为元音的声门化（glottalization），我们这里没有用音标把它表示出来。

当我们在前面讨论这些规则时，认为它们都是确定性的，即给定一个环境就总是可以应用某条规则，然而事实并非如此。每一条音位变体规则都依赖于一组十分复杂的因素，这些因素需要在概率上加以解释。本节的剩余部分将总结更多的此类规则，并讨论各种影响的因素。很多此类规则都是模拟协同发音（coarticulation）的。协同发音是由于相邻语音片断的发音运动而引起的某一语音片断发音的改变。把英语的音位和它们的音位变体联系起来的大多数音位变体规则可以分为如下为数不多的几种类型：同化（assimilation）、异化（dissimilation）、脱落（deletion）、闪音化（flapping）、元音弱化（vowel reduction）和增音（epenthesis）。

同化就是把一个语音片断改变得更像它邻接的语音片断。[t]在齿音[θ]之前齿音化为[t̪]就是同化。英语中的另一种普遍的同化是腭化（palatalization）。腭化也是一种跨语言的同化现象。当一个语段的下一个语段是上腭音或龈腭音时，它就会收缩离开它通常的位置而向硬腭靠拢，这时就发生了腭化。在大多数情况下，/s/ 变为 [ʃ]，/z/ 变为 [ʒ]，/t/ 变为 [tʃ]，/d/ 变为 [dʒ]。在图 5.7 中，*because* 的发音变为[bikʌʒ]（ARPAbet [b iy k ah zh]），这就是腭化。这时，对于 because 的最后一个片断，词汇读音为 /z/，实际发音为[ʒ]，因为它下面的一个词是 *you' ve*。因此，字符序列 *because you' ve* 的发音应该是[bikʌʒuv]。腭化规则可简单地表示如下：

$$\left\{\begin{matrix}[s]\\[z]\\[t]\\[d]\end{matrix}\right\} \Rightarrow \left\{\begin{matrix}[ʃ]\\[ʒ]\\[tʃ]\\[dʒ]\end{matrix}\right\} / __ \{y\} \qquad (5.10)$$

图 5.8 是 Switchboard 语料库中的有关例子。

短语	IPA 词汇发音	IPA 弱化发音	ARPAbet 弱化发音
set your	[sɛtjɔr]	[sɛtʃɚ]	[s eh ch er]
not yet	[nɑtjɛt]	[nɑtʃɛt]	[n aa ch eh t]
last year	[læstjir]	[læstʃir]	[l ae s ch iy r]
what you	[wʌtju]	[wətʃu]	[w ax ch uw]
this year	[ðɪsjir]	[ðɪʃir]	[dh ih sh iy r]
because you've	[bikʌzjuv]	[bikʌʒuv]	[b iy k ah zh uw v]
did you	[dɪdju]	[dɪdʒyʌ]	[d ih jh y ah]

图 5.8　Switchboard 语料库中的腭化的例子；词目 *you*（包括 *your, you' ve, you' d*）在很多场合都会引起腭化；如果后面的词是 *year*（特别在短语 *this year* 和 *last year* 中），也会引起腭化

注意，在图 5.8 中，[t]是否发生腭化取决于诸如频度之类的词汇因素（在高频词和高频短语中，[t]最容易发生腭化）。

脱落（deletion）在英语的口语中是很普遍的。在上面的例子中，我们已经看到了在单词 *about* 和 *it* 中，词末的 /t/ 的脱落现象。/t/ 和 /d/ 的脱落经常发生在辅音之前，或者当它们是在三辅音序列或双辅音序列中的一部分时，也常常脱落。

$$\left\{ \begin{matrix} t \\ d \end{matrix} \right\} \Rightarrow \emptyset / \mathrm{V} __ \mathrm{C} \tag{5.11}$$

图 5.9 是 /t/ 和 /d/ 脱落的一些例子。

短语	IPA 词汇发音	IPA 弱化发音	ARPAbet 弱化发音
find him	[faɪndhɪm]	[faɪnɪm]	[f ay n ix m]
around this	[əraʊndðɪs]	[ɪraʊnɪs]	[ix r aw n ih s]
mind boggling	[maɪnbɔgl̩ɪŋ]	[maɪnbɔgl̩ɪŋ]	[m ay n b ao g el ih ng]
most places	[moʊstpleɪsɪz]	[moʊspleɪsɪz]	[m ow s p l ey s ix z]
draft the	[dræftði]	[dræfði]	[d r ae f dh iy]
left me	[lɛftmi]	[lɛfmi]	[l eh f m iy]

图 5.9 Switchboard 语料库中 /t/ 和 /d/ 脱落的一些例子。其中有些例子也可能出现腭化，而不出现完全的脱落

影响 /t/ 和 /d/ 脱落的很多因素都被广泛地研究过。例如，/d/ 比 /t/ 更容易脱落。在辅音之前，/t/ 和 /d/ 两者都更容易脱落(Labov, 1972)。and 词末的 /d/ 以及 *just* 中的 /t/ 都特别容易脱落(Labov, 1975; Neu, 1980)。Wolfram(1969)发现，在快速说话时或很随便地说话时，容易发生脱落，年轻人和男性更容易发生脱落。当两个单词包围着一个片断而作为一个短语单元使用时，或者频繁地一起出现时，都容易发生脱落(Bybee, 1996)。当两个单词具有很高的互信息（mutual information）或者三元可预测性（trigram predictability）时，容易脱落(Gregory et al., 1999)。当两个单词由于其他原因被紧密地联系在一起时容易脱落(Zwicky, 1972)。Fasold(1972)，Labov(1972)以及其他一些研究者指出，如果词末的 /t/ 和 /d/ 是过去时态的词尾，它们就很少脱落。例如，在 Switchboard 语料库中，单词 *around*（73% 出现 /d/ 脱落）比单词 *turned*（30% 出现 /d/ 脱落）更容易出现 /d/ 脱落，尽管这两个单词的频度很接近。

很多学者指出，特别是应该参阅 Rhodes(1992)，闪音化的情况比在第 4 章中介绍的复杂得多。闪音化时，前面的元音要发得高一些，尽管这并不总是必要的（例如，在单词 *thermometer* [θɚˈmɑmɪɾɚ]中总是出现闪音）；后面的元音倾向于非重读，尽管这也不总是必要的。/t/ 比 /d/ 更容易发生闪音化。闪音化与音节、音步和单词的界限都有着非常复杂的交互关系。当说话者的话说得很快时就容易出现闪音化；闪音化也容易发生在词末，特别是当在这个位置与后面的单词出现搭配关系时(Gregory et al., 1999)，此时具有较高的互信息。当说话者进行强势发音（superarticulate）时，闪音化就较少发生，因为这时说话者要用特别清楚的形式来说话，例如，当用户对一个语音识别系统讲话时，就要使用强势发音(Oviatt et al., 1998)。还有一种鼻化闪音[ɾ̃]。发鼻化闪音时，舌头的运动与口腔闪音相似，但是软腭下降得低一些。最后，即使在环境满足要求时，闪音化也不一定发生，所以说闪音规则或转录机是有概率的，下面会讨论这个问题。

现在我们来研究一种最重要的音位过程——元音弱化（vowel reduction）。在元音弱化时，很多非重读音节中的元音变为弱化元音。最常见的弱化元音是混元音（schwa）（[ə]）。重读音节是肺中出来的空气很强的音节。重读音节一般比较长，比较响，音高一般比非重读音节的更高。英语中的非重读音节中的元音一般都不具备完整的形式；发音动作不像发完整的元音时那样完全。这时，口腔的形状是中性的；舌位不特别高，也不特别低。例如，单词 *parakeet*（长尾小鹦鹉）中的第二元音就是混元音，这个单词的发音为：[pærəkit]。

混元音是一个最常见的弱化元音，但它不是惟一的弱化元音，至少在方言中是这样的。Bolinger (1981)提出了三种弱化元音：弱化央元音[ə]，弱化前元音[i]和弱化圆唇元音[ɵ]。但是，在大多数计

算机发音词典以及音位系统的计算机模型中，只限于讨论一个弱化元音（[ə]）（例如，PRONLEX和CELEX），或者至多两个（[ə]=ARPAbet[ax]和[ɨ]=ARPAbet[ix]）。Miller(1998)曾经训练一个神经网络根据语音上下文来给元音[ə]或元音[ɨ]归类，建议在语音识别和文本-语音转换系统中，只需要一个弱化元音是比较合适的。当然，Wells(1982, p.167-168)已经注意到，在包括美国英语和爱尔兰英语的英语方言中，[ə]和[ɨ]有合流的倾向，称这样的现象为弱元音合并（weak vowel merger）。

最后还要说明的是，不是所有的非重读元音都要弱化。任何一个元音，特别是双元音，即使在非重读的位置也能保持它们的完整特性。例如，元音[eɪ]（ARPAbet [ey]）可以出现在单词*eight*['eɪt]的重读位置，也可以出现在单词*always*['ɔ.weɪz]的非重读位置。一个元音是否弱化取决于很多因素。例如，单词*the*可以按完全的元音读为[ði]，也可以按弱化元音读为[ðə]。在快速讲话时，在比较随便的场合，当下一个单词以辅音开头时，*the*倾向于发为弱化元[ðə]。当下一个单词以元音开头时，如果说话者进行有准备、有计划的讲话，*the*倾向于发为完全元音[ði]；如果说话者不知道下面将要说什么，他倾向于使用完全元音而不使用弱化元音(Fox Tree and Clark, 1997)。关于在TIMIT和Switchboard语料库中影响元音弱化的因素的详细研究，可参阅Keating et al.(1994)和Jurafsky et al.(1998)。影响元音弱化的其他因素还有单词的频度，是否为短语中的最后一个元音，甚至说话者的个人语言习惯等。

5.8　发音问题研究中的贝叶斯方法

HEAD KNIGHT OF NI: Ni!
KNIGHTS OF NI: Ni! Ni! Ni! Ni! Ni!
ARTHUR: Who are you?
HEAD KNIGHT: We are the Knights Who Say... 'Ni'!
RANDOM: Ni!
ARTHUR: No! Not the Knights Who Say 'Ni'!
HEAD KNIGHT: The same!
BEDEVERE: Who are they?
HEAD KNIGHT: We are the keepers of the sacred words: 'Ni', 'Peng', and 'Neee-wom'!

Graham Chapman, John Cleese, Eric Idle, Terry Cilliam, Terry Jones, and Michael Palin, *Monty Python and the Holy Grail* 1975[①].

我们曾经使用贝叶斯算法进行拼写错误的最优更正，这个算法也可以用来解决语音识别中通常称为“发音”（pronunciation）的子问题。在解决“发音”问题时，对于给出的音子序列（series of phones），我们的任务是计算出生成这些音子序列的概率最大的单词。在本章里，我们假定音子符

① 本节的开场白是上面的《大蟒与圣盘》中称为“Ni”的骑士领袖与亚瑟等人的一段对话，由于“Ni”与英文“骑士”（knight）的发音极其相似，他们为此而费了许多口舌。下面给出了译文，供读者对照参考，但译成中文后显得平淡无奇。

NI的骑士领袖：Ni!
NI的众骑士们：Ni! Ni! Ni! Ni! Ni!
ARTHUR：你们是什么人?
NI的骑士领袖：我们是说“Ni”的骑士!
RANDOM：Ni!
ARTHUR：不，没有骑士说“Ni”！
骑士领袖：骑士和“Ni”是一回事儿!
BEDEVERE：他们是什么人?
骑士领袖：我们是保存神圣的单词“Ni”，“Pen”和“Neee-wom”的人!

号串是正确的，这样就把这个问题大大地简化了，而一个实际的语音识别系统对于每个音子都要进行概率评估，所以，对于任何音子的等同性，语音识别系统从来都没有把握。在第7章中，我们将解除这种假定，现在先来研究这个简化了的问题。

开始时还要做另一种简化，假定已经知道了单词的边界在哪里。本章在后面将说明，我们能够同时发现单词的边界（“片断”）和模拟发音变异。

在句子的开头，当单词*I*之后出现音子序列[ni]时，我们来研究如何解释这个音子序列[ni]的问题。先停下来并看一看，在继续读下去之前，能不能想出一个单词，它的发音是[ni]。在这里，我们不允许选择前面开场白中的那个“Ni”。

也许你能够想到单词*knee*。这个单词实际的发音确实是[ni]。但是，通过调查Switchboard语料库可以知道，发音为[ni]的单词一共有7个！这7个单词是：“*the*（这），*neat*（整洁），*need*（必须），*new*（新），*knee*（膝盖），*to*（向）和*you*（你）”。

单词*the*的发音为什么会是[ni]呢？对于这个发音（以及除了*knee*之外的其他各个单词的发音）的解释要根据我们在第4章中讨论过的依赖于上下文的发音变异。例如，我们已经看到，[t]和[d]处于词末时经常脱落，特别是当它处于具有舌冠特征的音之前时；因此，当*neat*处于单词*little*之前时，就发音为[ni]（*neat little* → [nilə̩l]）。*the*的发音变为[ni]是由在第4章中讨论过的回退同化（regressive assimilation）过程引起的。我们知道，在鼻音同化时，在鼻音之前或之后的音子要采取鼻音的发音方法，所以[θ]在实际上就有可能发音为[n]。在Switchboard语料库中，在单词*in*, *on*和*been*之后，the在不少场合下都发音为[ni]（*in the* → [ɪnni]）。在*New York*这个词中，*new*的发音常常为[ni]，在[y]之前，元音[u]变为前元音[i]。

在*talking to you* → [tɔkɪniyu]的过程中，*to*的发音变为[ni]，这里的[u]在[y]之后发生腭化，*talking*词末的音发为[n]，而*to*开头的音也发为[n]，这两个[n]共同作用的结果，发为一个音子[n]。因为这个音子是两个分开的单词合起来形成的，我们不打算模拟这种特殊的映射。在本节的其他地方，只考虑5个词作为[ni]的候补词汇形式：*knee*, *the*, *neat*, *need*和*new*。

在前一节中我们已经知道，贝叶斯错拼更正算法包括两个部分：候选生成和候选打分。语音识别系统通常采用不同的体系结构来处理存储的语音。在这样的体系结构中，每个发音事先被扩充为它的所有可能的变体，然后再按它们的分数预先存储起来。因此，不必进行候选生成，单词[ni]就可以简单地存储为一个可能生成该单词的词表。让我们采用这种方法，计算每个单词的先验概率和似然度。

我们将根据式（5.12）选择先验概率和似然度的乘积为最大的单词，在式（5.12）中，y表示音子序列（在我们的例子中，这是[ni]，w表示候选单词[*the*, *new*等]）。这样，最大可能的单词为：

$$\hat{w} = \underset{w \in W}{\operatorname{argmax}} \overbrace{P(y|w)}^{\text{似然度}} \overbrace{P(w)}^{\text{先验概率}} \tag{5.12}$$

看来我们似乎可以使用错拼更正中的含混矩阵方法来生成式（5.12）中的似然度$p(y \mid w)$。但是事实证明，这样的含混矩阵在发音问题中不如在拼写问题中好用。拼写错误只是稍微改变单词的形式，对于单词形式的改变不太大，而在发音问题中，词汇形式和表层形式之间的改变是非常大的。含混矩阵对于单个错误工作得很好，如上所述，它在拼写错误的问题中得到了普遍的使用。可是，从第4章中我们知道，发音变异受到周围音子、词汇频度、重音以及其他韵律因素的强烈影响，因此，发音变异的概率模型所包含的因素要比简单的含混矩阵包含的因素多得多。

我们可以用一个简单的方法来生成发音的似然度，这个方法就是概率规则（probabilistic rule）。首先提出用概率规则来处理发音问题的是Labov(1969)（其中把概率规则称为变量规则）。其思想是使用第4章中讲过的发音变异，并把发音变异与概率联系起来；然后就可以在一个词表上运行这些概率规则，生成各种可能的不同表层形式，每个表层形式带上它们本身的概率。例如，在解释*the*为什么会发音为[ni]的鼻音同化规则中，如果前面的单词以[n]为结尾，或者有时以[m]为结尾，则词首的[ð]变为[n]：

$$[0.15]\ \text{ð} \Rightarrow n\ /\ [+nasal]\ \#\ __ \tag{5.13}$$

规则左部的[0.15]是概率；这个概率可以根据足够大的标注语料库（例如，经过标音的Switchboard语料库）计算得到。设*ncount*表示当前面的单词以鼻音为结尾时，单词开头的词汇形式[ð]实现为表层形式[n]的次数（在Switchboard语料库中为91次）。设*envcount*表示当前面的单词以鼻音为结尾时，词汇形式[ð]（当它在表层中实现时）的总出现次数（在Switchboard语料库中为617次）。结果其概率为：

$$\begin{aligned} P(\text{ð} \rightarrow n\ /\ [+nasal]\ \#\ __) &= \frac{ncount}{envcount} \\ &= \frac{91}{617} \\ &= 0.15 \end{aligned}$$

我们也可以用同样的方法计算出在其他单词的[ni]发音中，同化规则和脱落规则的概率。图5.10说明了这样的一些样本规则以及根据Switchboard语料库的发音数据库训练得到的概率。

单词	规则名	规则	P
the	鼻化同化	ð ⇒ n / [+*nasal*] # ___	[0.15]
neat	词末t脱落	*t* ⇒ ∅ / *V* ___ #	[0.52]
need	词末d脱落	*d* ⇒ ∅ / *V* ___ #	[0.11]
new	u前音化	*u* ⇒ *i* / ___ # [y]	[0.36]

图 5.10　这些单词发音为[ni]时，由于在连续语音的上下文中而产生的发音变异的简单规则

现在我们来计算每个单词的先验概率$P(w)$。对于拼写更正，我们使用大规模语料库中单词的相对频度来表示先验概率，如果一个单词在4400万单词的语料库中出现44 000次，那么它的先验概率估算值就是44 000 / 44 000 000，即0.001。对于发音问题，我们要把书面文本语料库和口语语料库结合起来，根据这样结合的语料库来计算先验概率。Brown语料库从题材各异的（包括新闻、小说、非小说、学术著作等）500篇书面材料样本中收集了100万单词的语料，这个语料库是Brown大学在1963年至1964年间收集的(Kučera and Francis, 1967; Francis, 1979; Francis and Kučera, 1982)。Switchboard语料库（树库）从电话会话中收集了140万词的口语语料。我们从这两个语料库中收集书面语和口语的语料样本。下面的表中列出了我们所研究的5个单词的概率；每个概率是根据它们的出现频度和上述结合起来的语料库中的单词数来计算并加以归一化而得出的（加0.5*单词类型数；所以，总的分母是2 486 075 + 30 836）。

w	freq(w)	*p* (*w*)
knee	61	0.000 024
the	114 834	0.046
neat	338	0.000 13
need	1417	0.000 56
new	2625	0.001

现在我们几乎可以回答原来的问题了，这个问题是：在句子的开头，当发音为[ni]的单词的前一个单词为*I*时，这个发音最可能对应的单词是哪个？首先，让我们把$p(w)$和$p(y \mid w)$相乘，得到一个估算值；从最大的概率估算值到最小的概率估算值排序如下（这时，*the* 的概率估算值为 0，因为它前面一个音子不是鼻音[n]，而且没有其他的规则允许[ð]的发音实现为[n]）：

Word	$p(y\|w)$	$p(w)$	$p(y\|w)p(w)$
new	0.36	0.001	0.000 36
neat	0.52	0.000 13	0.000 068
need	0.11	0.000 56	0.000 062
knee	1.00	0.000 024	0.000 024
the	0	0.046	0

这个算法告诉我们，*new*是最大可能的单词。然而，这却是一个错误的答案。根据Switchboard语料库中的事实，在单词*I*之后的符号串[ni]应该来自单词*need*。人们解决这个问题的最简单途径是词汇平面上的知识；人们知道，词串*I need*…要比词串*I new*.…具有更大的可能性。但是，在解决这个问题时也没有必要抛弃贝叶斯模型，我们只需要修改它，使我们的模型也知道*I need*比*I new*具有更大的可能性。在第6章中将会看到，我们可以使用稍微聪明一点的$p(w)$估算方法，这种估算称为二元语法（bigram）估算；在这种情况下不是只考虑单独的*need*的概率，而是要考虑跟随在*I*之后的*need*的概率。

这个贝叶斯算法事实上是所有现代语音识别系统的一部分。各种算法的不同之处在于如何检查声学信号中的音子，以及使用什么搜索算法来有效地计算贝叶斯概率，以便在连接起来的语音中找出正确的词串（见第7章）。

5.8.1 发音变异的决策树模型

在前一节中可以看到，使用概率来模拟发音变异可以增强手写规则的效率。Riley(1991)和Withgott and Chen(1993)提出了另一种手工书写规则的方法，这种方法被证明是很有用的。这种方法使用决策树（decision tree），特别是使用一种分类回归树（Classification and Regression Tree，简称CART），从标注语料库中自动推导出词汇到表层发音的映射关系(Breiman et al., 1984)。决策树提取由特征集所描述的情况，并把这种情况分类为范畴和相关的概率。在发音问题研究中，可以训练决策树来提取一个词汇音子和它的各种上下文特征（包围的音子、重音、音节结构信息以及词汇的等同性），并选择一个适合的表层音子来实现它。我们可以把在前面的错拼更正中使用的含混矩阵看成是一种蜕化的决策树，因此替代矩阵取一个词汇音子作为输入，然后输出在潜在的表层音子中的一个概率分布来替代这个词汇音子。决策树的优点是它可以从标注语料库中自动推导出来，而且都很精确。决策树只提取相关的特征，所以数据稀疏问题比含混矩阵少一些，因为含混矩阵要以每个相邻的音子作为条件。

例如，图5.11是根据Switchboard语料库得出的关于音位/t/发音的一个决策树。这个决策树不包括闪音化（闪音化由另外的决策树来描述），但是它模拟/t/在辅音前比在元音前更可能脱落的事实。注意，实际上这个决策树自动推导出了元音类和辅音类。另外还要注意，如果/t/没有在一个辅音前面脱落，它就很可能是没有除阻的。最后还要注意，/t/很容易在音节头的位置脱落。

对于发音的决策树模型有兴趣的读者，可以参阅Riley(1991)和Withgott and Chen (1993)，也可以参阅关于决策树的导论性教材，如Russell and Norvig(1995)。

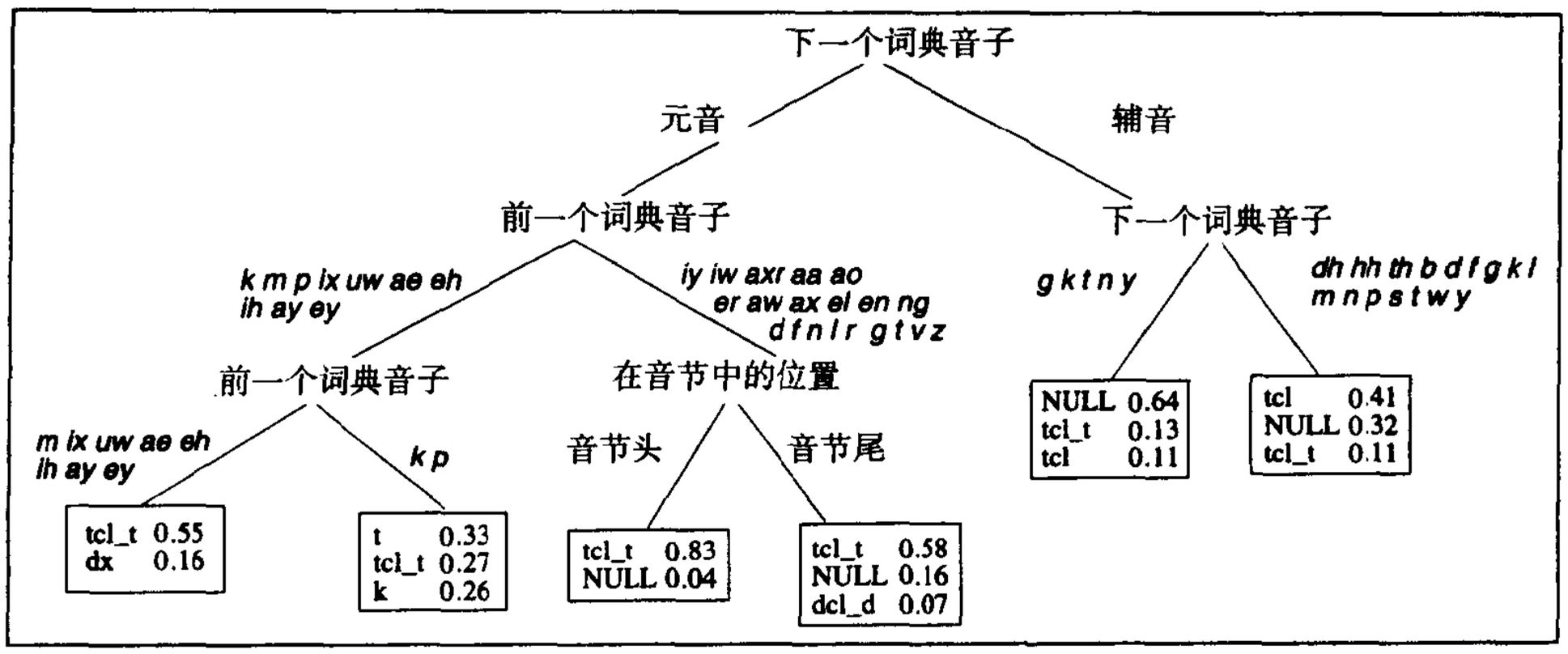


图5.11　根据Switchboard语料库得出的关于音位/t/发音的经过手工修剪的决策树（由Eric Fosler-Lussier提供）。这个特殊的决策树没有模拟闪音化，因为闪音已经在词典中列出了。这个决策树能够自动推导出元音和辅音范畴。我们在每个叶子结点上只列出了最可能的实现情况

5.9　加权自动机

我们在前面说过，为了提高效率，通常把编辑好的各种发音变异存储在词表中。这种词表的两种最普通的表示是trie和加权有限状态自动机/转录机或概率FSA/FST(Pereira et al., 1994)。我们将在第7章中讨论trie，这里重点讨论加权自动机。

加权自动机是由有限自动机扩充而成的，在加权自动机的每个弧上标有概率，表示下一步走哪一条途径的可能性。离开同一个结点的所有弧上的概率之和应该为1。图5.12是关于单词*tomato*的两个加权自动机，取自Russell and Norvig(1995)。上面的自动机表示在不同的方言中*tomato*一词的第二个元音的两种不同发音。下面一个自动机说明了更多的发音情况（请读者算一下有多少个不同的情况？），表示*tomato*中的第一个元音的可能的弱化和脱落，以及词末[t]的可能的闪音化。

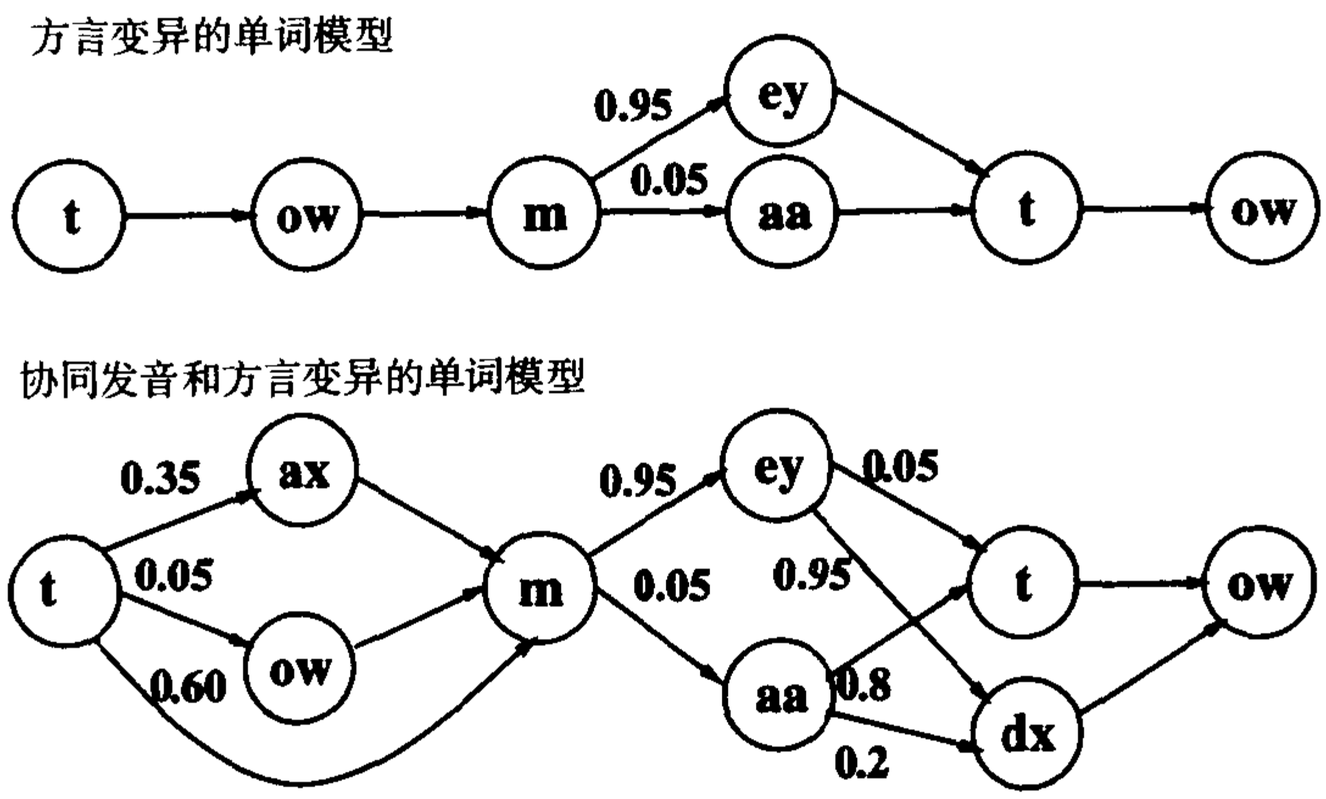


图5.12　你说[t ow m ey t ow]，我说[t ow m aa t ow]。单词*tomato*的两个发音网络，取自Russell and Norvig(1995)。上面一个自动机模拟社会语言变异（在某些英国英语或美国英语东部方言中）；下面一个自动机加上了协同发音效应。注意音位变异和社会语言变异之间的关联作用；在具有元音[ey]的方言中，闪音化的可能性比其他方言大

马尔可夫链是加权自动机的一种特殊情况，在马尔可夫链中，输入符号序列惟一地确定了自动机将通过的状态。由于这些状态不能表示符号序列内在的歧义问题，只有当概率值赋予无歧义的序列时，马尔可夫链才是有用的。第6章将要讨论的N元语法模型都是马尔可夫链，因为在处理每个单词时都将其看成是没有歧义的。实际上，在语音和语言处理中使用的加权自动机等价于隐马尔可夫模型（Hidden Markov Models，简称HMM）。为什么在本章中介绍加权自动机，而在第7章中才介绍HMM呢？这是因为这两个模型提供了不同的比喻；把某些问题想像成加权自动机比想像成HMM更容易理解一些。加权自动机的比喻通常用于输入字母表比较清晰地映射于底层字母表的场合。例如，在打字输入的错拼更正问题中，包含字母和自动机状态的输入序列可以对应于字母。因此，就可以很自然地把这个问题想像成从一个符号集合转录为同一个符号集合的问题，只是做了某些修改而已；正因为这样，使用加权自动机来进行错拼更正就是非常自然的了。在手写输入错拼更正的问题中，输入序列是可以看见的，输入字母表是由直线、角和曲线构成的字母表。这里，我们不是把一个字母表转录成它自己，在把某个输入序列考虑为状态序列之前，应该把这个输入序列加以分类。隐马尔可夫模型（HMM）提供了一个更恰当的比喻，因为HMM对于输入序列和状态序列很自然地分别使用不同的字母表来处理。但是，因为任何一个概率自动机，只要其中的输入序列不是惟一地指定其状态序列，它就可以被模拟为一个HMM，其差别只是比喻上的差别，而不是解释能力上的差别。

加权自动机可以使用很多方法来构造。Cohen(1989)首先提出的一种方法是：从联机的发音词典开始，使用前面介绍过的手写规则构造不同的潜在表层形式。通过计算在语料库中每个发音的出现次数可以得到概率。如果语料库的数据太稀疏，也可以通过学习每个规则的概率，并且对于每个表层形式把这些规则概率相乘而得到概率(Tajchman et al., 1995)。最后，这些加权的规则或上一节讲过的决策树，都可以自动地编到加权有限状态转录机中(Sproat and Riley, 1996)。另外，对于每个常用词，我们可以简单地从标音语料库中发现足够的发音实例，只要把这些发音实例结合起来加到一个网络中，就可以构造出模型(Wooters and Stolcke, 1994)。

上面关于*tomato*的那个网络只是为了举例说明，不是从任何实际系统中来的。图5.13是从Switchboard语料库中根据对实际发音的训练而得到的关于单词*about*的加权自动机（这些发音在第4章中已经讨论过了）。

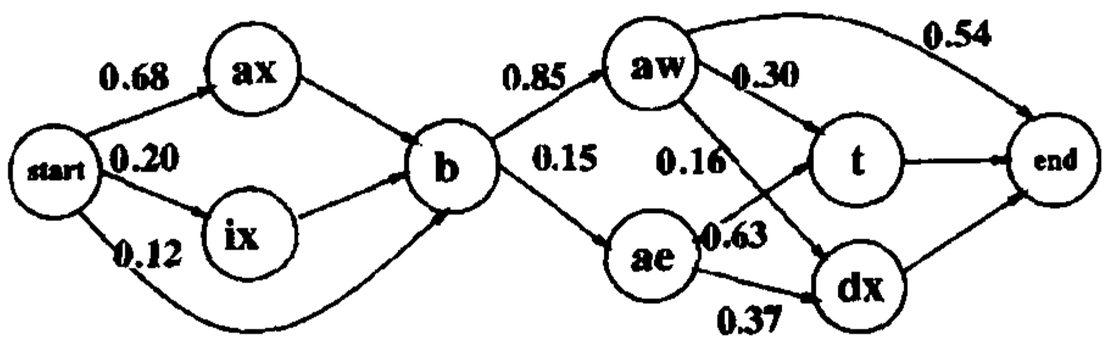


图5.13 从Switchboard语料库的实际发音中训练得到的关于单词*about*的发音网络

5.9.1 从加权自动机计算似然度：向前算法

基于自动机的词表的优点在于，这种词表有足够的算法来生成5.8节中正确单词辨别的贝叶斯模型所需的概率。这些算法可以应用于加权自动机，也可以应用于隐马尔可夫模型（将在第7章中讨论）。在关于贝叶斯方法的例子中，输入是一个音子序列[n iy]，必须在*the*, *neat*, *need*, *new*和*knee*等单词中进行选择。这种选择是通过计算两个概率来进行的，这两个概率分别是每个单词的先验概率和音子串[n iy]对每个给定单词的似然度。前面讨论这个例子时曾经说过，[n iy]对给定单词*need*的似然度是0.11，因为我们从Switchboard语料库中计算词末d脱落规则的概率是0.11。对于*need*来

说，这个概率是非常清楚而透明的，因为 *need* 只有两种可能的发音：[n iy]和[n iy d]。但是，对于像 *about* 这样的单词来说，由于这个单词具有很多不同的概率，情况就很复杂。如果使用一个事先设计好的加权自动机，通过这个自动机能够观察到不同路径的所有不同概率，就可以把这种复杂情况变得简单一些了。

有一种简单的算法可以对给定的加权自动机计算出音子串对于某一单词的似然度，这个算法称为向前算法（forward algorithm）。向前算法是自动语音识别系统（ASR）的一个重要部分，但在本章中只是非常简单地使用了这个算法，其原因是，当我们考虑某个单词并且使用加权自动机来进行计算而存在着多条路径时，使用向前算法特别有用。这种情况，本章的加权自动机是应付不了的，而应该使用第7章中的HMM来处理。此外，向前算法也是定义本章后面将要讨论的Viterbi算法的一个重要步骤。

对于似然度的计算问题，首先让我们给出加权自动机、输入和输出的形式定义。

一个加权自动机包括：

1. 状态序列 $q = (q_0\, q_1\, q_2\, \cdots\, q_n)$，每个状态对应于一个音子；
2. 状态转移概率的集合 $a_{01}, a_{12}, a_{13}, \cdots$，这个转换概率记录着从一个音子到下一个音子的概率。

我们用结点（node）来表示状态，当两个结点之间的转移概率不为零时，用两个结点之间的边（edge）来表示转移概率[①]。输入到模型中的符号序列（当我们把模型看成识别器时），或者由模型产生的符号序列（当我们把模型看成生成器时），称为观察序列（observation sequence），表示为 $O = (o_1 o_2 o_3 \cdots o_i)$（大写字母用于表示序列，小写字母用于表示序列中的单个元素）。当我们讨论加权自动机以及以后讨论 HMM 时，将会用到这些术语。

图 5.14 是带有观察序列的单词 *need* 的一个自动机。

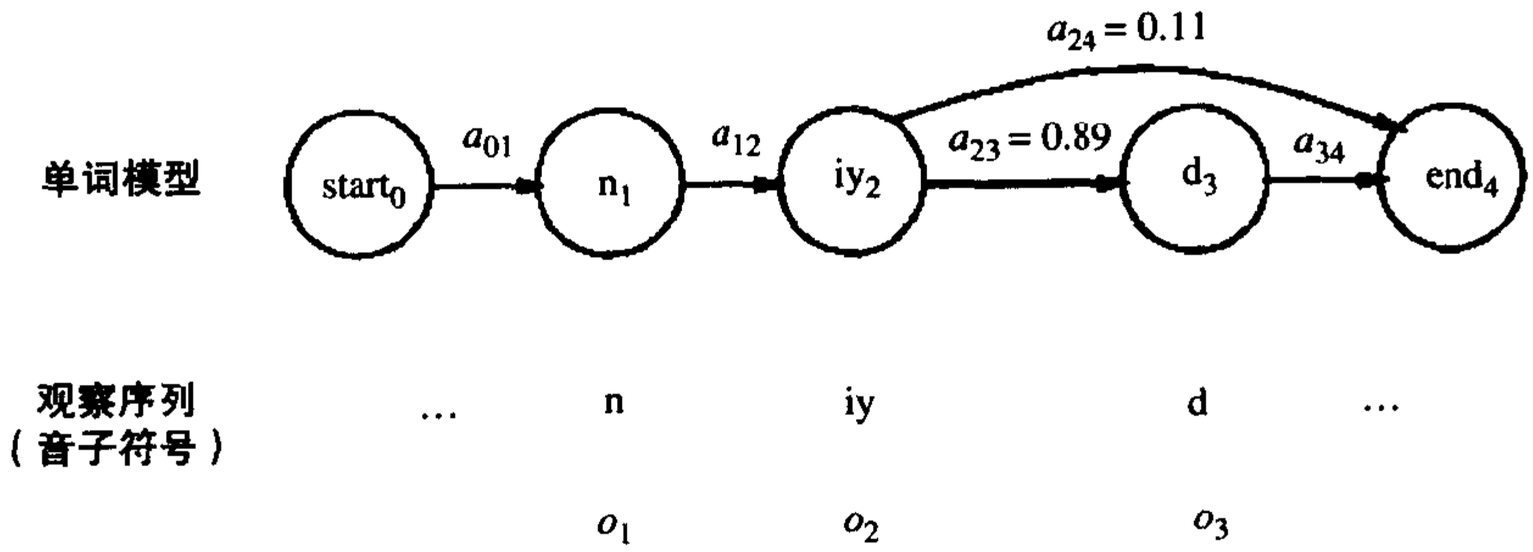


图 5.14　一个简单的用于单词 *need* 的加权自动机或马尔可夫链发音网络，其中说明了转换概率以及一个样本的观察序列。如果没有其他规定，在状态 x 和 y 之间的转移概率 a_{xy} 为 1

确定某一观察序列是由什么样的底层单词产生的问题，称为解码问题（decoding problem）。为了找出对于给定的观察序列[n iy]，哪个候选单词是概率最大的单词，我们需要对于每个候选单词（*the*, *need*, *neat*, *knee* 和 *new*）计算其乘积 $P(O|w)P(w)$，也就是说，要计算给定单词的观察序列 O 的似然度和该单词的先验概率的乘积。

---

① 我们使用两个“特殊”状态［通常称为“非发射状态”（non-emitting state）］作为初始状态和终结状态。我们也可以避免使用这样的特殊状态。在这种情况下，自动机还必须说明两个东西：
1. 状态的初始概率分布 π，使得自动机在状态 i 开始时其概率为 π_i。显而易见，如果在某些状态 j 有 $\pi_j = 0$，就意味着它们不能是初始状态。
2. 合法的接收状态的集合。

向前算法可以用来对每个单词进行这样的计算：我们给向前算法一个观察序列和某个单词的发音自动机，这个算法就可以返回一个$P(O|w)P(w)$。所以，解决解码问题的一个办法是，对于每个单词运行向前算法，并选择具有最大值的单词作为结果。正如我们在前面看到的，如果使用贝叶斯方法进行计算，在单词序列*I need*中的后一部分发音[n iy]，贝叶斯方法产生的结果是不好的（这个方法首选的单词是*new*，第二选择的单词是*neat*，第三选择的单词才是*need*）。由于向前算法也是实现贝叶斯方法的途径之一，它返回的结果与贝叶斯方法的结果是一样的，所选择单词的顺序与贝叶斯方法所得到的顺序也是一样的。在第6章中，我们将研究如何使用二元语法（bigram）的概率来提升这个算法，使这个算法能够利用“观察到[n iy]的前一个单词是*I*”这样的知识。

向前算法把每个候选单词的发音网络作为输入。因为单词*the*仅仅在鼻音后面其发音才是[n iy]，而且我们假定这个单词的实际上下文是“出现在*I*的后面”，所以在这里跳过*the*，只考虑单词*new*, *neat*, *need*和*knee*。注意，在图5.15中根据前面对于这些单词频度的计算结果在网络上增加了每个单词的概率。

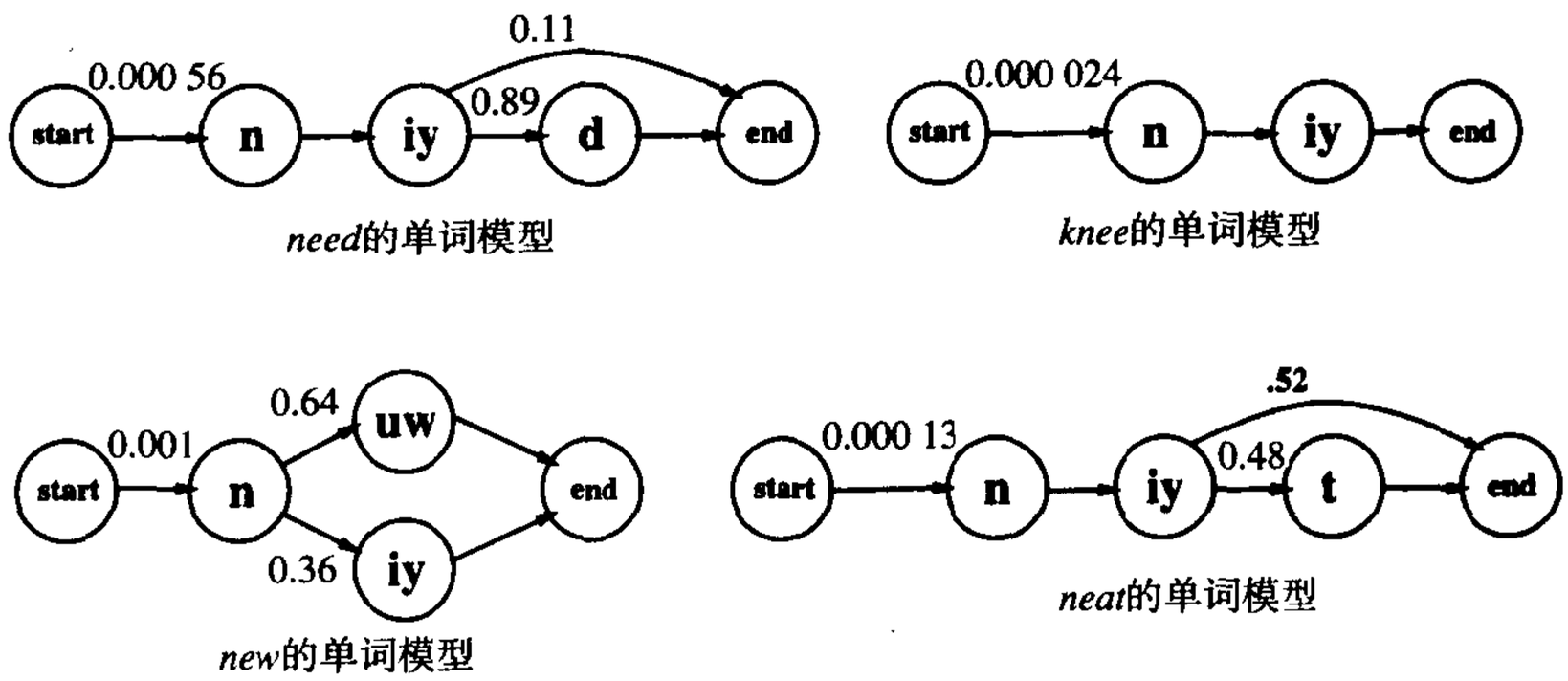


图5.15 单词*need*, *neat*, *new*和*knee*的发音网络。网络中的所有数据都是根据Switchboard语料库得出的。每个网络中还增加了相应单词的一元语法概率（即根据Switchboard语料库以及Brown语料库得到的每个单词的归一化频度）。单词的概率一般并不包括在该单词的发音网络中，这里加上单词的概率只是为了简便地讲解向前算法

向前算法是另一种动态规划算法，它可以看成是最小编辑距离算法的轻度泛化（slight generalization）。与最小编辑距离算法相同的是，当它求观察序列的概率时要使用一个表来存储中间值。与最小编辑距离算法不同的是，在列上所标记的不总是以线性顺序排列的状态，并且还隐含着一个状态图。在这个状态图中，从一个状态到另一个状态之间存在着多条路径。在最小编辑距离算法中，我们从周围的三个单元来计算每个单元的值，并把计算结果填到矩阵中。而在向前算法中，对于一个状态，可能会有其他的状态进入它当中，这样的循环递归关系就比较复杂了。另外，向前算法要计算能够产生观察的所有可能路径的概率之和，而最小编辑距离算法只需要计算概率的最小值[①]。向前算法矩阵的每个单元记为$forward[t, j]$，表示给定自动机λ在看了前面t个观察之后的状态j的概率。由于在这里使用单词的概率$p(w)$来增强状态图，在这个向前算法的例子中，计算时要把这个似然度乘以$p(w)$。计算每个单元$forward[t, j]$的值时，要把引导入这个单元的每条路径的概率加起来求和。形式地说，每个单元表示如下的概率：

① 向前算法之所以要计算概率之和，是因为当解释一个给定的观察序列时可能会有多条路径通过网络。第7章将更详细地讨论这一点。

$$forward[t, j] = P(o_1, o_2, \cdots, o_t, q_t = j|\lambda)P(w) \tag{5.14}$$

这里，$q_t = j$ 的意思是："当状态序列中的第 t 个状态是状态 j 时的概率"。我们计算这个概率的方法是把引导入当前单元的所有路径顺次延伸加起来求和。在时刻 $t-1$ 从状态 i 出发的路径顺次延伸的计算方法是把下面三个因素相乘：

1. 从前一单元开始的前面路径的概率 $forward[t-1, i]$；
2. 从前一状态 i 到当前状态的转移概率 a_{ij}；
3. 当前状态 j 与观察符号 t 匹配的观察似然度 b_{jt}。对于这里考虑的加权自动机来说，如果观察符号与状态相匹配，则 b_{jt} 为 1，否则为 0。在第 7 章中将研究更复杂的观察似然度。

算法如图 5.16 所示。

图 5.17 是把向前算法应用于单词 *need* 的实例。这个算法也可以相似地应用于可以产生符号串 [n iy]的其他单词，它们的概率在前面已经给出过。为了计算出隐藏在观察[n iy]下的最可能的底层单词，需要对每个候选单词分别运行向前算法，并挑选出概率最大的单词。第 7 章中将更加详细地给出向前算法的数学描述，并介绍有关的向前 - 向后算法（forward-backward algorithm）。

function FORWARD(*observations,state-graph*) **returns** *forward-probability*

num-states ← NUM-OF-STATES(*state-graph*)
num-obs ← *length*(*observations*)
Create probability matrix *forward*[*num-states* + 2, *num-obs* + 2]
forward[0,0] ← 1.0
for each time step *t* **from** 0 **to** *num-obs* **do**
  for each state *s* **from** 0 **to** *num-states* **do**
    for each transition *s'* from *s* specified by *state-graph*
      forward[*s'*,*t*+1] ← *forward*[*s*,*t*] * *a*[*s*, *s'*] * *b*[*s'*, o_t]
return the sum of the probabilities in the final column of *forward*

图 5.16　对于给定的单词计算观察序列的似然度的向前算法。$a[s, s']$是从当前状态 s 到下一个状态 s' 的转移概率，$b[s', o_t]$是对给定的 o_t，s' 的观察似然度。对于这里考虑的加权自动机来说，如果观察符号与状态匹配，则 $b[s', o_t]$为 1，否则为 0

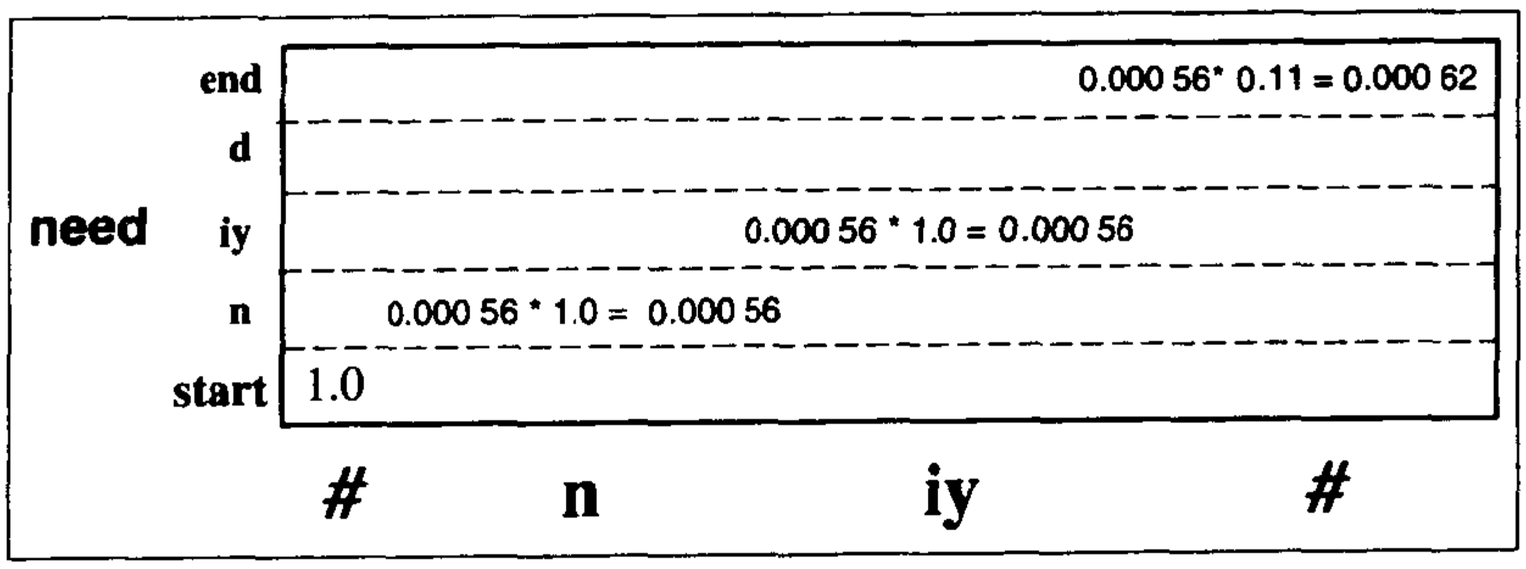


图 5.17　向前算法应用于单词 *need*，计算概率 $P(O|w)P(w)$。这个例子并不要求很强的向前算法，在第 7 章中将使用向前算法来处理更复杂的例子

5.9.2　解码：Viterbi 算法

我们在这里介绍的向前算法看起来似乎有点儿小题大做。由于通过发音网络的路径中只有一条路径将会与输入符号串相匹配，为什么要用那么大的矩阵呢？为什么要考虑那么多的可能路径呢？

另外，作为一种解码的方法，对每个单词都运行一次向前算法看起来效率也不算很高（想一想，如果不是对所有可能的单词，而是对所有可能的句子来计算似然度，那效率将会何等之低！）。向前算法之所以显得有些小题大做的部分原因，是由于以前曾经假定我们所输入的序列是没有歧义的，把发音问题看得过于简单；向前算法力图纠正这种简单化的倾向，宁愿矫枉过正。我们在第7章中将会看到，当观察序列是带噪音的声学值时，将会有很多路径通过自动机，这时如果要对这些值相加求和，向前算法将会发挥重要的作用。

但是，如果对于不同的单词分别运行向前算法，这种算法确实是一种效率非常低的解码方法。幸运的是，向前算法还有一种简单的变体，称为 Viterbi 算法，这种算法可以让我们同时考虑所有单词，并且还能计算出最佳路径。Viterbi这个术语在语音和语言处理中用得很普遍，但正如向前算法一样，Viterbi算法确实也是经典的动态规划算法的一种很标准的应用，并且它看起来很像最小编辑距离算法。Viterbi 算法首先由 Vintsyuk(1968)应用在语音识别中，并且这种算法也有像 Kruskal (1983)所说的那种“多重独立发现和发表的引人注目的历史”。Viterbi的名字在语音识别中使用得非常普遍，尽管人们也使用诸如动态规划对齐（Dynamic Programming Alignment，简称DP对齐）、动态时间偏移（Dynamic time warping）以及一遍解码（one-pass decoding）等术语。Viterbi 算法这个术语应用于单个词的加权自动机和隐马尔可夫模型的解码，也应用于连续语音等复杂问题的解码，我们将在第7章中看到它的这种应用。本章将介绍怎样使用这种算法来发现通过构成单词网络的最佳路径，从而对于给定的单词符号串观察序列选出具有最佳概率的单词。

我们在这里介绍的这种Viterbi算法取一个单独的加权自动机和所观察音子的集合$o = (o_1 o_2 o_3 \cdots o_t)$作为输入，返回具有最佳概率的状态序列$q = (q_1 q_2 q_3 \cdots q_t)$及其概率。我们可以这样来建立一个单独的加权自动机，在这个加权自动机中，用一个初始状态和一个终结状态把 *knee*, *need*, *new* 和 *neat* 这四个单词的发音网络并行地结合起来。图 5.18 说明了这些网络结合的情况。

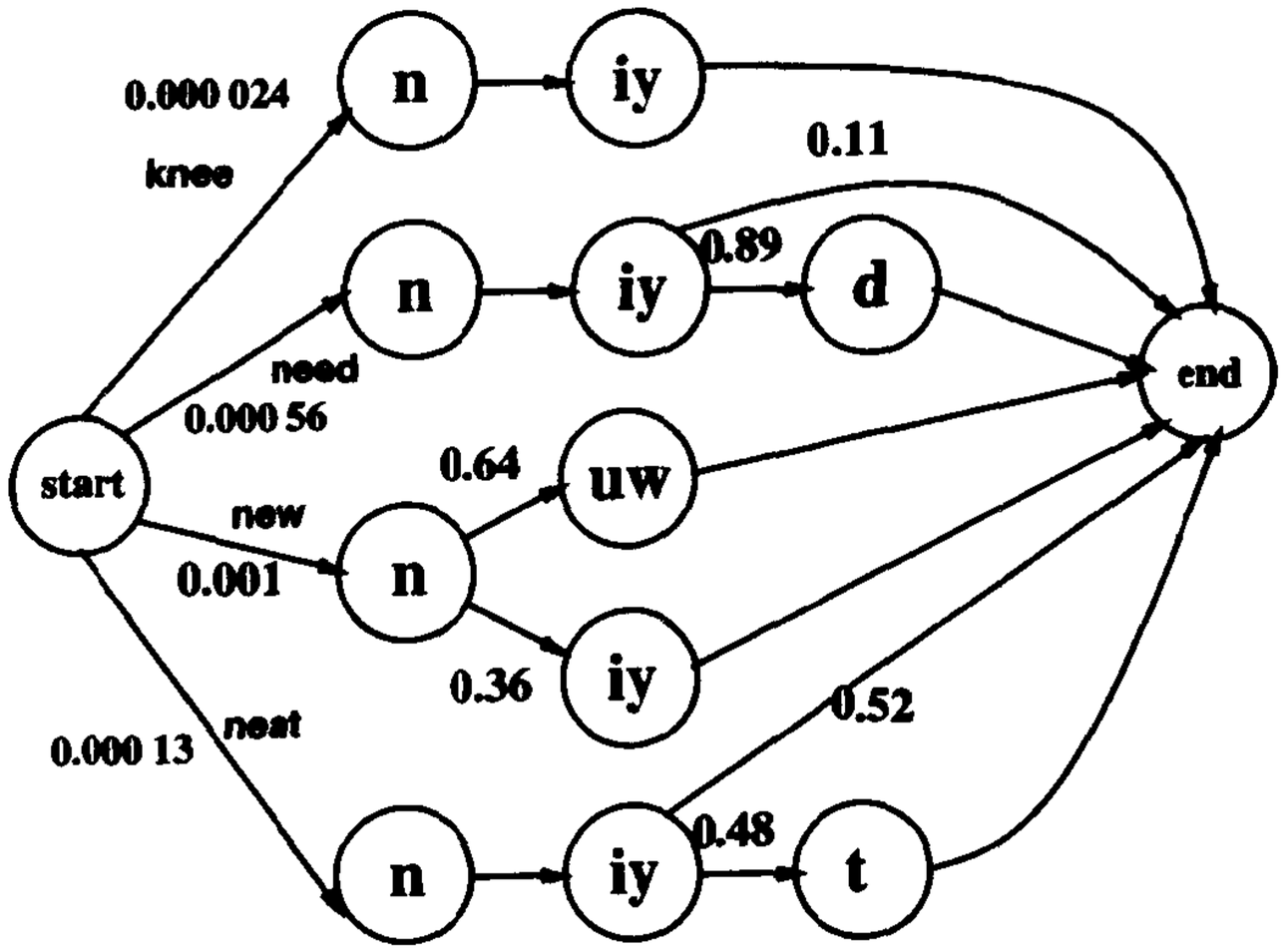


图 5.18　把单词 *need*, *neat*, *new* 和 *knee* 的发音网络结合为一个单独的加权自动机。单词的概率一般并不是该单词的发音网络的一部分，但在这里都加上了单词的概率，这样做主要是为了使我们对于Viterbi算法的解释变得简单一些

图5.19说明了这个Viterbi算法的伪代码。正如在最小编辑距离算法和向前算法中那样，Viterbi算法也建立了一个概率矩阵，矩阵的列（column）表示每个单词的索引号 t，矩阵的行表示状态图中的每个状态。正如在向前算法中那样，对于把这4个单词结合起来的这个单独的自动机中的每个

状态 q_i，每个列都有一个单元。事实上，Viterbi算法中的代码与向前算法中的代码一样，只不过是做了上述的两个修改而已。向前算法把前面所有路径的总和放到当前单元中，而 Viterbi 算法则把前面所有路径中最大的放到当前单元中。

算法首先建立N+2或四个状态列。第一列是初始的伪观察值，第二列代表第一个观察音子[n]，第三列代表[iy]，第四列代表最后的伪观察值。开始时，在第一列中置开始状态的概率为为1.0，置其他概率为0；读者可以参看图5.20。为便于阅读，概率为0的单元都简单地用空白表示。

然后，正如在向前算法中那样，我们转移到下一个状态。对于列0中的每个状态，计算转移到列1中的每个状态的概率。取引导入当前单元的所有路径的延伸中最大的值为 *viterbi*[t, j]的值。在时刻 $t-1$ 从状态 i 出发的一条路径的延伸，只要把在向前算法中使用的三个因素相乘即可计算得到：

1. 从前一单元开始的前面路径的概率 *forward*[$t-1$, i]；
2. 从前一状态 i 到当前状态 i 的转移概率 a_{ij}；
3. 当前状态 j 与观察符号 t 匹配的观察似然度 b_{jt}。对于在这里考虑的加权自动机来说，如果观察符号与状态相匹配，则 b_{jt} 为 1，否则为0。在第 7 章中将研究更复杂的观察似然度。

```
function VITERBI(observations of len T,state-graph) returns best-path

  num-states ← NUM-OF-STATES(state-graph)
  Create a path probability matrix viterbi[num-states+2,T+2]
  viterbi[0,0] ← 1.0
  for each time step t from 0 to T do
    for each state s from 0 to num-states do
      for each transition s' from s specified by state-graph
        new-score ← viterbi[s, t] * a[s,s'] * b_s'(o_t)
        if ((viterbi[s',t+1] = 0) || (new-score > viterbi[s', t+1]))
          then
                viterbi[s', t+1] ← new-score
                back-pointer[s', t+1] ← s
  Backtrace from highest probability state in the final column of viterbi[] and
return path
```

图 5.19　在连续语音识别中发现最优状态序列的Viterbi算法，为简单起见，使用音子作为输入。给出音子的观察序列和加权自动机（状态图），算法返回具有最大概率的自动机的路径，并且接收观察序列。$a[s, s']$表示从当前状态 s 到下一个状态 s' 的转移概率，$b[s', o_t]$表示对于给定的 o_t，s' 的观察似然度。对于这里考虑的加权自动机，如果观察符号与状态匹配，$b[s', o_t]$为 1，否则为0

在图 5.20 中，在输入 n 这一列，每个单词都以[n]开头，因此在状态为 n 的单元内，每个单元都具有非零概率。在这一列中的其他单元的项目为零，因为它们的状态与 n 不匹配。当我们向前进入下一列时，每个与*iy*匹配的单元的值等于前一个单元的内容乘以从前一单元到这一单元的转移概率。这样，单词 *new* 的状态 *iy* 的 *viterbi*[2, iy_{new}]值等于 *new* 的“单词概率”乘以 *new* 在这个位置的发音为元音 *iy* 的概率。注意，如果只是看 *iy* 这一列，单词 *need* 就是当前具有“最大概率”的单词。但是，如果继续向前移动，进入最后一列，取得最后胜利的将是单词 *new*，因为 *need* 进入 *end* 的转移概率最小（等于0.11），而 *new* 进入 *end* 的转移概率为 1.0。现在我们可以跟随返回指针，回溯并找到最后概率为 0.000 36 的路径，从而判断单词 *new* 取得最后胜利。

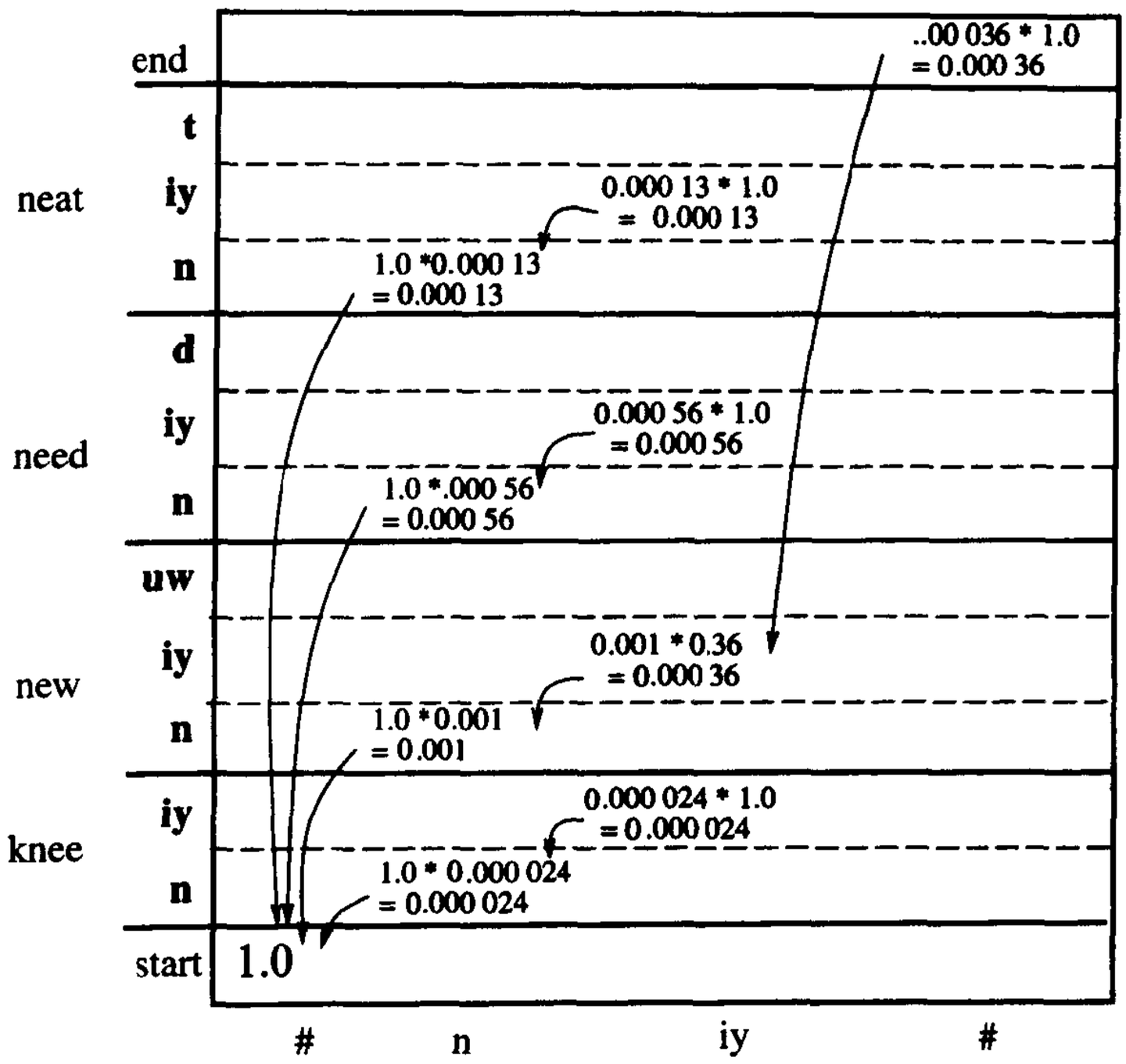


图 5.20 Viterbi 算法在单独的状态列中的各个项目。每个单元保持了在当前情况下的最好路径的概率以及沿着这条路径指向前一个单元的指针。从最后状态 *end* 回溯，可以重建达到最好的单词 *new* 的状态序列 $n_{new}\ iy_{new}$

5.9.3 加权自动机和切分

加权自动机和 Viterbi 算法在切分（segmentation）的各种算法中起着重要的作用。所谓“切分”就是把一个连续的符号序列分割成“语块”的处理问题。例如，句子切分（sentence segmentation）就是在语料库中自动找出句子边界的问题。类似地，单词切分（word segmentation）就是在语料库中找出单词边界的问题。在书面英语中，把单词彼此切分开没有什么困难，因为在单词之间有正词法规定的空白。但是，在使用汉字书写系统的汉语和日语中，情况并非如此。书面汉语没有单词界限的标志。每个汉字前后相连，彼此之间没有空白。由于每个汉字大体上相当于一个语素，所以单词可以由一个或多个汉字构成。要知道单词可以在什么地方被切分开来，通常是十分困难的。但是，在很多应用问题中，特别是在自动剖析和文本-语音转换的应用中，正确地进行单词切分是非常必要的。一个句子按不同的方式切分成一些什么样的单词，往往会影响到它的读音。

我们来考虑 Sproat et al.(1996)研究的如下例子：

日文章鱼怎么说?
“How do you say ‘octopus’ in Japanese?” (5.15)

这个句子存在着两种不同的切分，但只有其中的一个是正确的。似乎有道理的切分是把前两个汉字结合起来构成单词“日文”（**rì-wén**，声调符号表示音节的声调），然后把后两个汉字结合起来构成单词“章鱼”（**zhāng-yú**）。

日文	章鱼	怎么	说	?
rì-wén	zhāng-yú	zěn-me	shuō	
Japanese	octopus	how	say	(5.16)

"How do you say octopus in Japanese?"

日	文章	鱼	怎么	说	?
rì	wén-zhāng	yú	zěn-me	shuō	
Japan	essay	fish	how	say	(5.17)

"How do you say Japan essay fish?"

Sproat et al.(1996)提出了一种非常简单的算法，选取包含最高频度的单词的切分为正确的切分。换言之，这种算法把每种潜在切分中的每个单词的概率相乘，从中选取概率乘积最大的切分。

在实现这个算法时，要把表示汉语词表的加权有限状态转录机和Viterbi算法结合起来。这个汉语词表是一个稍微增强了的有限状态转录机词表（FST-lexicon，简称FST词表），关于有限状态转录机，在第4章中已经介绍过；FST词表中的每个单词表示为弧的一个序列，序列中的每个弧代表该单词中的一个汉字。正如概率算法中通常的做法，这个算法实际上使用了单词的负对数概率$-\log(P(w))$。对数概率是很有用的，因为很多概率的乘积值太小了，使用对数概率可以避免数值下溢。使用对数概率还意味着我们是用成本相加（adding cost）来代替概率相乘（multiplying probability），要找出"最小成本的解"（minimum cost solution）来代替"最大概率的解"（maximum probability solution）。

下面来研究图5.21中的例子。图5.21(a)是一个样本词表，只包含五个潜在的单词：

单词	发音	含义	成本 $(-\log p(w))$
日文	rì-wén	'Japanese'	10.63
日	rì	'Japan'	6.51
章鱼	zhāng- yú	'octopus'	13.18
文章	wén-zhāng	'essay'	9.51
鱼	yú	'fish'	10.28

系统把输入的句子表示为一个没有加权的有限状态自动机FSA，如图5.21(b)所示。为了把输入句子和词表结合起来，需要把这个有限状态自动机FSA转换成有限状态转录机FST。算法使用一个函数*Id*，这个函数*Id*取一个FSA *A*，并且把它转换成FST，这个FST把能够而且只能够被*A*接收的所有符号串映射为它们本身。设*D**表示D的转换闭包，也就是说，通过加圈形成一个自动机，每个圈从词表的终点返回到词表的起点，每加一个单词就加一个圈。所有可能的切分的集合是$Id(I)\circ D^*$，也就是说，把输入转录机$Id(I)$和词表*D*的转换闭包*D**结合起来，形成有限状态转录机FST，如图5.21(c)所示。然后，从中选出最好的切分，最好的切分就是$Id(I)\circ D^*$中成本最低的切分，如图5.21(d)所示。

应用Viterbi算法，可以很容易地找出最好的路径，如图5.21(d)所示。读者可以自己作为练习做一做。另外，这个切分算法以及前面看到的错拼更正算法，也可以加以扩充，把单词之间的概率（*N*元语法概率）也考虑进去。我们将在第6章介绍*N*元语法。

5.9.4　用切分来进行词表的自动归纳

上面介绍的加权自动机的切分算法要依靠存储在词表中的加权值。但是，这个词表怎样才能首先学习到呢？很多切分算法都提到关于没有词表的切分这样的"先验"问题。例如，de Marcken (1996)和Brent and Cartwright(1996)分别提出的算法，取没有切分过的输入音子序列，应用信息论

的原理，试验各种可能的切分，经过反复迭代，从切分结果中归纳出词表。他们的算法都依赖于最小描述长度（Minimum Description Length，简称MDL）原则的一个随机版本和音位组配转移概率，在不同的模型之间进行选择。词表或语法的描述长度（例如，测量其中的符号数目）是词表的信息复杂度的试探性测度。由于总是优先选择符号较少的词表，MDL就等于默认了它宁愿选择比较简单而更一般化的词表。Brent and Cartwright(1996)假定儿童使用MDL切分，通过从话语中切分单词进行学习，从而建立一个词表。Saffran et al.(1996)说明了，8个月的婴儿能够使用音子序列概率作为单词切分的一种根据。

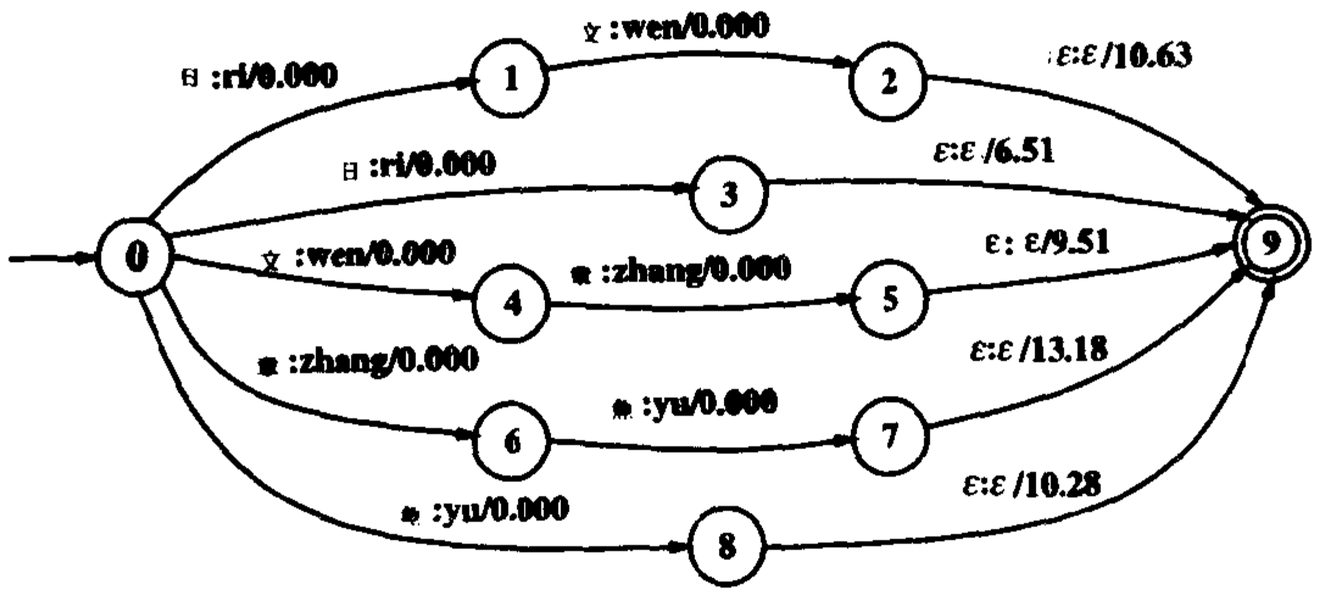


图 5.21 把 Sproat 等的算法应用于五个输入单词(Sproat et al, 1996)

5.10 人类发音研究

在5.7节中，我们讨论了影响人的发音变异的很多因素。本节将总结性地简单介绍人类如何从心理词典中检索单词的一个计算机模型，这种单词检索机制是人类词汇产生机制的一个部分。这个模型来自 Gary Dell 和他的同事们，为简洁起见，我们把多个模型的特征加以总结和简化(Dell, 1986, 1988; Dell et al., 1997)，把它们合在一起进行简单回顾。首先来考虑某些数据。在第 3 章中

① Dell(1988)还提出了第四层，即音节结构层，这里没有涉及该层。

讲过，舌头打滑这样的口误的产生（把 *barn door*[谷仓门]发音成 *darn bore*[织补洞]）常常可以给我们提供关于词汇产生机制的重要线索。Dell(1986)总结了关于口误研究的一些前期成果。其中主要有“词汇倾向效应”（lexical bias effect）和“重复音位倾向效应”（repeated phoneme bias）。词汇倾向效应使得口误造成的错误总是单词错误而不是非词错误。例如，*dean bad* → *bean dad* 这样的口误是 *deal back* → *beal dack* 这样的口误的三倍。重复音位倾向效应指的是，两个单词中有两个音子，当这两个单词中存在着一个相同的音子时，就更容易发生口误。例如，*deal beack* 比 *deal back* 更容易口误为 beal。

Dell(1986, 1988)提出的模型是一个具有三个层次的网络：语义层、单词层（词目层）和音位层，[1]

在语义层有概念结点，在词目层每个单词有一个结点，在音位层每个音有若干个分割开的结点，一个音可以分割为音节头、元音和音节尾。每个词目结点与构成该词的音位单元相连接，同时也与表示概念的语义单元相连接。这种连接用来从一个结点到另一个结点进行激活（activation），它是双向的，又是非常敏感的。词汇产生的过程分为两个阶段。在第一个阶段，从语义概念到单词进行激活。激活过程将一层一层地降到音位单元，然后又转回到其他的词汇单元。在某个点把最高激活的单词选出来。在第二个阶段，这个被选出来的单词要经过很大的激活波动。这个激活过程又再次通过音位层。这时，具有最高激活度的音位结点被选出来，并且按顺序进行访问。

图 5.22 是 Dell 的模型。舌头打滑的错误之所以发生，是因为过多的激活达到错误音位结点。例如，词汇倾向的发生可以这样来模拟：当激活从欲说单词的音子传到相邻的单词时，它就把这些单词的音子激活。这样，当这些音子出现在实际的单词中时，不正确的音子往往会被“过度”激活。

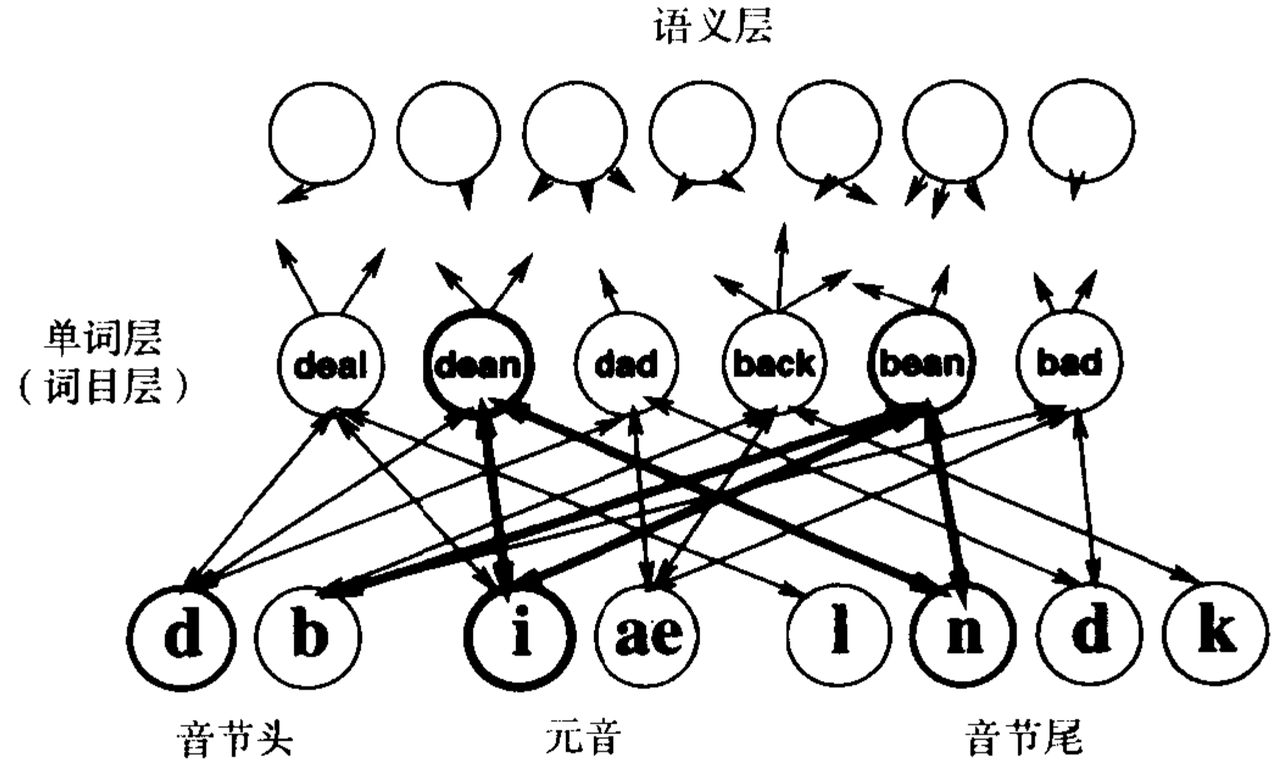


图 5.22　说明词汇倾向机制的 Dell 网络模型(1986, 1988) [根据 Dell（1988, p.134 做了修改）]。粗线结点表示有很多激活的结点。欲说的单词 *dean*（院长）在很多场合由于舌头打滑而口误为 *bean*（豆），因为存在着结点 *bean*。粗箭头表示引起可能的口误的连接

这种分为两个阶段的网络模型也可以扩充来解释关于词汇产生过程的其他事实。失语症患者在语言产生过程和理解过程中也会有各种麻烦。Dell et al.(1997)表明，在上述网络模型中如果减弱不同的连接，也可能会引起失语症中的不同语言错误。这样的事实支持“连续性假说”（continuity hypothesis）。这种假说认为，失语症的某些部分只不过是在单词检索中的正常困难的连续扩充，因而它们也可以为网络模型提供进一步的证据。对网络模型的细节有兴趣的读者可以参阅有关的参考材料和有关的计算模型。例如，Roelofs (1997)把网络模型扩充来处理音节化、语音编码和更复杂的序列结构问题。读者也可以参阅 Levelt et al.(1999)。

5.11 小结

本章介绍了几个重要的比喻和算法，它们在语音和语言处理中将非常有用。本章要点总结如下：

- 我们可以把很多语言问题表示为：当一个整齐而正确的符号串通过噪声信道时被干扰而受到损坏，我们的任务是设法恢复符号串的本来面目。恢复原来符号串的一个有力手段是研究原来符号串的一切可能结果，按照它们的条件概率加以排序。
- 条件概率一般使用贝叶斯规则就可以很容易地计算出来。贝叶斯规则把概率分解为先验概率和似然度。在错拼更正或发音模型中，先验概率可以通过单词的频度或单词的二元语法频度来计算。似然度可以通过在数据库中训练简单的概率模型（例如，含混矩阵、决策树或手写规则）来计算。
- 计算两个符号串之间的距离的任务来自错拼更正或其他问题。最小编辑距离算法是动态规划技术在解决这种问题时的一个应用。最小编辑距离算法可以用来产生两个符号串之间的距离，或者用来对齐这两个符号串。
- 单词的发音是有很多变异的。发音变异的原因有两个：词汇变异和音位变异。词汇变异包含社会语言学的因素，例如方言、语域或风格。
- 影响音位变异的一个最重要的因素是周围音子的辨识。其他重要因素还有音节结构、重音模式、单词的辨识和频度。
- 解码的任务实质上是这样一个问题：发现和确定那些生成观察符号“噪声”序列的正确的底层符号序列。
- 向前算法是计算在给定的加权自动机中观察序列似然度的最有效的途径。正如最小编辑距离算法一样，向前算法也是动态规划的一种变体。在第7章中，当我们研究隐马尔可夫模型时，向前算法将特别有用，因为它可以容许我们对于具有同样观察序列的多条路径进行求和。
- Viterbi 算法也是动态规划的一种变体。当我们考虑所有可能的符号串，使用贝叶斯规则来计算它们生成所观察到的“噪声”序列时，Viterbi 算法是解决这种解码问题的有效途径。
- 在诸如汉语和日语这些没有词界标志的语言中，单词的切分是另一种类型的最优化问题，这类问题可以用 Viterbi 算法来解决。

5.12 文献和历史说明

检查和更正拼写错误的算法至少从Blair(1960)开始就有了。大多数早期算法都是基于关键相似度的，例如 Soundex 算法(Odell and Russell, 1922; Knuth, 1973)。Damerau (1964)给出了一个基于词典的算法来进行拼写错误的检查；从此以后，大多数错误检查算法都是基于词典的。Damerau 还提出了一种单一错误的更正算法。从 Wagner and Fischer(1974)开始，大多数算法都依赖于动态规划。Kukich(1992)是一篇关于拼写错误检查和更正的综述文章。对于非OCR文本拼写错误，更正的概率算法流行得比较晚，例如 Kashyap and Oommen (1983)和 Kernighan et al.(1990)。

相比之下，光学字符识别领域中的概率算法发展得比较早；Bledsoe and Browning(1959)研制了用于 OCR 错拼更正的一种概率算法，这种算法使用一部大型词典，把单词中的每个字母的似然度相乘，计算出在词典中给定的每个单词和所观察字符序列之间的似然度。在这个意义上，Bledsoe 和 Browning 已经预见到现代贝叶斯方法应该与语音识别相结合了。Shinghal and Toussaint(1979)以及 Hull and Srihari(1982)使用二元语法的字母转移概率和 Viterbi 算法，在错误拼写的 OCR 输入中选择可能性最大的正确形式。

把动态规划应用于序列比较，就出现了Kruskal(1983)指出的“多重独立发现和发表的值得注意的历史现象”①。

Kruskal 等给出了这种算法在四个不同领域中独立发表的事例，如下所示：

引证发表人	领域
Viterbi(1967)	信息论
Vintsyuk(1968)	语音识别
Needleman and Wunsch(1970)	分子生物学
Sakoe and Chiba(1971)	语音识别
Sankoff(1972)	分子生物学
Reichert et al.(1973)	分子生物学
Wagner and Fischer (1974)	计算机科学

根据语音和语言处理术语的标准用法，当把动态规划应用于任何种类的概率最大化问题时，使用Viterbi这个术语；对于非概率问题，经常使用动态规划这5个普通的术语。向前算法是从隐马尔可夫模型推导出来的，关于向前算法的历史将在第 7 章中进行总结性的介绍。Sankoff and Kruskal (1983)收集了不同领域中关于序列比较研究的理论和应用的探索情况。Forney(1973)是一篇综述文章，在信息和通信理论的背景下介绍了 Viterbi 算法的来龙去脉。

加权有限状态自动机是首先由Pereira et al.(1994)进行描述的，这种描述来自Booth and Thompson(1973)把有限状态转录机的工作和概率语言的工作结合起来的研究。

① 这里举出的七项已经很可观了。参见 1.6 节中关于多重发现的论述。

第6章　N元语法

But it must be recognized that the notion "probability of a sentence" is an entirely useless one, under any known interpretation of this term.

Noam Chomsky (1969, p. 57)[1]

Anytime a linguist leaves the group the recognition rate goes up.

Fred Jelinek (then of the IBM speech group) (1988)[2]

有一本书，名为M*A*S*H，这本书同时也被拍成了电影和电视，其中有一位称为Radar O'Reilly的人，他是M*A*S*H单位的第4077号职员，举止文雅，办事规矩，他有一种离奇的本领，能够猜到和他说话的人将要讲什么。我们当中的大多数人都不会有这种本领，只有当猜测那些非常了不起的抒情诗人写的歌词中某段歌词的下一个词时，也许我们有可能猜出来。或者，有时我们也会做这样的猜测，例如猜测哪个单词最可能跟在下面的句子片断后面：

I'd like to make a collect...

你们当中的大多数人也许会认为，最可能的单词是*call*，尽管下一个词也可能是*telephone*，*person-to-person*或*international*。想一想还有其他的可能吗？这个事实给我们的启示在于，猜测单词并不是那么不可思议的事，至少它并不像我们原来想像的那样要求那么精确和完善。为什么猜测单词这么重要呢？因为猜测下一个单词，或称单词预测（word prediction），是语音识别、手写识别、残疾人增强交际以及拼写错误更正的一个重要子任务。在这样的任务中，单词辨认是很困难的，因为输入会有很多噪声和歧义。因此，查看前一个单词即可提供关于下一个单词将会是什么的重要线索。Russell and Norvig(1995)曾经举过一个在*Take the Money and Run*（携款潜逃）中的例子。一位银行出纳员读懂了Woody Allen抢劫来的一张写得乱七八糟的票据时对他说*I have a gub*。语音识别系统（或者某个人）可以根据其对于单词序列的知识（*a gub*不是英语的一个单词序列）以及概率知识而处理好这样的问题。例如，在抢劫这样的特殊环境下，*I have a gun*（我有一只枪）要比*I have a gub*或*I have a gull*（我有一个海鸥）的概率高得多。

预测下一个单词的这种能力对于残疾人增强交际系统是至关重要的（Newell et al., 1998)。已经有一些计算机系统可以在交际方面帮助残疾人。例如，对于那些不能使用口语或手势语言来交际的残疾人，如著名物理学家霍金（Steven Hawking），就可以使用增强交际系统来帮他们说话，让他们通过手的简单动作来选择单词，把这些单词拼写出来，或者从可能单词的选项中把需要的单词挑出来。但是，这时的拼写速度是很慢的，而且单词选项也不可能包含英语中所有可能的单词。在这种情况下，如果能够知道哪些单词是说话人下一步最想使用的，就可以把这些单词放到选项中。

---

① 然而应当认识到，"句子的概率"这个概念，在任何已知的对于这个术语的解释中，都是一个完全无用的概念。

② 每当一个语言学家离开我们的研究组，语音识别率就提高一步。
这是Jelinek于1988年12月7日在自然语言处理评测讨论会上的讲话。Palmer and Finin(1990)描述这个讨论会时没有写下这段引文；一些当时参加会议的人回忆，Jelinek讲的话更为尖刻，他说："每当我解雇一个语言学家，语音识别系统的性能就会改善一些。"

最后，让我们来考虑真词拼写错误的问题。在真实英语的单词中会出现拼写错误（尽管写作的人并不想这样做），所以真词错误的检查很困难（我们无法仅仅通过查词典的方式发现真词错误）。图 6.1 给出了一些例子。

They are leaving in about fifteen *minuets* to go to her house.
The study was conducted mainly *be* John Black.
The design *an* construction of the system will take more than a year.
Hopefully, all *with* continue smoothly in my absence.
Can they *lave* him my messages?
I need to *notified* the bank of [this problem.]
He is trying to *fine* out.

图 6.1　摘自 Kukich(1992）的真词拼写错误

这些错误都可以通过算法检查出来，这时算法需要在其他特征中查看真词错误周围的单词。例如，短语 *in about fifteen minuets* 在英语语法上是无懈可击的，但是单词与单词之间的结合情况却很少见[①]。这时拼写检查程序能够检查出这种低概率的结合关系。在上面的例子中，三个单词的结合概率也很低（*they lave him, to fine out, to notified the*）。当然，在没有拼写错误的句子中，也会出现一些概率很低的单词序列，这种情况是对于错拼检查研究的一个挑战。在 6.6 节中将会看到，各种不同的机器学习算法使用了周围单词的信息和其他特征来进行上下文敏感的拼写错误检查（context-sensitive spelling error correction）。

对下一个单词的猜测与另外一个问题有着密切的联系，这个问题就是单词序列概率的计算。例如，下面的单词序列在英语的书面文本中的出现概率不为零：

... all of a sudden I notice three guys standing on the sidewalk taking a very good long gander at me.
（突然间，我注意到有三个站在人行道旁边的男人老是在盯着我。）

但是，如果把同样的这些单词按另外的顺序随便编排一下，其出现概率就变得非常低了。

good all I of notice a taking sidewalk the me long three at sudden guys gander on standing a a the very

给一个句子分配概率的算法也可以用来给下一个单词在一个不完整的句子中分配概率。在下面几章将会看到，整个句子的概率或单词串的概率对于词类标注（part-of-speech-tagging）（见第 8 章）、词义排歧（word-sense disambiguation）和概率剖析（probabilistic parsing）（见第 12 章）都是非常有用的。

我们在本章中将要介绍的单词预测模型是 *N* 元语法模型（*N*-gram）。*N* 元语法模型利用前面 *N*–1 个单词来预测下一个单词。在语音识别中，传统上使用语言模型（language model）或 LM 这个术语来称呼单词序列的统计模型。在本章的其他地方，我们根据不同的上下文分别使用语言模型（language model）或语法（grammar）这两个术语。

6.1　语料库中单词数目的计算

[前面问道："是不是英语中没有足够的单词供他使用"]
"不，单词是足够的，但是它们不都是很恰当的。"
James Joyce, 在 Bates 的报告(1997)

① minuets 的意思是"小步舞"，与"in the fifteen"的结合概率很低——译者注。

概率的计算依赖于对事物的计算。在我们谈论概率之前，需要决定所要计算的是什么以及到哪里找东西来计算。

我们在第5章中说过，自然语言的统计处理要依赖于语料库（corpora，它的单数形式是corpus），语料库是联机的文本或口语的集合体。为了计算单词的概率，需要计算训练语料库中的单词数目。让我们来看一看 Brown 语料库，这是一个规模为 100 万单词的语料库，样本来自 500 篇书面文本，包括不同的类别（新闻、小说、非小说、学术著作等）。这个语料库是由 Brown 大学在 1963 年至 1964 年收集的(Kučera and Francis, 1967; Francis, 1979; Francis and Kučera, 1982)。式（6.1）是 Brown 语料库中的一个句子，这个句子中有多少单词呢?

He stepped out into the hall, was delighted to encounter a water brother. （6.1）

如果我们不把标点符号也算成单词，这个句子就有13个单词；如果我们把标点符号也计算在内，这个句子就有 15 个单词。是否把点号（“.”）和逗号（“,”）算为单词，取决于不同的任务。对于诸如语法检查、拼写错误检查、作者辨认这样的任务，标点符号的位置是很重要的（例如，检查句子开头的专有名词是不是大写字母，查找标点符号的特殊使用模式以便惟一地辨认某个作者）。在自然语言处理的应用中，问号是辨别某人提问的重要线索。在词类标注中，问号也是一个有用的线索。因此，在这些应用中经常把标点符号看成单词。

口语语料库与文本语料库不同，口语语料库通常没有标点符号，但口语语料库有另外一些现象，需要我们决定是不是把它们算为单词。Switchboard 是一个关于陌生人之间电话会话的口语语料库，这个语料库是 20 世纪 90 年代初期收集的，包含 2430 个会话，每个会话平均 6 分钟，总共有 240 小时的会话，包含 300 万单词(Godfrey et al., 1992)。下面是 Switchboard 中的一个话段样本，因为口语的单位不同于书面语的单位，当我们谈到口语问题时将使用“话段”（utterance）这个术语而不使用“句子”这个术语。

I do uh main-mainly business data processing （6.2）

与口语中的多数话段一样，这个话段中存在着“片断”（fragment）。“片断”是一个单词在中间被拦腰切开而形成的，例如，上面句子中的 mainly 被拦腰切开形成的“main-”就是片断。此外还有像 *uh* 这样的有声停顿（filled pause），它在书面英语中是不存在的。我们在这里是不是把它们看成单词呢？这也取决于应用的具体情况。如果在自动语音识别的基础上建立一个自动听写系统，我们宁愿把“片断”从话语中剔除。但是，像 *uhs* 和 *ums* 这样的有声停顿，事实上更倾向于被当成单词来处理。例如，Smith and Clark(1993)以及 Clark(1994)曾经指出，*um* 与 *uh* 的意思稍有不同。一般说来，当说话人已经胸有成竹而要说一个话段时，他就 *um*；而当说话人想说但是还没有找到恰当的单词来表达时，就用 *uh*。Stolcke and Shriberg(1996b)也发现，*uh* 可以用来作为预测下一个单词的线索（请读者想一想这是为什么？），因此很多语音识别系统都把 *uh* 当成一个单词来处理。

同一个单词的大写词例 *They* 和非大写词例 *they* 又怎样处理呢？对于大多数统计应用来说，是把它们混在一起来对待的，尽管在某些时候也把大写作为个别的单独特征来处理（在拼写错误更正或词类标注中）。在本章的其他地方假定我们的模型是不区分大小写的。

我们应该如何处理像 *cats* 对 *cat* 这样的单词的屈折形式呢？这也取决于应用的情况。当前大多数基于 *N* 元语法的系统都是以“词形”（wordform）为基础的，所谓“词形”就是在语料库中以屈折形式出现的单词形式。所以，*cat* 和 *cats* 要分别处理为两个单词。对于很多领域来说，这种办法不是一种很好的简化处理办法。在很多领域中，我们想把 *cat* 和 *cats* 看成同一个抽象单词的实例，

或者称为“词目”(lemma)的实例。一个词目是具有同一词干、同一主要词类、同一词义的词汇形式的集合。在第16章中，我们再讨论词形(词形把*cat*和*cats*区分开来)和词目(词目把*cat*和*cats*混在一起处理)的区别问题。

在英语中有多少单词呢？回答这个问题的一个办法是在语料库中进行计算。我们用“型”(type)来表示语料库中不同单词的数目，也就是词典容量的大小；用“例”(token)来表示使用中的单词数目。这样，下面来自Brown语料库的句子有16个单词“例”和14个单词“型”(不计算标点符号)。

They picnicked by the pool, then lay back on the grass and looked at the stars. (6.3)

(他们在池塘旁边野炊，然后躺在草地上观看星空。)

Switchboard语料库有240万个词形的“例”和大约两万个词形的“型”，其中包括专有名词。口语中的词汇不如书面语丰富。Kučera(1992)对莎士比亚(Shakespeare)的全部著作进行过计算，其中词形的“例”有884 647个，词形的“型”有29 066个。因此，884 647个词形的“例”是29 066个词形的“型”的重复使用。Brown语料库中的100万个词形的“例”包含61 805个词形的“型”，这些“型”属于37 851个词目。这些语料库都太小了。Brown et al.(1992)积累了一个58 300万个词形的“例”的英语语料库，包含293 181个词形的“型”。

词典是用来计算单词数目的另一种途径。词典中不包含有屈折变化的形式，用词典来计算词目比计算词形更方便。美国Heritage词典第三版有20万条“黑体形式”，这个数目比词目的数目高一些，因为一个词目可能会有一个或多个黑体形式(并且黑体形式包括多词短语)。

在本章的其他部分将继续区分“型”和“例”。“型”是指词形的“型”而不是词目的“型”，并且标点符号一般也算为单词。

6.2 简单的(非平滑的)N元语法

本章研究的单词序列模型是概率模型。概率模型是给单词的符号串指派概率的方法，不论是计算整个句子的概率，还是在一个序列中预测下一个单词的概率，都要使用概率模型。正如在第5章中那样，我们假定读者具备概率论的基本知识。

最简单的单词序列的概率模型是单纯地假定语言中的任何一个单词后面可以跟随该语言中的任何一个单词。在这种理论的一个概率版本中，假定任何一个单词后面可能跟随的该语言中的任意其他单词的概率是相等的。如果英语中有100 000个单词，那么任何一个单词后面跟随其他任何单词的概率将是1/100 000，即0.000 01。

在稍微复杂一些的单词序列模型中，任何一个单词后面可以跟随着其他任何单词，但后面一个单词要按照它正常的频度来出现。例如，单词*the*的频度相对比较高，在1 000 000个单词的Brown语料库中它出现69 971次(也就是说，在这个特定的语料库中，有7%的单词是*the*)。相比之下，单词*rabbit*在Brown语料库中只出现11次。

我们可以根据这样的相对频度对下面将要出现的单词指派一个概率分布的估值。这样，如果我们看到了任何符号串，就可以指派概率0.07给*the*，指派概率0.000 01给*rabbit*，从而猜测下面一个单词。例如，假定我们看到了如下的符号串：

Just then, the white

在这个上下文中，跟随着单词 *white* 之后，*rabbit* 似乎是一个比 *the* 更合理的单词。这说明，我们不是简单地看单词的单独相对频度，而是要看单词对于给定的前面一个单词的条件概率。也就是说，当前面是单词 *white* 时 *rabbit* 的概率（我们把这个条件概率表示为 *P*(*rabbit*|*white*)）要高于当前面是其他单词时 *rabbit* 的概率。

根据这样的直觉，让我们来研究怎样计算一个完整的单词串的概率（我们把这个单词串表示为 $w_1 \cdots w_n$，或者表示为 w_1^n）。如果把每个单词在它本身的位置的出现看成一个独立事件，就可以把这种概率表示如下：

$$P(w_1, w_2, \cdots, w_{n-1}, w_n) \tag{6.4}$$

我们也可以使用概率的链式规则来分解这个概率：

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\cdots P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \end{aligned} \tag{6.5}$$

然而，我们怎样才能计算出概率 $P(w_n|w_1^{n-1})$ 呢？当前面给定的单词序列很长时，我们不知道用什么简单方法来计算这时一个单词的概率是多少。例如，我们不能在一个很长的符号串之后，计算每个单词的出现次数，这时需要非常大的语料库。

我们通过一个有用的简化方法来解决这个问题：对于给定的前面的所有单词来逼近一个单词的概率。我们使用的逼近方法很简单：只需要计算当前面给定的单词只是一个单独的单词时，单词的概率是多少！这样的“二元语法模型”（bigram model）通过前面一个单词的条件概率 $P(w_n|w_1^{n-1})$ 来逼近前面给定的所有单词的概率 $P(w_n|w_{n-1})$。换言之，我们不是计算概率

$$P(\text{rabbit}|\text{Just the other day I saw a}) \tag{6.6}$$

而是使用如下的概率来逼近这个概率：

$$P(\text{rabbit}|\text{a}) \tag{6.7}$$

一个单词的概率只依赖于它前面单词的概率的这种假设称为马尔可夫假设。马尔可夫模型是一种概率模型，假设不必查看很远的过去就可以预见到某个单位将来的概率。我们在第5章中介绍马尔可夫链时就已经知道了这个术语的这种用法。我们曾经说过，马尔可夫链是一种加权有限状态自动机；在马尔可夫链中，马尔可夫这个术语的直觉含义就是：加权 FSA 的下一个状态总是依赖于它前面有限的历史（因为在有限自动机中的状态数目总是有限的）。基本的二元语法模型可以看成是每个单词只有一个状态的马尔可夫链。

我们可以把二元语法模型（只看过去的一个单词）推广到三元语法模型（看过去的两个单词），再推广到 *N* 元语法模型（看过去的 *N*–1 个单词）。二元语法模型称为一阶马尔可夫模型（因为它只看过去的一个词例），三元语法模型称为二阶马尔可夫模型，*N* 元语法模型称为 *N*–1 阶马尔可夫模型。直到 1958 年 Chomsky 发表对 Skinner 的 Verbal Behavior（言语行为）评论之前，单词的马尔可夫模型在工程技术、心理学和语言学中得到了普遍应用（参见本章后面的“文献和历史说明”这一节），但是后来不再流行了。IBM 公司的 Thomas J. Watson 研究中心语音识别实验室应用 *N* 元语法模型在语音识别中取得了很大的成绩，马尔可夫模型又重新引起了学术界的注意。

在一个序列中，*N* 元语法对于下一个单词的条件概率逼近的通用等式是：

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1}) \tag{6.8}$$

式（6.8）说明，对于所有给定的前面的单词，单词w_n的概率可以只通过前面N个单词的概率来逼近。

对于二元语法来说，我们把式（6.8）代入式（6.5），就可以计算出整个符号串的概率。结果如下：

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1}) \tag{6.9}$$

让我们来看语音理解系统中的一个例子。Berkeley Restaurant Project（Berkeley饭店规划）是一个基于语音的饭店咨询系统，用户可以通过这个系统询问关于California州Berkeley饭店的问题，系统从地方饭店的数据库中检索合适的信息显示给用户(Jurafsky et al., 1994)。这里是用户提问的一些样本：

I'm looking for Cantonese food.
（我在找广东菜的饭店。）
I'd like to eat dinner someplace nearby.
（我喜欢在附近的地方吃晚餐。）
Tell me about Chez Panisse.
（请告诉我关于 Chez Panisse饭店的情况。）
Can you give me a listing of the kinds of food that are available?
（你可以给我已经准备好的各种食品的清单吗？）
I'm looking for a good place to eat breakfast.
（我正在找一个适合吃早饭的地方。）
I definitely do not want to have cheap Chinese food.
（我确实不想吃便宜的中国食品。）
When is Caffe Venezia open during the day?
（近来 Venezia 咖啡店什么时候开门？）
I don't wanna walk more than ten minutes.
（走 10 分钟以上的地方我不想去。）

图6.2中的表是关于二元语法概率的一个样本，它说明了在单词*eat*之后可能出现的某些单词的概率，这些概率是从用户所说的句子中统计得出的（现在我们不考虑训练二元语法概率的算法）。注意，这些概率编码说明了某些事实，这些事实是：在本质上很严格的句法事实（在*eat*之后常常会是一个名词短语的开头，例如形容词、修饰词或名词等）以及某些与文化有关的事实（在英国询问如何找英国食品的概率是很低的）。

eat on	0.16	eat Thai	0.03
eat some	0.06	eat breakfast	0.03
eat lunch	0.06	eat in	0.02
eat dinner	0.05	eat Chinese	0.02
eat at	0.04	eat Mexican	0.02
eat a	0.04	eat tomorrow	0.01
eat Indian	0.04	eat dessert	0.007
eat today	0.03	eat British	0.001

图6.2　Berkeley Restaurant Project中说明*eat*后最容易出现的单词的二元语法的一个片断

除了图6.2的表中的概率之外，我们的语法还包括如图6.3所示的二元语法概率（<s>是一个特殊的单词，它的意思是“句子的开始”）。

<s> I	0.25	I want	0.32	want to	0.65	to eat	0.26	British food	0.60
<s> I'd	0.06	I would	0.29	want a	0.05	to have	0.14	British restaurant	0.15
<s> Tell	0.04	I don't	0.08	want some	0.04	to spend	0.09	British cuisine	0.01
<s> I'm	0.02	I have	0.04	want thai	0.01	to be	0.02	British lunch	0.01

图6.3 Berkeley Restaurant Project中关于二元语法的更多片断

现在我们可以计算句子*I want to eat British food*或句子*I want to eat Chinese food*的概率了，计算时只要把相邻两个单词的二元语法概率相乘即可，如下所示：

$$\begin{aligned} P(\text{I want to eat British food}) = {} & P(\text{I}|<\text{s}>)P(\text{want}|\text{I})P(\text{to}|\text{want}) \\ & P(\text{eat}|\text{to})P(\text{British}|\text{eat}) \\ & P(\text{food}|\text{British}) \\ & = 0.25*0.32*0.65*0.26*0.002*0.60 \\ & = 0.000\,016 \end{aligned}$$

我们可以看出，由于概率都小于1（根据概率的定义），相乘的概率越多，所有概率的乘积就越小，这样就会有数值下溢的危险。如果要计算一个相当长的符号串的概率（例如，一段文字或者一篇文件），习惯上就采用对数空间来进行计算，我们给每个概率取对数（称为对数概率logprob），把所有的对数相加（因为在对数空间中的加与在线性空间中的乘是等价的），然后再取结果的反对数。由于这个原因，事实上已经存储了很多关于*N*元语法的标准算法，并且所有概率都是用对数概率来计算的。在这段文本中，我们取的对数都是以2为底数的（也就是说，我们用log来表示$\log_2$）。

三元语法模型与二元语法模型相同，不过这时要用前面两个单词作为条件。例如，用*P*(*food*|*eat British*)来替代*P*(food|British)。为了计算每个句子开头的三元语法概率，可以使用两个假想的单词（pseudo-word）作为三元语法的条件，即*P*(*I*|<*start1*><*start2*>)，其中*start1*和*start2*是位于句子开头的假想的单词。

*N*元语法模型可以使用训练语料库和归一化的方法得到。对于概率模型来说，所谓归一化（normalizing），就是用某个总数来除，使最后得到的概率的值处于0和1之间，以保持概率的合法性。我们取某个训练语料库，从这个语料库中取某个特定的二元语法的计数（即出现次数），然后用第一个单词相同的二元语法的总数作为除数来除这个计数：

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_w C(w_{n-1}w)} \tag{6.10}$$

我们可以把这个等式加以简化，因为以给定单词w_{n-1}开头的所有二元语法的计数必定等于该单词w_{n-1}的一元语法的计数。读者可以想一想以便确信这样的结论。

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \tag{6.11}$$

对于一般的*N*元语法，参数估计为：

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})} \tag{6.12}$$

在式（6.12）中，用前面第一个单词的观察频度来除这个特定单词序列的观察频度，就可以得到*N*元语法概率的估计值。这个比值称为相对频度（relative frequency）。在最大似然估计（Maximum

Likelihood Estimation，简称MLE）技术中，相对频度是概率估计的一种方法，因为对于给定的模型M来说，最后算出的参数集能使训练集T的似然度，即$P(T \mid M)$达到最大值。例如，在容量为1 000 000个单词的Brown语料库中，假定单词*Chinese*出现了400次。那么，在另一个容量为1 000 000个单词的文本中，单词*Chinese*的出现概率是多少呢？MLE可以估计出，其概率也是$\frac{400}{1000000}$或0.0004。现在，0.0004并不是在一切情况下单词*Chinese*出现的概率估计值，但这个概率能使我们估计出，在容量为1 000 000个单词的语料库中，*Chinese*这个单词最可能出现的次数大约是400次。

除了使用相对频度来估计N元语法概率的方法之外，还有更好的方法（我们将在6.3节中讨论一种重要的算法）。但是，在其他方法中，使用相对频度这种思路的算法复杂得多。图6.4是从Berkeley Restaurant Project中得到的一个二元语法的某些二元语法计数。注意，大多数的计数为零。实际上，我们选择这7个单词样本时已经设法尽量使它们彼此接应得比较好；如果随机地选择7个单词，数据将更加稀疏。

	I	want	to	eat	Chinese	food	lunch
I	8	1087	0	13	0	0	0
want	3	0	786	0	6	8	6
to	3	0	10	860	3	0	12
eat	0	0	2	0	19	2	52
Chinese	2	0	0	0	0	120	1
food	19	0	17	0	0	0	0
lunch	4	0	0	0	0	1	0

图6.4 在Berkeley Restaurant Project语料库（容量约为10 000个句子）中，从1616个单词的“型”中选出7个单词的二元语法计数

图6.5是经过归一化之后的二元语法概率（用下列每个单词相应的一元语法计数来除各自的二元语法计数）。

I	3437
want	1215
to	3256
eat	938
Chinese	213
food	1506
lunch	459

7个单词的二元语法概率如下：

	I	want	to	eat	Chinese	food	lunch
I	0.0023	0.32	0	0.0038	0	0	0
want	0.0025	0	0.65	0	0.0049	0.0066	0.0049
to	0.00092	0	0.0031	0.26	0.00092	0	0.0037
eat	0	0	0.0021	0	0.020	0.0021	0.055
Chinese	0.0094	0	0	0	0	0.56	0.0047
food	0.013	0	0.011	0	0	0	0
lunch	0.0087	0	0	0	0	0.0022	0

图6.5 在Berkeley Restaurant Project语料库（容量约为10 000个句子）中，从1616个单词的“型”中，选出7个单词的二元语法概率

6.2.1 N元语法及其对训练语料库的敏感性

在本节中将研究不同的N元语法模型的一些例子，以便从直觉上了解这种模型的两个重要事实。第一个事实是，当增加N的值时N元语法模型的精确度也相应地增加。第二个事实是，N元语法性能强烈地依赖于训练它们的语料库（特别是语料库的种类和单词的容量）。

我们采用Shannon(1951)提出并同时由Miller and Selfridge(1950)使用过的直观化(visualization)技术，从直觉上来了解这些事实。我们的基本想法是，首先训练各种N元语法，然后用它们来随机地生成句子。在一元语法的场合，要直观地看到其工作的情况非常简单。我们假定，英语中所有单词覆盖的概率空间在0和1之间，我们在0和1之间选择一个随机数，然后把覆盖所选实际值的单词打印出来。同样的技术也可以用来生成阶数更高的N元语法，首先根据二元语法的概率从<s>开始生成一个随机的二元语法，然后接着这个二元语法再选择一个随机的二元语法(下面一个二元语法的似然度与它的条件概率是成比例的)，依次类推。

为了使我们对于N元语法的能力随着它的阶数的增高而增高这一事实有直观的了解，我们在莎士比亚全集的语料库上分别训练一元语法、二元语法、三元语法和四元语法模型。然后使用训练出来的这四个语法生成随机的句子。在下面的例子中，我们把每个标点符号也看成一个单词。而且，当使用语料库来训练语法时，把所有大写字母都改成小写字母。在生成句子之后，为了便于阅读，再把有关的小写字母改成大写字母。这样生成的一些句子如下：

1. 用一元语法来逼近莎士比亚

(a) To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have

(b) Every enter now severally so, let

(c) Hill he late speaks; or! a more to leg less first you enter

(d) Will rash been and by I the me loves gentle me not slavish page, the and hour; ill let

(e) Are where exeunt and sighs have rise excellency took of. Sleep knave we. near; vile like

2. 用二元语法来逼近莎士比亚

(a) What means, sir. I confess she? then all sorts, he is trim, captain.

(b) Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.

(c) What we, hath got so she that I rest and send to scold and nature bankrupt, nor the first gentleman?

(d) Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt

(e) Thou whoreson chops. Consumption catch your dearest friend, well, and I know where many mouths upon my undoing all but be, how soon. then; we'll execute upon my love's bonds and we do you will?

(f) The world shall- my lord!

3. 用三元语法来逼近莎士比亚

(a) Sweet prince, Falstaff shall die. Harry of Monmouth's grave.

(b) This shall forbid it should be branded, if renown made it empty.

(c) What is't that cried?

(d) Indeed the duke; and had a very good friend.

(e) Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

(f) The sweet! How many then shall posthumus end his miseries.

4. 用四元语法来逼近莎士比亚

(a) King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;

(b) Will you not tell me who I am?

(c) It cannot be but so.

(d) Indeed the short and the long. Marry, 'tis a noble Lepidus.

(e) They say all lovers swear more performance than they are wont to keep obliged faith unforfeited!

(f) Enter Leonato's brother Antonio, and the rest, but seek the weary beds of people sick.

训练模型的上下文越长，句子的连贯性越好。在一元语法生成的句子中，单词与单词之间没有接应和连贯关系。我们可以看到，在一元语法生成的句子中，没有一个句子是以句号或其他可以作为句末标点的符号结尾的。在二元语法生成的句子中，单词与单词之间只存在着非常局部的接应和连贯关系。三元语法和四元语法生成的句子，看起来已经似乎是莎士比亚的句子了。当然，仔细地查看一下四元语法生成的句子，可以看出它们更像莎士比亚的句子。*It cannot be but so* 这几个词，就是直接从 *King John* 那里来的。这是因为，尽管莎士比亚的著作有很多不同的标准版本，但是其总词数不会多于 100 万单词。前面说过，Kučera(1992)曾经计算过莎士比亚全集的词数，出现单词数为 884 647 个（“例”），不同单词数为 29 066 个（“型”，包括专有名词）。这意味着，即使是二元语法模型，其数据也非常稀疏；从 29 066 个不同的单词（“型”）可以形成 $29\,066^2$ 个，即 84 400 万个以上的二元语法关系。在这种情况下，用 1 000 000 单词的训练集来估计那些不常见单词的频度，显然非常不充分。实际上，在莎士比亚著作中，不同的二元关系类型不会超过 300 000 个。莎士比亚著作的规模如果用来训练四元语法那就更小了，因此我们的生成系统对于前面头4个词的四元语法（*It cannot be but*），下面可能接续的单词只有 5 个（*that, I, he, thou* 和 *so*）；对于很多包含 4 个单词的四元语法，它们的接续单词都只有 1 个。

为了研究语法对于它的训练集的依赖关系，我们用一个完全不同的语料库来训练*N*元语法。这个语料库是华尔街日报语料库（Wall Street Journal，简称 WSJ）。一个以英语为母语的人能够读莎士比亚和华尔街日报，两者都是英语的一个子集。从直觉上来说，我们也许会觉得莎士比亚的*N*元语法将会与华尔街日报的*N*元语法互相重叠，彼此覆盖。为了检验这种感觉是否正确，这里有三个句子，是根据每天从华尔街日报文章的 4000 万单词的语料库中训练出来的一元语法、二元语法和三元语法生成的。这些语法是经过 Good-Turing 平滑的 Katz 回退语法（Katz backoff grammar），下一节将学习如何建造这种语法。在下面生成的句子中，为了便于阅读，我们用手工把英语的专有名词的首字母改为了大写字母。

1.（一元语法）Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

2.（二元语法）Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

3.（三元语法）They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

把这些句子与前面那些所谓莎士比亚的句子相比较；表面上看来，二者似乎都想模拟“像英语的句子”，但显而易见的是，二者的句子之间没有重叠覆盖的现象。即使在一个很小的短语中出现

重叠覆盖，这种重叠和覆盖也非常小。莎士比亚语料库和华尔街日报语料库之间的这种差异告诉我们，为了很好地在统计上逼近英语，需要一个规模很大的语料库，这个语料库应该包容不同的种类，并且覆盖不同的领域。尽管这样，像*N*元语法这样的简单统计模型也没有能力模拟不同种类的不同风格。当我们阅读莎士比亚的著作时，只想看见莎士比亚的句子，而思想不会跳到华尔街日报文章中去。

方法论探讨：训练集和测试集

正如*N*元语法一样，统计模型中的概率来自训练它的语料库。这个训练语料库（training corpus）需要精心地设计。如果训练语料库太偏向于某个任务或某个领域，概率就可能太窄，对于新的句子缺乏一般性。如果训练语料库太泛，概率就不能充分地反映有关的任务或领域的特点。

另外，假定我们试图计算某一特定的“测试”句子的概率。如果“测试”句子是训练语料库的一部分，它将会具有人为拔高的概率。训练语料库不应该因为包含了这个句子而发生偏移。因此，当我们对于具有相关数据的某个给定语料库应用语言的统计模型时，一开始就要把数据分为训练集（training set）和测试集（test set）。我们在训练集上训练统计参数，然后使用这些参数计算测试集中的概率。

这种把数据分为训练集和测试集的方法，也可以用来评估不同*N*元语法的总体结构。例如，为了比较6.3节中将要介绍的各种平滑算法（smoothing algorithm），可以使用一个很大的语料库，并把它分为训练集和测试集。然后，在训练集上训练两个不同的*N*元语法模型，再看看哪个*N*元语法模型能比较好地模拟测试集。但是，“模拟测试集”是什么意思呢？对于统计模型与测试集匹配的情况，有一个有用的度量方法，称为困惑度（perplexity）。困惑度是熵（entropy）的一个变体，我们将在后面介绍它。

在某些场合，我们需要一个以上的测试集。例如，假定我们有若干个不同的语言模型，我们想首先挑一个最好的模型，然后看这个模型在某个合理的测试集上运行的情况，但是，在此之前我们没有见过它，不知道它是不是最好的。我们首先使用调试测试集（development test set，又称为devtest）来挑选最好的语言模型，这时可能要调试某些参数。当我们认为模型已经调试成最好的模型时，就可以在真正的测试集上运行它。

在比较模型的时候，使用统计测试（在统计课程或社会科学的统计学教材中介绍的统计测试）来决定两个模型之间的差异是否有意义，这是很重要的。Cohen(1995)是一本重点介绍人工智能中的统计研究方法的有用参考书。Dietterich(1998)重点介绍了用于比较分类器的统计测试方法。

6.3 平滑

标准的*N*元语法模型的一个主要问题在于，这种模型必须从某些语料库训练得到，而每个特定的语料库都是有限的，因此能够接受英语的*N*元语法的完美语料库都注定会忽略了一些东西。这意味着，从任何训练语料库得到的二元语法矩阵都是稀疏的（sparse），它们注定会存在着大量的公认为“零概率二元语法”的场合，当然实际上它们也会真的具有某些非零的概率。此外，当非零计数很小时，MLE方法也会产生很糟糕的估计值。

这个问题的某些部分是*N*元语法所特有的，因为它们不能使用长距离的上下文，总是倾向于过低地估计那些在训练语料库中不是彼此邻近出现的符号串的概率。不过，我们可以使用一些技术给那些“零概率的二元语法”指派非零概率。这种给某些零概率和低概

Never do I ever want
to hear another word!
There isn't one,
I haven't heard!
Eliza Doolittle in
Allan Jay Lerner's *My Fair Lady* lyrics[①]

words people
never use —
could be
only I
know them
Ishikawa Takuboku 1885–1912[②]

① 译文：从来我不想，去听其他词！凡我未听者，全都不是词。
② 译文：世人从不用的词，可能只有我知道。

率的N元语法重新赋值并给它们指派非零值的工作称为“平滑”。在下面几节中将介绍某些平滑算法，并说明怎样使用它们来修改图6.5中的二元语法概率。

6.3.1 加1平滑

一种简单的平滑方法是，取二元语法的计数矩阵，在把它们归一化为概率之前先给所有的计数加1。这种算法称为“加1平滑”。虽然这种算法的效果不是很好，并且也不经常使用，但它引入了其他平滑算法中将使用的许多概念，并且这种算法还可以使我们对平滑算法有最基本的认识。

让我们首先来研究加1平滑对于一元语法概率的应用，因为这样会简单一些。非平滑的一元语法概率的最大似然度估计是用单词的“例”的总数N去除单词数来计算的：

$$P(w_x) = \frac{c(w_x)}{\sum_i c(w_i)} = \frac{c(w_x)}{N}$$

不同的平滑可以依赖于一个可调整的数c^*来估算。对于加1平滑来说，这个数c^*的调整可以通过在词数上加1然后乘以一个归一化因子$\frac{N}{N+V}$来实现，其中V是该语言中单词的“型”的总数，称为词汇容量（vocabulary size）。因为我们对于每个单词的“型”数都加了1，所以单词的“例”的总数将随着“型”的数目的增加而增加。这样，加1平滑的调整数可以定义为：

$$c_i^* = (c_i + 1)\frac{N}{N+V} \tag{6.13}$$

这个数可以用N来归一化，然后转变为概率p_i^*。

还有一种不同的方法是把平滑算法看成打折（discounting），也就是把某个非零的数降下来，使得到的概率量可以指派给那些为零的数。因此，很多文章不提打折数c^*，而用折扣d_c来定义平滑算法，折扣d_c等于打折数c^*与原数c之比：

$$d_c = \frac{c^*}{c}$$

与此不同，我们也可以直接从单词数来计算概率p_i^*：

$$p_i^* = \frac{c_i + 1}{N+V}$$

现在我们对于一元语法的情况已经在直觉上有了一些认识。让我们来平滑Berkeley Restaurant Project中的二元语法。图6.6说明了图6.4中的二元语法的加1平滑数。

	I	want	to	eat	Chinese	food	lunch
I	9	1088	1	14	1	1	1
want	4	1	787	1	7	9	7
to	4	1	11	861	4	1	13
eat	1	1	3	1	20	3	53
Chinese	3	1	1	1	1	121	2
food	20	1	18	1	1	1	1
lunch	5	1	1	1	1	2	1

图6.6 在Berkeley Restaurant Project语料库中（容量约为10 000个句子），从1616个单词的“型”中选出7个单词的加1平滑二元语法计数

图6.7说明了对于图6.5中的二元语法的加1平滑概率。我们记得，正规的二元语法概率是用一元语法数去归一化每行的词数而计算出来的：

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \tag{6.14}$$

对于加1平滑二元语法数，我们首先需要用词汇中的所有单词的“型”数V来提升一元语法数：

$$p^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V} \tag{6.15}$$

我们需要把V（=1616）加到每个一元语法数上：

I	3437 + 1616 = 5053
want	1215 + 1616 = 2931
to	3256 + 1616 = 4872
eat	938 + 1616 = 2554
Chinese	213 + 1616 = 1829
food	1506 + 1616 = 3122
lunch	459 + 1616 = 2075

经过平滑化的二元语法概率的结果如图6.7所示：

	I	want	to	eat	Chinese	food	lunch
I	0.0018	0.22	0.00020	0.0028	0.00020	0.00020	0.00020
want	0.0014	0.00035	0.28	0.00035	0.0025	0.0032	0.0025
to	0.00082	0.00021	0.0023	0.18	0.00082	0.00021	0.0027
eat	0.00039	0.00039	0.0012	0.00039	0.0078	0.0012	0.021
Chinese	0.0016	0.00055	0.00055	0.00055	0.00055	0.066	0.0011
food	0.0064	0.00032	0.0058	0.00032	0.00032	0.00032	0.00032
lunch	0.0024	0.00048	0.00048	0.00048	0.00048	0.00096	0.00048

图6.7 在Berkeley Restaurant Project语料库中（容量约为10 000个句子），从1616个单词的“型”中选出7个单词的加1平滑二元语法概率

最方便的办法是重新建立一个计数矩阵，使我们能够清楚地看出平滑算法怎样改变原来的计数。这个调整计数可以用式（6.13）来计算。图6.8说明了这些重新建立的计数。

	I	want	to	eat	Chinese	food	lunch
I	6	740	0.68	10	0.68	0.68	0.68
want	2	0.42	331	0.42	3	4	3
to	3	0.69	8	594	3	0.69	9
eat	0.37	0.37	1	0.37	7.4	1	20
Chinese	0.36	0.12	0.12	0.12	0.12	15	0.24
food	10	0.48	9	0.48	0.48	0.48	0.48
lunch	1.1	0.22	0.22	0.22	0.22	0.44	0.22

图6.8 在Berkeley Restaurant Project语料库中（容量约为10 000个句子），从1616个单词的“型”中选出7个单词的加1平滑二元语法计数

注意，加 1 平滑使原来的计数发生了很大改变。*C*(*want to*)从 786 改变为 338。在概率空间中，可以看出也发生了很大改变：*P*(*to* | *want*)从没有平滑时的 0.65 下降到平滑后的 0.28。

我们再来看折扣*d*（新计数与旧计数之间的比值）。折扣*d*说明，二元语法中前面单词的计数的改变之大令人吃惊；以*Chinese*开头的二元语法的折扣因子竟然为8（1829/213=8，213/1829=0.12）！

I	0.68
want	0.42
to	0.69
eat	0.37
Chinese	0.12
food	0.48
lunch	0.22

单词的计数和概率之所以发生这样大的改变，是因为很多概率量被转移到为零的那些项目当中去了。问题在于，我们随便地把“1”这个值加到每个计数上。如果把比较小的值加到这些计数上（“加一半”，“加千分之一”），就可能避免这样的问题，但这样一来就要重新对于每种情况来训练这些参数。

总而言之，加 1 平滑是一种很糟糕的算法。Gale and Church(1994)总结了加 1 平滑的各种问题，主要的问题是加 1 平滑在预测带零计数的二元语法的实际概率时，与其他平滑方法（例如下面将要介绍的 Good-Turing 方法）相比显得非常差。另外，他们还说明，加 1 平滑方法所产生的各种计数实际上比没有平滑的 MLE 方法还差。

6.3.2　Witten-Bell 打折法

比较好一些的平滑算法是 Witten-Bell 打折法（Witten-Bell Discounting），这种方法与加 1 平滑算法相比只是稍微复杂一些，在 Witten and Bell(1991)中把这种方法称为方法 C。Witten-Bell 打折法的基本根据是一种简单而聪明的对于零频度事件的直觉。让我们把一个零频度单词或*N*元语法看成是刚才没有发生的事件。如果这个事件要发生，那么它将是这个新的*N*元语法中我们首次看到的事件。因此，看一个零频度*N*元语法的概率就可以用首次看一个*N*元语法的概率来模拟。这就是统计语言处理中“再发生”（recurring）的概念。

关键概念 #4　看过一次的事物（Things Seen Once）：使用刚才已经看过一次的事物的数量来帮助估计从来没有见过的事物的数量。

借助于“看过一次的事物”的数量来估计“从来没有见过的事物”的概率，当本章后面讨论“Good-Turing平滑”时还要再来研究这个概念；并且，当在第8章中讨论未知词的词类标注方法时，也要再讨论到这个概念。

我们怎样计算首次看到*N*元语法的概率呢？可以通过在训练语料库中计算首次看到*N*元语法的次数来计算。产生这样的计数是非常简单的，因为“首次”*N*元语法的计数恰恰就是在该数据中看到的*N*元语法的“型”的计数（因为首次看的时候必须每个“型”都恰巧只看一次）。

这样，我们就可以通过用“例”的数加上所观察的“型”的数来除“型”的数，估计所有的零*N*元语法的全部概率量。

$$\sum_{i:c_i=0} p_i^* = \frac{T}{N+T} \tag{6.16}$$

为什么我们要用“例”的数加上“型”的数来进行归一化呢？我们可以把训练语料库想像成一序列事件，一个事件表示每个“例”，一个事件表示每个新的“型”。这样，式（6.16）可给出一个新的“型”事件发生的最大似然估计。注意，所观察的“型”T的计数不同于在加1平滑中所用的“全部型”或“词汇容量”V；T是我们已经看到的“型”，而V是我们可能看到的“型”的全部计数。

式（6.16）给出全部的“未看见的N元语法的概率”。我们需要在所有为零的N元语法中分摊它。我们可以只选择等分的方法来分摊它。设Z是具有零数的N元语法的全部计数（应该是“型”，这里没有任何“例”）。前面的每个为零的一元语法现在都可以同等地共享这个已经重新分布的概率量：

$$Z = \sum_{i:c_i=0} 1 \tag{6.17}$$

$$p_i^* = \frac{T}{Z(N+T)} \tag{6.18}$$

如果零的N元语法的所有概率用式（6.16）来计算，那么外加的概率必定会从某个地方产生出来；我们使用给所有看到的N元语法的概率打折的方法来得到这些外加的概率：

$$p_i^* = \frac{c_i}{N+T} \quad \text{若}(c_i > 0) \tag{6.19}$$

换言之，我们可以把平滑了的计数直接表示如下：

$$c_i^* = \begin{cases} \frac{T}{Z}\frac{N}{N+T}, & \text{若} c_i = 0 \\ c_i\frac{N}{N+T}, & \text{若} c_i > 0 \end{cases} \tag{6.20}$$

Witten-Bell打折法在一元语法时很像加1平滑，但如果把这个等式扩充到二元语法，就会看到区别很大。这是因为在二元语法时“型”的计数是以前面的历史为条件的。为了计算未看过的二元语法$w_{n-1}w_{n-2}$的概率，我们使用“看以w_{n-1}开头的新的二元语法的概率”。这就使得要用单词的历史来说明对“首次二元语法”的估计。与那些杂乱无章的单词比起来，那些倾向于以很小数值的二元语法出现的单词将提供一个较低的二元语法估值。

在前面单词为w_x以二元语法的“型”的数目为T、而且二元语法的“例”的数目为N的条件下，我们把这个事实用公式表示为：

$$\sum_{i:c(w_xw_i)=0} p^*(w_i|w_x) = \frac{T(w_x)}{N(w_x)+T(w_x)} \tag{6.21}$$

像一元语法的情况一样，我们也需要把这个概率量在所有未见的二元语法中进行分摊。这里，我们再设Z是具有零计数的给定首词的二元语法的全部计数（应该是“型”，这里没有任何“例”）。每个前面的为零的二元语法现在都同等地共享这个重新分布的概率量：

$$Z(w_x) = \sum_{i:c(w_xw_i)=0} 1 \tag{6.22}$$

$$p^*(w_i|w_{i-1}) = \frac{T(w_{i-1})}{Z(w_{i-1})(N+T(w_{i-1}))} \quad \text{若}(c_{w_{i-1}w_i} = 0) \tag{6.23}$$

对于非零的二元语法，采用同样的方式，引入历史参数T来打折：

$$\sum_{i:c(w_xw_i)>0} p^*(w_i|w_x) = \frac{c(w_xw_i)}{c(w_x)+T(w_x)} \tag{6.24}$$

为了用式（6.24）来平滑图6.5中的关于饭店的二元语法，对于每个首次看到的单词，我们都需要二元语法“型”$T(w)$的数值，它们的数值如下：

I	95
want	76
to	130
eat	124
Chinese	20
food	82
lunch	45

此外，对于这里的每个词，我们还需要Z的值。因为我们知道在词汇表中有多少个单词（$V = 1616$），对于以给定单词w开头的可能的二元语法恰恰有V个，所以，对于给定的前面的单词的未见二元语法的“型”的数值应该是V减去所观察的“型”的数值：

$$Z(w) = V - T(w) \tag{6.25}$$

这里是每个单词的Z的值：

I	1521
want	1540
to	1486
eat	1492
Chinese	1596
food	1534
lunch	1571

图6.9是打折以后的饭店二元语法的计数。

	I	want	to	eat	Chinese	food	lunch
I	8	1060	0.062	13	0.062	0.062	0.062
want	3	0.046	740	0.046	6	8	6
to	3	0.085	10	827	3	0.085	12
eat	0.075	0.075	2	0.075	17	2	46
Chinese	2	0.012	0.012	0.012	0.012	109	1
food	18	0.059	16	0.059	0.059	0.059	0.059
lunch	4	0.026	0.026	0.026	0.026	1	0.026

图6.9 在Berkeley Restaurant Project语料库中（容量约为10 000个句子），从1616个单词的“型”中选出7个单词的Witten-Bell平滑二元语法的计数

Witten-Bell算法的折扣值比加1平滑合理得多：

I	0.97
want	0.94
to	0.96
eat	0.88
Chinese	0.91
food	0.94
lunch	0.91

也可以按不同的方式来使用 Witten-Bell 打折法（或其他方法）。在式（6.21）中，我们是以前面的单词作为平滑二元语法概率的条件的。这就是说，我们以前面的单词w_x作为“型”的计数$T(w_x)$和“例”的计数$N(w_x)$的条件。但是，我们也可以不考虑二元语法由两个单词组成的事实，把二元语法看成似乎是一个单一的事件。这样，T就变成了所有二元语法的“型”的数值，N就变成了所有二元语法发生时的“例”的数值。用这样的办法把二元语法当成一个单元来处理，实际上打折的就不是条件概率$P(w_i|w_x)$，而是联合概率（joint probability）$P(w_xw_i)$了。在这种情况下，概率$P(w_xw_i)$就可以像一元语法那样来处理了。这一类的打折与从式（6.21）开始进行的“条件”打折相比，并不经常使用（尽管这种办法在下面将要介绍的 Good-Turing 打折法中是经常使用的）。

在6.4节中可以看出，这种打折的方法在一些更复杂的语言模型中也在起作用。Witten-Bell 打折法是 Placeway et al.(1993)语音识别系统中经常使用的方法。

6.3.3 Good-Turing 打折法

在本节中，我们介绍一种比 Witten-Bell 算法稍微复杂一些的打折方法，称为 Good-Turing 打折法（Good-Turing Discounting）。不想重点研究打折法的读者可以跳过这一节不读。

Good(1953)首先描述了 Good-Turing 算法，而这种算法的原创思想则来自 Turing。这种算法的完全证明参见 Church et al.(1991)。Good-Turing 平滑的基本思想是，用观察计数较高的N元语法数的方法来重新估计概率量的大小，并把它指派给那些具有零计数或较低计数的N元语法。换言之，我们要检查出现次数为c的N元语法数N_c。把出现次数为c的N元语法数看成频度c出现的频度。这样，应用平滑二元语法联合概率的思想，N_0就是计数为0的二元语法b的数值，N_1就是计数为1的二元语法的数值，依次类推，则有：

$$N_c = \sum_{b:c(b)=c} 1 \tag{6.26}$$

对于所有的c，根据N_c的集合，Good-Turing 估计给出一个平滑计数c^*：

$$c^* = (c+1)\frac{N_{c+1}}{N_c} \tag{6.27}$$

例如，这种修正了的从未出现的二元语法的计数(c_0)，就可以使用把出现一次的二元语法（单一元素或只在语料库中出现一次的单词的二元语法N_1）的数值分摊给从未出现的二元语法的数值(N_0)。这种使用我们看过一次的事物的计数来估计从未看见过的事物的计数的方法，在本章的前面讲 Witten-Bell 打折算法时曾经讨论过，读者应该记忆犹新。Nádas(1984)指出，首先把 Good-Turing 算法用于N元语法平滑的是 Katz。图6.10给出了 Church and Gale(1991)将 Good-Turing 打折法应用于二元语法的一个例子，使用的是容量为2200万单词的 Associated Press（AP）新闻语料。第一列是计数c，也就是二元语法的所观察例子的数值。第二列是这个计数所具有的二元语法数。例如，449 721个二元语法的计数为2。第三列是c^*，这是用 Good-Turing 打折法重新估计的计数。

Church et al.(1991)指出，Good-Turing 估计所依据的假设是每个二元语法的分布都是二项式。Good-Turing 估计假定我们知道未见的二元语法数N_0。我们之所以知道N_0，是因为对于给定的词汇容量V，二元语法的总数必定是V^2（N_0等于V^2减去我们未见的二元语法）。

实际上，并不是对于所有的计数c都使用打折估计c^*。假定较大的计数是可靠的（对于某个阈值k，$c>k$），Katz(1987)建议取k的值为5。这样，我们可定义：

$$c^* = c \text{ for } c > k \tag{6.28}$$

c (MLE)	N_c	c^* (GT)
0	74 671 100 000	0.000027 0
1	2 018 046	0.446
2	449 721	1.26
3	188 933	2.24
4	105 668	3.24
5	68 379	4.22
6	48 190	5.19
7	35 709	6.21
8	27 710	7.24
9	22 280	8.25

图 6.10　容量为 2200 万个单词的 AP 二元语法中，“频度出现的频度”的二元语法以及 Good-Turing 重新估值(Church and Gale, 1991)

当引入某个 k 的时候，c^* 的正确等式是(Katz, 1987)：

$$c^* = \frac{(c+1)\frac{N_{c+1}}{N_c} - c\frac{(k+1)N_{k+1}}{N_1}}{1 - \frac{(k+1)N_{k+1}}{N_1}}, \quad 1 \leqslant c \leqslant k \tag{6.29}$$

在使用 Good-Turing 打折法时，与使用其他打折法一样，通常都用处理计数为 0 的方法来处理计数低的（特别是计数为 1 的）N 元语法。

6.4　回退

前面讨论的打折法可以帮助我们解决零频度 N 元语法问题。但除此之外，自然语言中还存在着可以汲取的其他知识资源。如果不存在特定的三元语法的例子 $w_{n-2}w_{n-1}w_n$ 来帮助我们计算概率 $P(w_n|w_{n-1}w_{n-2})$，就可以使用二元语法概率 $P(w_n|w_{n-1})$来估计这个概率。相似地，如果没有有关的计数来计算二元语法概率 $P(w_n|w_{n-1})$，可以使用一元语法的概率 $P(w_n)$来估计它。

根据这种 N 元语法的“层级关系”，有两种办法可以用来帮助我们解决平滑的问题，一种办法是“删除插值法”（deleted interpolation），一种办法是“回退”（backoff）。现在我们先重点讨论回退，在本节之后会迅速回顾一下关于删除插值法的问题。回退 N 元语法模型是 Katz(1987)提出的。正如在删除插值法中那样，在回退模型中根据 N–1 元语法模型来建立 N 元语法模型。不同之处在于，在回退模型中如果有非零的三元语法计数，则只依靠这些三元语法计数，根本不插入二元语法和一元语法计数。仅当阶数较高的 N 元语法中存在零计数时才采用回退模型，把阶数较高的 N 元语法降为阶数较低的 N 元语法。

回退的三元语法可以表示如下：

$$\hat{P}(w_i|w_{i-2}w_{i-1}) = \begin{cases} P(w_i|w_{i-2}w_{i-1}), & \text{若 } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha_1 P(w_i|w_{i-1}), & \text{若 } C(w_{i-2}w_{i-1}w_i) = 0 \\ & \text{且 } C(w_{i-1}w_i) > 0 \\ \alpha_2 P(w_i), & \text{其他} \end{cases} \tag{6.30}$$

现在我们暂时不考虑等式中 α 的值；关于这些加权因子的必要性将在下面讨论。我们先来看看表示回退模型一般情况的（递归）等式。

$$\hat{P}(w_n|w_{n-N+1}^{n-1}) = \tilde{P}(w_n|w_{n-N+1}^{n-1}) + \theta(P(w_n|w_{n-N+1}^{n-1}))\alpha\hat{P}(w_n|w_{n-N+2}^{n-1}) \quad (6.31)$$

现在，我们也暂时不考虑等式中的 α 和 $\tilde{P}$。根据 Katz 的方法，我们用 θ 来表示二元函数，这个函数仅当阶数较高的模型具有零概率时选择一个阶数较低的模型进行回退处理：

$$\theta(x) = \begin{cases} 1, & \text{若} x = 0 \\ 0, & \text{其他} \end{cases} \quad (6.32)$$

每个 $P(\cdot)$ 是一个最大似然估计 MLE（也就是用分摊计数的办法直接进行计算）。在下一节中，我们将更详细地讨论这些等式。为此，我们需要理解 α 值的作用，懂得如何计算它。

6.4.1 回退与打折相结合

前面讨论了回退，说明了如何使用回退算法来给未见事件指派概率量。为简单起见，我们假定所有这些未见事件都是等概率的，所以概率量将平等地在所有未见事件中进行分摊。现在，我们可以把打折和刚才讲过的回退算法结合起来，使得我们给未见事件分摊概率时变得聪明一些。使用打折算法可以知道有多少概率量可以在所有未见事件中进行分摊，而使用回退算法则可以知道用什么办法来分摊这些概率比较聪明。

首先，请读者停下来回答下面的问题（不要向前看）：为什么在式（6.30）或式（6.31）中需要 α 值？为什么不能只有三套没有权值的概率？

问题的答案是：如果没有 α 值，那么等式的结果就不是一个真正的概率。这是因为我们从相对频度得到的原概率 $P(w_n|w_{n-N+1}^{n-1})$ 是真正的概率，这就是说，如果我们对于给定的 w_n，在所有 N 元语法上下文中，其概率之和将为 1：

$$\sum_{i,j} P(w_n|w_i w_j) = 1 \quad (6.33)$$

但是，如果情况果真如此。而当概率为零时，我们回退到一个阶数较低的模型，我们就要把多余的概率量加到等式中，这样一来，单词的总概率就将大于 1！

因此，所有的回退语言模型都必须打折。现在我们来解释式（6.31）中的 α 和 $\tilde{P}$。$\tilde{P}$ 用于给最大似然估计 MLE 的概率打折，以便为低阶 N 元语法节省概率量。我们用 $\tilde{P}$ 来表示打折概率，它为那些直接从计数计算出来的旧的相对频度节省 P。α 用于保证所有低阶 N 元语法概率量之和恰恰等于通过对高阶 N 元语法打折节省下来的概率量。下面是最后得到的正确等式：

$$\hat{P}(w_n|w_{n-N+1}^{n-1}) = \tilde{P}(w_n|w_{n-N+1}^{n-1}) + \theta(P(w_n|w_{n-N+1}^{n-1})) \cdot \alpha(w_{n-N+1}^{n-1})\hat{P}(w_n|w_{n-N+2}^{n-1}) \quad (6.34)$$

现在我们来研究等式中这些成分的形式定义。我们把 $\tilde{P}$ 定义为 N 元语法条件概率(c^*)的打折最大似然估计 MLE：

$$\tilde{P}(w_n|w_{n-N+1}^{n-1}) = \frac{c^*(w_{n-N+1}^{n})}{c(w_1^{n-N+1})} \quad (6.35)$$

这个概率 $\tilde{P}$ 将比如下的最大似然估计 MLE 稍微小一些：

$$\frac{c(w_{n-N+1}^{n})}{c(w_{n-N+1}^{n-1})}$$

（也就是说，平均起来c^*比c小）。这样将留下一些概率量给低阶N元语法。现在我们需要建立把这些概率量分摊给低阶N元语法的α权值。让我们用函数β来表示留下的概率量的总数，β是$N-1$元语法的上下文。对于给定的$N-1$元语法的上下文，通过从1减去以该上下文开始的所有N元语法的全部打折概率量，可以计算全部留下的概率量：

$$\beta(w_{n-N+1}^{n-1}) = 1 - \sum_{w_n:c(w_{n-N+1}^{n})>0} \tilde{P}(w_n|w_{n-N+1}^{n-1}) \tag{6.36}$$

这就为我们给出了分摊到所有$N-1$元语法（例如，如果原来的语法是三元语法，那么，$N-1$元语法就是二元语法）的全部概率量。每个单独的$N-1$元语法（二元语法）只能得到这个概率量的一个分量，因此需要通过使某个N元语法（三元语法）开始的所有$N-1$元语法（二元语法）的全部概率来归一化β。计算从一个N元语法分摊给一个$N-1$元语法的概率量多少的最终等式用函数α来表示：

$$\alpha(w_{n-N+1}^{n-1}) = \frac{1 - \sum_{w_n:c(w_{n-N+1}^{n})>0} \tilde{P}(w_n|w_{n-N+1}^{n-1})}{1 - \sum_{w_n:c(w_{n-N+1}^{n})>0} \tilde{P}(w_n|w_{n-N+2}^{n-1})} \tag{6.37}$$

注意，α是前面单词串w_{n-N+1}^{n-1}的函数，因此我们给每个三元语法打折的量(d)，以及给低阶N元语法重新指派的量（α），都要对每N元语法（更精确地说，对出现在N元语法中的每个N-1元语法）重新计算。

我们这里只需要说明，当一个$N-1$元语法上下文的计数为0时，也就是当$c(w_{n-N+1}^{m-1}) = 0$时应该怎样做，我们的定义是：

$$P(w_n|w_{n-N+1}^{n-N+1}) = P(w_n|w_{n-N+1}^{n-N+2}) \tag{6.38}$$

$$\tilde{P}(w_n|w_{n-N+1}^{n-1}) = 0 \tag{6.39}$$

并且

$$\tilde{\beta}(w_{n-N+1}^{n-1}) = 1 \tag{6.40}$$

在式（6.35）中，打折概率$\tilde{P}$可以用Witten-Bell打折法中的打折计数c^*，见式（6.20），来计算，或者用Good-Turing打折法来计算。

这里是以稍微清楚的形式表示的三元语法回退模型：

$$\hat{P}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \tilde{P}(w_i|w_{i-2}w_{i-1}), & \text{若 } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha(w_{n-2}^{n-1})\tilde{P}(w_i|w_{i-1}), & \text{若 } C(w_{i-2}w_{i-1}w_i) = 0 \\ & \text{且 } C(w_{i-1}w_i) > 0 \\ \alpha(w_{n-1})\tilde{P}(w_i), & \text{其他} \end{cases}$$

实际上，当打折的时候通常忽略了为1的计数，也就是说，当我们处理计数为1的N元语法时，就好像它们从来没有发生过一样。

Gupta et al.(1992)介绍了把概率指派给零三元语法的一种回退方法，这是回退方法的一种变体。

6.5 删除插值法

删除插值法（Deleted Interpolation）是Jelinek and Mercer(1980)提出的，这种方法使用线性插值的手段，把不同阶的N元语法结合起来。例如，当计算三元语法时，把一元语法、二元语法和三

元语法三种模型都结合起来。这就是说，当估计概率$P(w_n|w_{n-1}w_{n-2})$时，我们要把一元语法、二元语法和三元语法都混合在一起。每种语法用线性权值λ来加权：

$$\begin{aligned}\hat{P}(w_n|w_{n-1}w_{n-2}) &= \lambda_1 P(w_n|w_{n-1}w_{n-2}) \\ &\quad +\lambda_2 P(w_n|w_{n-1}) \\ &\quad +\lambda_3 P(w_n)\end{aligned} \tag{6.41}$$

使得各个λ的和为1：

$$\sum_i \lambda_i = 1 \tag{6.42}$$

实际上，在删除插值法中我们不仅仅只为三元语法训练三个λ，还把每个λ看成上下文的函数。如果对于一个特定的二元语法有特定的精确计数，假定三元语法的计数是基于二元语法的，那么这样的办法将更加可靠；因此，可以使这些三元语法的λ值更高，从而在插值时给三元语法更高的权值　这样，插值公式可进一步扩展为：

$$\begin{aligned}\hat{P}(w_n|w_{n-2}w_{n-1}) &= \lambda_1(w_{n-2}^{n-1})P(w_n|w_{n-2}w_{n-1}) \\ &\quad +\lambda_2(w_{n-2}^{n-1})P(w_n|w_{n-1}) \\ &\quad +\lambda_3(w_{n-2}^{n-1})P(w_n)\end{aligned} \tag{6.43}$$

给定$P(w\cdots)$的值，使用第7章中的一种EM算法(Baum, 1972; Dempster et al., 1977; Jelinek and Mercer, 1980)来训练λ的值，使得从主训练语料库中分出来的语料库的似然度最大。关于算法的进一步细节在Bahl et al.(1983)中有细致的描述。

6.6　拼写和发音的N元语法

在第5章中我们已经看到了如何使用贝叶斯噪声信道算法来进行错拼更正，如何从给定的表层发音中挑出其对应的单词。同时，我们也看到了这两种算法都是不成功的，它们返回的是错误的单词，因为无法模拟含有多个单词的词。现在，我们的N元语法提供了这样的模型，我们回过头来再次讨论这两个问题。

6.6.1　上下文有关的错拼更正

在第5章中，我们介绍了检查拼写错误的一些方法，这些方法是：查找该单词是否在词典中存在，如果不存在就是错拼单词；用符合英语构词法的有限状态自动机来生成，如果不能生成就是错拼单词；如果符合正词法规则的概率很低，就是错拼单词。但是，这些方法对于检查和更正真词错拼（real-word spelling error）或真词错误更正（real-word error detection）都是不充分的。因为真词错拼之类的错误单词都是在真实的英语单词中实际存在的单词。真词错拼的发生，是由于在排版印刷时出现错误操作（插入错误、脱落错误或换位错误），使得偶然排印出一个英语中存在的真词（例如，把*there*打成*three*），或者由于写作的人用同音词或准同音词来错误地进行替换（例如，用*dessert*来替换*desert*，用*piece*来替换*peace*）。对于这种类型错误的更正称为上下文有关的错拼更正（context-sensitive spelling error correction）。

这种类型的错误的重要性如何呢？ Peterson(1986)根据对于单个排版印刷错误（插入、脱落、替代、换位）的初步分析，估计由于这些排版印刷错误而产生的英语真词量（对于一个相当大的350 000个单词的词表）大约为15%。Kukich(1992)根据对于语料库的实验研究，总结出不同的分析结果，认为英语真词错误的比例在25%~40%。图6.11是Kukich(1992)给出的一些例子。他把这种

错误分为局部性错误（local error）和全局性错误（global error）两种。局部性错误是根据直接围绕该词的上下文就可以检查出的错误，全局性错误是需要根据更广泛的上下文才能检查出的错误。

局部性错误
The study was conducted mainly *be* John Black. They are leaving in about fifteen *minuets* to go to her house. The design *an* construction of the system will take more than a year. Hopefully, all *with* continue smoothly in my absence. Can they *lave* him my messages? I need to *notified* the bank of [this problem.] He *need* to go there right *no w*. He is trying to *fine* out.
全局性错误
Won't they *heave if* next Monday at that time? This thesis is supported by the fact that since 1989 the system has been operating *system* with all four units on-line, but ...

图 6.11　Kukich(1992)发现的一些真词拼写错误，分为局部性错误和全局性错误两种

上下文有关的拼写错误检查的方法之一是基于 *N* 元语法的方法。

用于拼写检查与更正的基于单词的 *N* 元语法方法是 Mays et al.(1991)提出的。这种方法的基本思想是，对于句子中的每个单词生成它的一切可能的错误拼写，或者只包括排版印刷错误而造成的错误拼写（字母的插入、删除或替换），或者也包括同音词造成的错误拼写（可能包括正确拼写），然后选出使该句子具有最高先验概率的拼写。这就是说，给定一个句子 $W = \{w_1, w_2, \cdots, w_k, \cdots, w_n\}$，其中 w_k 的不同拼写是 w_k', w_k'',等等。在这些可能的拼写中，我们使用 *N* 元语法计算 $P(W)$，从中选择最大的 $P(W)$。也可以使用基于词类的 *N* 元语法来代替基于单词的 *N* 元语法，发现不可靠的词类组合，但这种方法可能不如发现不可靠的单词组合的效果好。

上下文有关的错拼更正还有其他的统计方法，有些方法是专为更正而提出的，有些方法则是针对更为一般的词汇歧义消解而提出的（例如，词义的歧义消解或重音复原）。除了我们刚才描写的三元语法方法之外，还包括贝叶斯分类法，或者贝叶斯分类与三元语法相结合的方法(Gale et al., 1993; Golding, 1997; Golding and Schabes, 1996)、判定表方法(Yarowsky, 1994)、基于转换的学习方法(Mangu and Brill, 1997)、潜在语义分析法(Jones and Martin, 1997)、筛选法[Winnow](Golding and Rose, 1999)。把这些方法进行对比，Golding and Roth(1999)发现，筛选算法效果最好。总的来说，这些算法在很多方面是相似的，它们的根据都是单词和词类的 *N* 元语法这样的特征，Roth(1998, 1999)指出，其中很多算法使用一族线性预测算子来进行预测，称为线性统计询问（Linear Statistical Queries，简称 LSQ）假设。第 17 章将定义这些算法，并且在单词意义的歧义消解的背景下，进一步讨论这个问题。

6.6.2　发音模型的 *N* 元语法

N 元语法也可以用于我们在第 5 章中讨论过的词形发音问题，并能取得较好的结果。当时的输入是在单词 *I* 之后的发音[n iy]。我们曾经指出，发音[n iy]对应的单词有五个：*need*, *new*, *neat*, *the* 和 *knee*。我们在第 5 章中使用的算法是根据每个单词的一元语法概率和发音似然度的乘积来进行计算的，结果我们不正确地选择了单词 *new*，主要的理由是这个单词的一元语法概率比较高。

如果加上简单的二元语法概率，即使不进行适当的平滑，也足以正确地解决这个问题。在下面的表中，没有像第 5 章中那样只列出单词的一元语法概率 $p(w)$，而是列出了五个候选单词在单词 *I* 之后的二元语法概率（把 Brown 语料库和 Switchboard 语料库结合起来，单词 *I* 共出现 64 736 次）：

Word	$C('I'w)$	$C('I'w)+0.5$	$p(w\|'I')$
need	153	153.5	0.0016
new	0	0.5	0.000 005
knee	0	0.5	0.000 005
the	17	17.5	0.000 18
neat	0	0.5	0.000 005

现在把相应单词的这些新的概率代入组合模式的公式进行计算，预测出正确的单词应该是need，如下表所示：

Word	$p(y\|w)$	$p(w)$	$p(y\|w)p(w)$
need	0.11	0.0016	0.000 18
knee	1.00	0.000 005	0.000 005
neat	0.52	0.000 005	0.000 0026
new	0.36	0.000 005	0.000 0018
the	0	0.000 18	0

6.7 熵

I got the horse right here

Frank, Loesser, Guys and Dolls[①]

熵（entropy）和困惑度（perplexity）是用于评估N元语法系统的最普通的计量方法。下面将对信息论的一些最必要的基本知识进行总结性的介绍，并介绍熵和困惑度的计算方法。我们强烈建议读者阅读一本较好的信息论教材，Cover and Thomas(1991)就是很好的教材。

熵是信息量的度量，在自然语言处理、语音识别和计算语言学中，熵都是非常有价值的。熵可以用来度量一个特定语法中的信息量是多少，度量给定语法和给定语言的匹配程度有多高，预测一个给定的N元语法中的下一个单词是什么。如果有给定的两个语法和一个语料库，我们可以使用熵来估计哪个语法与语料库匹配得更好。我们也可以使用熵来比较两个语音识别任务的困难程度，还可以使用它来测量一个给定的概率语法与人类语法的匹配程度。

熵的计算要求我们在所要预测的范围内（单词、字母和词类，我们称为χ的集合）建立一个随机变量X，并且要有一个特定的概率函数，称为$p(x)$，那么这个随机变量X的熵为：

$$H(X) = -\sum_{x\in\chi} p(x)\log_2 p(x) \tag{6.44}$$

从原则上说，对数可以使用任何底数。本书中我们在所有计算中采用的底数都是2，因此，熵的就用比特（bit）来度量，因为计算比特时是以2为底数的。

对于计算机科学家来说，定义熵的最直观方法，就是把熵想像成对某一判断进行信息编码的比特数的下界，或者把熵想像成在最优编码中信息量的大小。

Cover and Thomas(1991)提出了如下的例子。假定我们想给Yonkers赛马场的赛马下赌注，但是Yonker赛马场距离我们太远，我们只好给赛马场登记赌注的人发一个短的消息，告诉他我们给哪匹马下赌注。假定有八匹马参加比赛。

给这个消息编码的一种方法是用二进制代码来表示马的号码；这样，号码为1的马的二进制代码是001，号码为2的马的二进制代码是010，号码为3的马的二进制代码是011，依次类推，号码

① 我猜中了这匹马。

为8的马的二进制代码是000。如果我们用一天的时间来下赌注，每匹马用比特来编码，每次比赛我们要发出3比特的信息。

我们能不能把这件事做得好一点呢？我们可以根据赌注的实际分布来传送消息，假定每匹马的先验概率如下：

马	1	$\frac{1}{2}$	马	5	$\frac{1}{64}$
马	2	$\frac{1}{4}$	马	6	$\frac{1}{64}$
马	3	$\frac{1}{8}$	马	7	$\frac{1}{64}$
马	4	$\frac{1}{16}$	马	8	$\frac{1}{64}$

计算这些马的随机变量 X 的熵，我们就可以知道其比特数的下界，具体计算如下：

$$\begin{aligned} H(X) &= -\sum_{i=1}^{i=8} p(i)\log p(i) \\ &= -\tfrac{1}{2}\log\tfrac{1}{2} - \tfrac{1}{4}\log\tfrac{1}{4} - \tfrac{1}{8}\log\tfrac{1}{8} - \tfrac{1}{16}\log\tfrac{1}{16} - 4(\tfrac{1}{64}\log\tfrac{1}{64}) \\ &= 2\text{比特} \end{aligned} \tag{6.45}$$

每次比赛平均为2比特的代码可以这样来编码：用最短的代码来表示我们估计概率最大的马，估计概率越小的马，其代码越长。例如，我们可以用0来给估计概率最大的马编码，按照估计概率从大到小的排列，其余的马的代码分别为：10，110，1110，111100，111101，111110和111111。

如果我们对于每匹马的概率估计都是一样的，情况将如何呢？前面我们已经看到，如果对于每匹马都使用等长的二进制编码，每匹马都用3比特来编码，因此平均的比特数为3。这时的熵是一样的吗？是的，在这种情况下，每匹马的估计概率都是1/8。我们选择马的熵是这样计算的：

$$H(X) = -\sum_{i=1}^{i=8}\frac{1}{8}\log\frac{1}{8} = -\log\frac{1}{8} = 3\text{比特} \tag{6.46}$$

2^H 这个值称为困惑度(Jelinek et al., 1977; Bahl et al., 1983)。从直觉上，我们可以把困惑度理解为在随机实验中选择随机变量的加权平均数。因此，在等概率估计的8匹马之间进行选择时（这时熵 $H=3$ 比特），困惑度是 2^3，也就是8。在概率有差异的8匹马之间进行选择时（这时熵 $H=2$ 比特），困惑度是 2^2，也就是4。

前面计算的是单个变量的熵。然而，我们在很多场合需要计算序列的熵。例如，对于一个语法来说，我们需要计算单词的序列 $W=\{\cdots w_0, w_1, w_2, \cdots, w_n\}$ 的熵。我们的办法之一是让变量能够覆盖单词的序列。例如，我们可以计算在语言 L 中长度为 n 的单词的一切有限序列的随机变量的熵。计算公式如下：

$$H(w_1, w_2, \cdots, w_n) = -\sum_{W_1^n \in L} p(W_1^n)\log p(W_1^n) \tag{6.47}$$

我们可以把熵率（entropy rate）定义为用单词数来除这个序列的熵所得的值（我们也可以把熵率想像成每个单词的熵）：

$$\frac{1}{n}H(W_1^n) = -\frac{1}{n}\sum_{W_1^n \in L} p(W_1^n)\log p(W_1^n) \tag{6.48}$$

但是，为了计算一个语言的真正的熵，我们需要考虑无限长度的序列。如果我们肯把语言想像成产生单词序列的随机过程 L，那么它的熵率 $H(L)$ 可定义为：

$$H(L) = \lim_{n\to\infty} \frac{1}{n} H(w_1, w_2, \cdots, w_n)$$
$$= \lim_{n\to\infty} \frac{1}{n} \sum_{W\in L} p(w_1, \cdots, w_n) \log p(w_1, \cdots, w_n) \quad (6.49)$$

Shannon-McMillan-Breiman 定理(Algoet and Cover, 1988; Cover and Thomas, 1991)指出，如果语言在某种意义上是正则的（确切地说，如果语言既是平稳的又是遍历的），那么有：

$$H(L) = \lim_{n\to\infty} -\frac{1}{n} \log p(w_1 w_2 \cdots w_n) \quad (6.50)$$

这意味着，我们可以取语言中一个足够长的序列来替代该语言中所有可能的序列的总和。Shannon-McMillan-Breiman 定理的直觉解释是：一个足够长的单词序列可以在其中包含其他很多较短的序列，而且每个这样的较短的序列都可以按照它们各自的概率重复出现在较长的序列中。

如果随着时间的推移，随机过程指派给序列的概率不变，则可以认为这个随机过程是平稳的。换言之，在平稳随机过程中，单词在时间 t 的概率分布与在时间 $t+1$ 的概率分布是相同的。马尔可夫模型以及 N 元语法的概率分布都是平稳的。例如，在二元语法中，P_i 只依赖于 P_{i-1}，因此如果把时间的索引号移动到 x，P_{i+x} 仍然依赖于 P_{i+x-1}。然而自然语言却不是平稳的，我们在第9章中将会看到，在自然语言中，下一个单词的概率可能依赖于任意距离的事件并且依赖于时间。所以，我们的统计模型对于自然语言的正确分布和熵的描述都是近似的。

最后，使用这种尽管不正确但却非常方便的简单假设，我们就能够取一个很长的输出样本来计算某个随机过程的熵，并且计算它的平均对数概率。下一节将讨论这样做的原因和方法，讨论为什么要这样做（也就是说，在哪些问题上熵可以告诉我们一些有用的东西）以及怎样来计算这种长序列的概率。

6.7.1 用于比较模型的交叉熵

在本节中将介绍交叉熵（cross entropy），并讨论交叉熵在比较不同的概率模型时的用途。当我们不知道生成某些数据的实际概率分布 p 时，交叉熵是很有用处的。交叉熵使得我们可以使用某个 m 作为 p 的一个模型（即 m 作为 p 的近似）。m 对于 p 的交叉熵定义如下：

$$H(p,m) = \lim_{n\to\infty} \frac{1}{n} \sum_{W\in L} p(w_1, \cdots, w_n) \log m(w_1, \cdots, w_n) \quad (6.51)$$

这意味着，我们根据概率分布 p 取出一个序列，根据 m 来求它们的概率的对数之和。

按照 Shannon-McMillan-Breiman 定理，对于一个平稳遍历过程，有：

$$H(p,m) = \lim_{n\to\infty} -\frac{1}{n} \log m(w_1 w_2 \cdots w_n) \quad (6.52)$$

由于交叉熵 $H(p, m)$是熵 $H(p)$的上界，所以交叉熵很有用。对于任何模型 m，我们有：

$$H(p) \leqslant H(p,m) \quad (6.53)$$

这种情况表明，我们可以使用某个简化的模型 m 来帮助我们估计根据概率 p 取出的一个符号序列的真正的熵。m 越精确，交叉熵 $H(p, m)$就越接近于真正的熵 $H(p)$。所以，$H(p, m)$与 $H(p)$之差是模型精确程度的度量。在两个模型 m_1 和 m_2 之间，交叉熵较低的模型将是更精确的模型。交叉熵永远不可能低于真正的熵，所以模型的估值不能低于真正的熵。

方法论探讨：困惑度

在前面的“方法论探讨”中，曾经提出用测试集的困惑度来比较两个概率模型的思想。尽管可能会引起歧义，但我们还是按照语音和语言处理中的惯例，使用“困惑度”（perplexity）这个术语，而不使用更有技术色彩的正规术语“交叉困惑度”（cross perplexity）。这里举一个在“商业新闻听写系统”中计算困惑度的例子。我们用Good-Turing打折法训练一元、二元和三元的，具有Katz风格的回退语法，语料容量为3800万单词（包括“句子开始”这样的“词例”），语料来自华尔街日报（Wall Street Journal，简称WSJ），即WSJ0语料库(LDC, 1993)。我们使用包含19 979个单词的词汇（也就是说，在训练集和测试集中，其他单词的“型”将被映射为“未知词”的“例”<UNK>）。然后，我们在150万词的测试集上计算每种模型的困惑度，这里的困惑度定义为$2^{H(p,m)}$。下面的表说明了在150万词的WSJ中不同语法的困惑度。

*N*元语法的阶	困惑度
一元语法	962
二元语法	170
三元语法	109

在计算困惑度时，模型m不应该具有任何关于测试集t的知识。任何种类的关于测试集的知识将会引起困惑度的人为降低。例如，有时我们不是把所有未知词都映射为<UNK>这个“词例”，而是使用了一个封闭词汇测试集，而事先已经知道这个测试集中的词是什么。这样将会大大地降低困惑度。由于这些知识都平等地提供了所要比较的每个模型，所以封闭词汇的困惑度对于模型的比较还是有用的。但是，这个交叉困惑度不再保证大于测试集的真正困惑度，因此在解释结果时必须非常小心。一般来说，两个语言模型的困惑度，仅当都使用相同的词汇时才是可以比较的。

6.7.2 英语的熵

我们在前一节中说过，某个模型m的交叉熵可以用来作为某个随机过程的真正熵的上界。我们可以使用这样的方法来估计英语的真正的熵。为什么要关心英语的熵呢？

第一个原因是英语的真正的熵将为我们将来对概率语法的试验提供一个可靠的下界。另一个原因是我们可以利用英语的熵值帮助理解语言中的哪一部分提供的信息最大。例如，判断英语的预测能力主要依赖于词序、语义、形态、组成成分还是语用线索。这样可以在很大程度上帮助我们了解语言模型应该着重研究哪一方面。

计算英语熵值的方法通常有两种。第一种方法是Shannon(1951)使用的方法，这是Shannon在信息论领域的开创性工作的一部分，其思路是利用受试人来构造一个信息试验，要求受试人来猜测字母，观察猜测的字母中有多少是正确的，从而估计字母的概率，然后估计序列的熵值。

实际的试验是这样来设计的：我们给受试人看一个英语文本，然后要求受试人猜测下一个字母。受试人利用他们的语言知识来猜测最可能出现的字母，然后猜测下一个最可能的字母，依次类推。我们把受试人猜对的次数记录下来。Shannon指出，猜测数序列的熵与英语的熵是相同的。其直觉解释是：如果受试人进行n个猜测，那么，给定猜测数序列，我们能够通过选择第n个最可能的字母的方法，重建原来的文本。这样的方法要求猜字母而不是猜单词（因为受试人有时必须对所有的字母进行穷尽的搜索），所以Shannon计算的是英语中每个字母的熵，而不是英语中每个单词的熵。他报告的结果是：英语字母的熵是1.3比特（对于27个字母而言，包括26个字母以及空白）。Shannon的这个估值太低了一些，因为他是根据单篇的文本（Dumas Malone的*Jefferson the Virginian*）来进行试验的。Shannon还注意到，对于其他的文本（例如新闻报道、科学著作和诗歌），

他的受试人往往会猜错（因此这时的熵比较高）。最近有人模仿 Shannon 做了类似的试验，他们以赌场为例，让受试人对下一个出现的字母打赌(Cover and King, 1978; Cover and Thomas, 1991)。

第二种计算英语的熵的方法有助于避免导致Shannon结果失误的单篇文本的问题。这个方法使用一个很好的随机模型，在一个很大的语料库上训练这个模型，用它给一个很长的英语序列指派一个对数概率，计算时使用 Shannon-McMillan-Breiman 定理：

$$H(\text{English}) \leqslant \lim_{n\to\infty} -\frac{1}{n}\log m(w_1 w_2 \cdots w_n) \tag{6.54}$$

例如，Brown et al.(1992)在 58 300 万单词的英语文本上（293 181 个“型”）训练了一个三元语法模型，用它来计算整个 Brown 语料库的概率（1 014 312 个“例”）。训练数据包括新闻、百科全书、小说、官方通信、加拿大议会论文集以及其他各种资源。

然后，他们使用词的三元语法给 Brown 语料库指派概率，把语料库看成是一个字母序列，从而计算 Brown 语料库的字符的熵。他们得到的结果是：每个字符的熵为 1.75 比特（这里的字符集包含了 95 个可印刷的全部 ASCII 字符）。

根据报告，英语书面文本中单词（包括空白）的平均长度是 5.5 个字符(Nádas, 1984)。如果这个报告的数值是正确的，就意味着在普通的英文中，Shannon 估计的每个字母的熵为 1.3 比特将对应于每个单词的困惑度为 142。前面报告的 WSJ 试验的困惑度数值明显低于这个数值，其原因在于训练集和测试集都来自英语的同样的子样本。这样就表明这个试验低估了英语的复杂性，因为华尔街日报看起来确实很不像莎士比亚的著作。

6.8 小结

本章介绍了 N 元语法，这是一种年代久远而使用广泛的语言处理实用工具。

- N元语法概率是一个单词在前面给定的N–1个单词的条件下的条件概率。N元语法概率可以通过在语料库中简单地计数并使之归一化的方法来进行计算（即最大似然估计），或者也可以通过更复杂的算法来进行计算。N元语法的优点是可以使用丰富的词汇知识，缺点是对训练语料库的依赖性太强。
- 平滑为估计那些从来不出现的 N 元语法概率提供了一种较好的解决办法。常用的平滑算法有回退算法、删除插值法、Witten-Bell 打折法和 Good-Turing 打折法。
- 评测诸如 N 元语法这样的基于语料库的语言模型时，要把语料库分为训练集和测试集两部分。在训练集上训练模型，在测试集上评测模型。测试集的熵 H 或者困惑度 2^H（更恰当的术语是交叉熵和交叉困惑度)用于对语言模型进行比较。

6.9 文献和历史说明

N元语法的数学原理最早是Markov(1913)提出的。Markov使用我们现在称为“Markov链”（马尔可夫链）（二元语法和三元语法）的数学概念来预测普希金的 Eugene Onegin（《欧根·奥涅金》）中下一个字母是元音还是辅音。Markov 把书中的 20 000 个字母分为 V（元音）和 C（辅音），并计算二元语法和三元语法的概率，如果要判定给定的字母是否为元音，需要根据它前面的一个或两个字母来决定。Shannon(1948)通过对N元语法的计算来逼近英语的单词序列。在Shannon工作的基础上，马尔可夫模型成为了20世纪50年代普遍使用的单词序列模型。从Chomsky(1956)开始，Chomsky发表了一系列非常有影响的文章，其中包括 Chomsky(1957)以及 Miller and Chomsky(1963)。

Chomsky雄辩地证明，“有限状态马尔可夫过程”尽管可能在工程方面会得到某些探索性的应用，但不能作为人类语法知识的完美认知模型。Chomsky的这种论点使得很多语言学家和计算语言学家彻底地抛弃了统计模型。

*N*元语法模型的复兴是从IBM公司和CMU（卡内基－梅隆大学）开始的。在Shannon的影响下，IBM公司Thomas J. Wanson研究中心的Jelinek, Mercer, Bahl和他们的同事们回过头来研究*N*元语法模型；在Baum和他的同事们的工作的影响下，CMU的Baker也回过头来研究*N*元语法。这两个实验室在语音识别系统中独立地成功使用了*N*元语法(Jelinek, 1976; Baker, 1975; Bahl et al., 1983)。根据Nádas(1984)的引证，IBM公司的Katz首次把Good-Turing算法应用于*N*元语法的平滑。Jelinek(1990)总结了在IBM公司的语言模型中应用*N*元语法模型以及其他的早期创新性语言模型的情况。

至少在1948年，Jeffreys(1948)就在工程中应用平滑来解决零概率的问题了，但是直到近年来平滑问题才引起了人们的高度关注。Church and Gale(1991)对Good-Turing方法及其证明做了很好的描述，也很好地描述了删除插值法和一种新的平滑方法。Sampson(1996)对Good-Turing算法进行了很有用的讨论。在Gale and Church(1994)中，总结了加1平滑算法的一些问题。在Witten and Bell(1991)的方法C中，描述了Witten-Bell平滑。正如Church和Gale一样，Chen and Goodman(1996)对于不同的平滑算法进行了经验性的比较，其中包括两种新的算法，即平均计数（average-count）算法和一计数（one-count）算法。Iyer and Ostendorf(1997)讨论了从附加的语料库中添加数据来进行平滑的一种平滑方法。

语言模型的一些最新工作重点探讨了建立更复杂的*N*元语法的方法。这些方法包括给当前已经发生的*N*元语法附加权值的方法，即Kuhn and de Mori(1990)的存储LM法（cache LM），选择长距离触发器（long-distance trigger）来替代局部*N*元语法的方法(Rosenfeld, 1996; Niesler and Woodland, 1999; Zhou and Lua, 1998)，使用可变长*N*元语法（variable-length N-gram）的方法(Ney et al., 1994; Kneser, 1996; Niesler and Woodland, 1996)。另外一种类型的方法使用语义信息来丰富*N*元语法，包括第15章中将要讨论的基于潜在语义索引（latent semantic indexing）的语义词联想方法(Coccaro and Jurafsky, 1998; Bellegarda, 1999)，另外还包括从联机词典和类属词典中提取语义信息的方法(Demetriou et al., 1997)。基于类的*N*元语法（Class-based *N*-gram）根据单词的类别（例如词类）来建立*N*元语法，我们将在第8章中讨论有关内容。基于更加结构化的语言知识的语言模型（例如概率剖析）将在第12章中讨论。最后，提升*N*元语法的一系列方法是基于话语知识的，例如使用当前话题的知识来提升*N*元语法(Chen et al., 1998; Seymore and Rosenfeld, 1997; Seymore et al., 1998; Florian and Yarowsky, 1999; Khudanpur and Wu, 1999)或使用言语行为和对话知识来提升*N*元语法（见第19章）。

第7章　HMM与语音识别

When Frederic was a little lad he proved so brave and daring,
His father thought he'd 'prentice him to some career seafaring.
I was, alas! his nurs'rymaid, and so it fell to my lot
To take and bind the promising boy apprentice to a **pilot** —
A life not bad for a hardy lad, though surely not a high lot,
Though I'm a nurse, you might do worse than make your boy a pilot, .
I was a stupid nurs'rymaid, on breakers always steering,
And I did not catch the word aright, through being hard of hearing;
Mistaking my instructions, which within my brain did gyrate,
I took and bound this promising boy apprentice to a **pirate**.

The Pirates of Penzance, Gilbert and Sullivan, 1877[①]

保姆Ruth的这个错误使得Frederic签订了一个很长时间的师徒合同去学习当海盗，后来又由于21岁生日和闰年的问题，把事情弄得更复杂，致使这个师徒合同又增加了大约63年。在Gilbert和Sullivan看来，出现这样的错误是很自然的，正如Ruth后来说的："这两个单词的发音确实太相似了！"。说得一点不错；口语的理解确实是一件非常困难的事情，值得注意的是，人们常常会犯类似的错误。自动语音识别（automatic speech recognition，简称ASR）研究的目标就是用计算机来解决这样的问题，建立语音识别系统，把声学信号映射为单词串。自动语音理解（automatic speech understanding，简称ASU）把这个目标扩展到不仅产生单词，还要产生句子，并且在某种程度上理解这些句子。

任何说话者和任何环境下的语音自动转写的一般问题还远远没有解决。但是，近年来ASR的技术在某些限定的领域内还是可行的，显示了ASR技术的成熟。ASR的一个重要应用领域是人和计算机的交互。很多任务已经可以采用可视的和可指的界面来解决，但对于那些完全用自然语言交际的任务，对于那些不适合使用键盘的任务，语音与键盘相比是一个潜在的较好界面。这些任务主要在手和眼用得比较多的领域，这时用户要用手或眼来操作目标或装备目标，以便控制它们。ASR的另一个应用领域是电话。在这个领域中，语音识别已经在一些方面得到了使用，例如数字输入，

---

① 在这段故事中，保姆Ruth把主人说的pilot（海员）错听为pirate（海盗），从而铸成大错，翻译成中文后已经看不出这种错误了。译文如下：

当Frederic还是一个年轻小伙子的时候，他又勇敢又大胆，
他的父亲想让他当学徒去学习航海。
我是他的保姆，好啊！于是我开始为此事忙起来，
以便使这个有希望的小伙子学习当海员（pilot），
对于这个勇敢的小伙子，这是一种不错的生活，尽管这个职业还不是那么体面。
然而作为保姆的我，却可能帮你做了一件阻碍你的孩子成为海员的蠢事。
我确实是一个笨头笨脑的保姆，我老是把东西打破，
而且因为我有耳聋失聪之症，我总是听不清楚单词的发音，
主人的这个命令在我的脑子里转了又转，
于是我安排这个有希望的小伙子去学习当海盗（pirate）。

识别“yes”以便接收集体呼叫，还有呼叫路径选择，如“Accounting, please”（请结账），“Prof. Regier, Please”（Regier 教授，请）。在某些应用中，结合语音和指示的多模态界面比没有语音的图形用户界面更有效(Cohen et al., 1998)。最后，ASR 正在应用于自动听写（dictation），也就是把一个特定的单独的说话人口授的比较长的独白转写成文字。口授在法律领域使用很普遍，它也可以作为增强交际的一个重要部分（在计算机和那些不能打字或不能说话的残疾人之间进行交互）。Milton（弥尔顿）失明之后，曾经给他女儿口授了 Paradise Lost（失乐园），这已经成为闻名遐迩的佳话。Henry James 在受重伤之后，口授了他晚期的一些小说，这也是众所周知的事实。

语音技术的不同应用需要对问题给予不同的约束，并由此导致了不同的算法。在本章中，我们选择把重点放在大词汇量连续语音识别（Large-Vocabulary Continuous Speech Recognition，简称 LVCSR）这个关键性领域的基础性问题，用一小节来讨论语音合成中的声学问题。一般来说，所谓“大词汇量”的意思是系统包含大约 5000 到 60 000 个单词的词汇，所谓“连续”是指所有单词是自然地一起说出来的。大词汇量连续语音识别与孤立单词语音识别（isolated-word speech recognition）形成了鲜明的对照，在孤立单词语音识别中，每个单词的前面和后面都有停顿。另外，我们将讨论的算法一般是“不依赖于说话人”的（speaker-independent）；这就是说，这些算法可以识别人的语音，而这样的语音是系统在过去从来没有遇见过的。

本章首先回顾语音识别的总体情况，然后介绍 HMM，接着介绍 Viterbi 算法和 A* 算法在解码中的应用，再介绍言语声学和言语特征，以及高斯模型和多层感知器模型（Multi-Layer Perceptron model，简称 MLP）在声学概率计算中的应用。尽管本章依赖于前面的三章，但我们在总结这个领域的许多工作时略去了很多关键性的内容；我们鼓励读者阅读本章后面所介绍的那些有用的教材和文章。本章还用很短的一节介绍了第 4 章中讨论过的语音合成的声学知识。

7.1　语音识别的总体结构

前面几章介绍了在语音识别中使用的一些核心算法。第 4 章介绍了音子和音节的概念。第 5 章介绍了噪声信道模型、贝叶斯规则的应用以及概率自动机。第 6 章介绍了 N 元语法的语言模型和困惑度的计量方法。在本章中，我们将介绍现代语音识别中前面各章没有介绍过的问题，包括隐马尔可夫模型、频谱特征（spectral feature）、用于训练 HMM 的向前 - 向后算法、Viterbi 算法以及用于解决解码问题的栈解码算法（stack decoding algorithm）。栈解码算法又称为 A* 解码算法，能够把音子概率矢量的串映射为单词串。

让我们首先回顾第 5 章中讲过的噪声信道模型。语音识别系统把语音的声学输入看成是源句子的一个噪声“版本”。为了对这个噪声句子进行“解码”，要考虑所有可能的句子。对于每个句子，要计算它生成噪声句子的概率，然后选取概率最大的句子。图 7.1 具体说明了这个“噪声信道”的比喻。

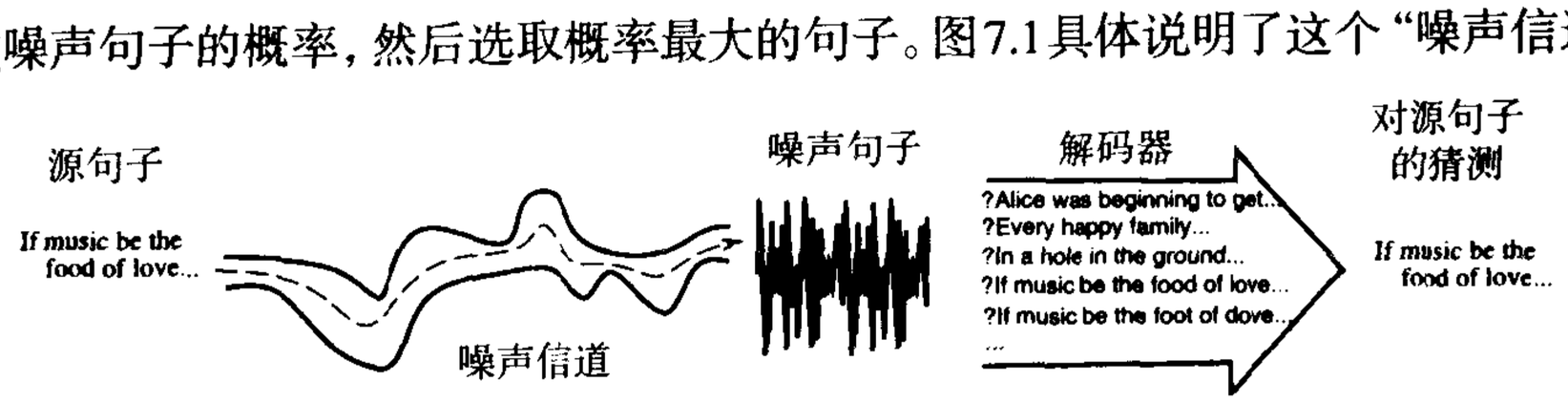


图 7.1　应用于整个句子的噪声信道模型（图 5.1 中的噪声信道模型是应用于个别单词的）。现代语音识别系统的工作就是要搜索一个很大的潜在源句子空间，并选择在生成噪声句子时具有最大概率的句子。为了做到这些，语音识别系统不惜使用各种模型：表示实现为一定的单词串的句子的概率模型（N 元语法），表示实现为一定的音子串的单词的概率模型（HMM），表示实现为声学特征或声谱特征的音子的概率模型（高斯 /MLP）

如图7.1所示，建立噪声信道模型需要解决两个问题。第一个问题是，为了挑选出与噪声输入匹配的最佳句子，需要对“最佳匹配”有一个完全的度量。因为语音是变化多端的，一个声学输入句子不可能与这个句子的任何模型都匹配得天衣无缝。在前面几章中已经指出，我们将使用概率作为度量，并且说明了如何把不同的概率估计结合起来，以便对给定的候选句子的噪声观察序列的概率得到一个完全的估计。第二个问题是，因为所有英语句子的集合非常大，我们需要一个有效的算法，使得不必对所有可能的句子都进行搜索，而只搜索那些有机会与输入匹配的句子。这就是解码问题或搜索问题，我们将总结两种方法：Viterbi解码算法或动态规划算法，栈解码算法或A*解码算法。

我们还将介绍语音识别的概率模型或贝叶斯模型(在第5章中讨论拼写和发音时已经使用过这种模型，这里将更细致地进行讲解)，并将在7.3节中介绍解码和搜索。

语音识别的概率噪声信道总体结构的目标如下：

“对于给定的某个声学输入O，在语言$\mathcal{L}$的所有句子中，哪个句子是最可能的句子？”

我们可以把声学输入O作为单个“符号”或“观察”的序列来处理(例如，把输入按每10微秒切分成音片，每个音片用它的能量或频度的浮点值来表示)。我们用索引号来表示时间间隔，用有顺序的o_i表示在时间上前后连续的输入音片(注意，大写字母表示符号的序列，小写字母表示单个的符号)：

$$O = o_1, o_2, o_3, \cdots, o_t \tag{7.1}$$

类似地，我们在表示句子时，也把它看成是似乎由单词简单地构成的单词串：

$$W = w_1, w_2, w_3, \cdots, w_n \tag{7.2}$$

无论是声学输入还是句子的这种表示，都是简化了的假设；例如，有时把句子切分成单词显得太细(当我们想模拟单词的组合而不是单个词时)，有时又显得太粗(当我们想讨论形态时)。在语音识别中，单词通常是根据正词法来定义的(当把每个单词映射为小写字母以后)：把*oak*与*oaks*当成不同的单词来处理；但是，助动词*can*(“*can you tell me…?*”)与名词can(“*i need a can of …*”)却被当成相同的单词来处理。最近的ASR研究开始注意建立更复杂的ASR单词模型，把第3章中介绍的形态学信息和第8章中将要学习的词类信息也添加到ASR的单词模型中。

前面的直觉概率表示如下：

$$\hat{W} = \underset{W \in \mathcal{L}}{\operatorname{argmax}} P(W|O) \tag{7.3}$$

函数$\operatorname{argmax}_x f(x)$的意思是“使得$f(x)$为最大值的$x$”。式(7.3)能保证给出最优的句子$W$，但现在需要使这个等式运行起来；这就是说，对于给定的句子W和声学序列O，我们需要计算出$P(W|O)$。

我们知道，对于任何给定的概率$P(x|y)$，可以使用贝叶斯规则，把这个概率$P(x|y)$分解如下：

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \tag{7.4}$$

我们在第5章中已经知道，可以用式(7.4)来替换式(7.3)中的有关项，得到：

$$\hat{W} = \underset{W \in \mathcal{L}}{\operatorname{argmax}} \frac{P(O|W)P(W)}{P(O)} \tag{7.5}$$

式(7.5)中右侧的大部分概率与概率$P(W|O)$相比更容易计算。例如，$P(W)$是单词串本身的先验概率，我们可以根据第6章中的N元语法的语言模型进行估计。下面将会看到，$P(O|W)$也很容易估计

出来。但是，声学观察序列的概率$P(O)$却很难估计。不过，正如在第5章中指出的，幸运的是，我们可以忽略$P(O)$。为什么呢？因为我们现在要对所有可能的句子求最大值，我们将对语言中的每个句子计算$\frac{P(O|W)P(W)}{P(O)}$。但是每个句子的$P(O)$是不会改变的！因为对于每个潜在的句子，我们总是要检查同样的观察O，而观察都有同样的概率$P(O)$。因此，我们有：

$$\hat{W} = \underset{W \in \mathcal{L}}{\operatorname{argmax}} \frac{P(O|W)P(W)}{P(O)} = \underset{W \in \mathcal{L}}{\operatorname{argmax}} P(O|W)\,P(W) \tag{7.6}$$

总的来说，对于给定的某个观察O，具有最大概率的句子W可以用每个句子的两个概率的乘积来计算，并且选乘积最大的句子为所求的句子。这两个术语的名称如下：$P(W)$是先验概率，称为语言模型（language model）；$P(O|W)$是观察似然度，称为声学模型（acoustic model）。

关键概念 #5

$$\hat{W} = \underset{W \in \mathcal{L}}{\operatorname{argmax}} \overbrace{P(O|W)}^{\text{似然度}} \overbrace{P(W)}^{\text{先验概率}} \tag{7.7}$$

第6章中已经讨论了如何用N元语法来计算语言模型的先验概率$P(W)$，本章将说明如何计算声学模型$P(O|W)$。首先，为了简化起见，我们假定输入序列是一个音子序列F，而不是一个声学观察序列。我们在第5章中曾经介绍过向前算法，这个算法对于给定的音子序列的观察，能够产生出对于给定单词的这些音子的观察概率。我们将说明，这样的概率音子自动机实际上是隐马尔可夫模型（HMM）的一种特殊情况，并且也将说明如何扩充这个模型，使之对于给定的一个完整句子能够给出音子序列的概率。

然而，正如我们说过的，向前算法遇到的一个问题是：为了发现哪个单词是最可能的单词（即“解码问题”），需要对每个单词再次运行向前算法。对于句子来说，要这样做显然是行不通的，因为无法对英语中的每个可能的句子都分别运行向前算法。因此，这里介绍两种不同的算法，对于给定的句子，它们能同时计算出观察序列的似然度，并且给出最可能的句子。这两种算法是Viterbi算法和 A* 解码算法。

对于一个包含音子串的简化输入，当我们解决了似然度的计算和解码问题之后，将说明如何把同样的算法应用于真实的声学输入而不是事先定义好的音子。为此，我们将简短地介绍声学输入和特征抽出（feature extraction）。所谓特征抽出，就是从输入的声波中提取有意义的特征的过程。然后，我们将介绍从这些特征中计算音子概率的两个标准模型：高斯模型和神经网络模型（neural net model）。神经网络模型又称多层感知器模型（multi-layer perceptron model）。

最后，我们将介绍训练隐马尔可夫模型的标准算法和音子概率估计算法，向前 - 向后算法或Baum-Welch算法(Baum, 1972)，期望最大算法（Expectation-Maximization algorithm，或称EM算法）的一种特殊情况(Dempster et al., 1977)。

作为本章的预览，图7.2是语音识别系统的各个组成部分的大致轮廓。图中说明，一个语音识别系统可以分为三个阶段：信号处理阶段（signal stage）、音子阶段（phone stage）和解码阶段（decoding stage）。信号处理阶段又称为特征抽取阶段，在这个阶段，语音的声学波形切分为音片框架（通常是 10 ms，15 ms 或 20 ms），把音片框架转换成声谱特征，声谱特征要给出不同频度的信号的能量大小的信息。音子阶段又称为亚词阶段（subword stage），在这个阶段，我们使用诸如神经网络或高斯模型这样的统计技术，尝试识别如p或b这样的单个语音。对于神经网络，这个阶段的输出是对于每个音片的音子的概率矢量，例如，对于某个音片，[p]的概率是0.8，[b]的概率是0.1，[f]的概率是0.02，等等；对于高斯模型，概率与此稍有不同。最后，在解码阶段，我们利用单词发

音词典和语言模型（概率语法），采用Viterbi算法或A* 解码算法发现对于给定声学事件具有最大概率的单词序列。

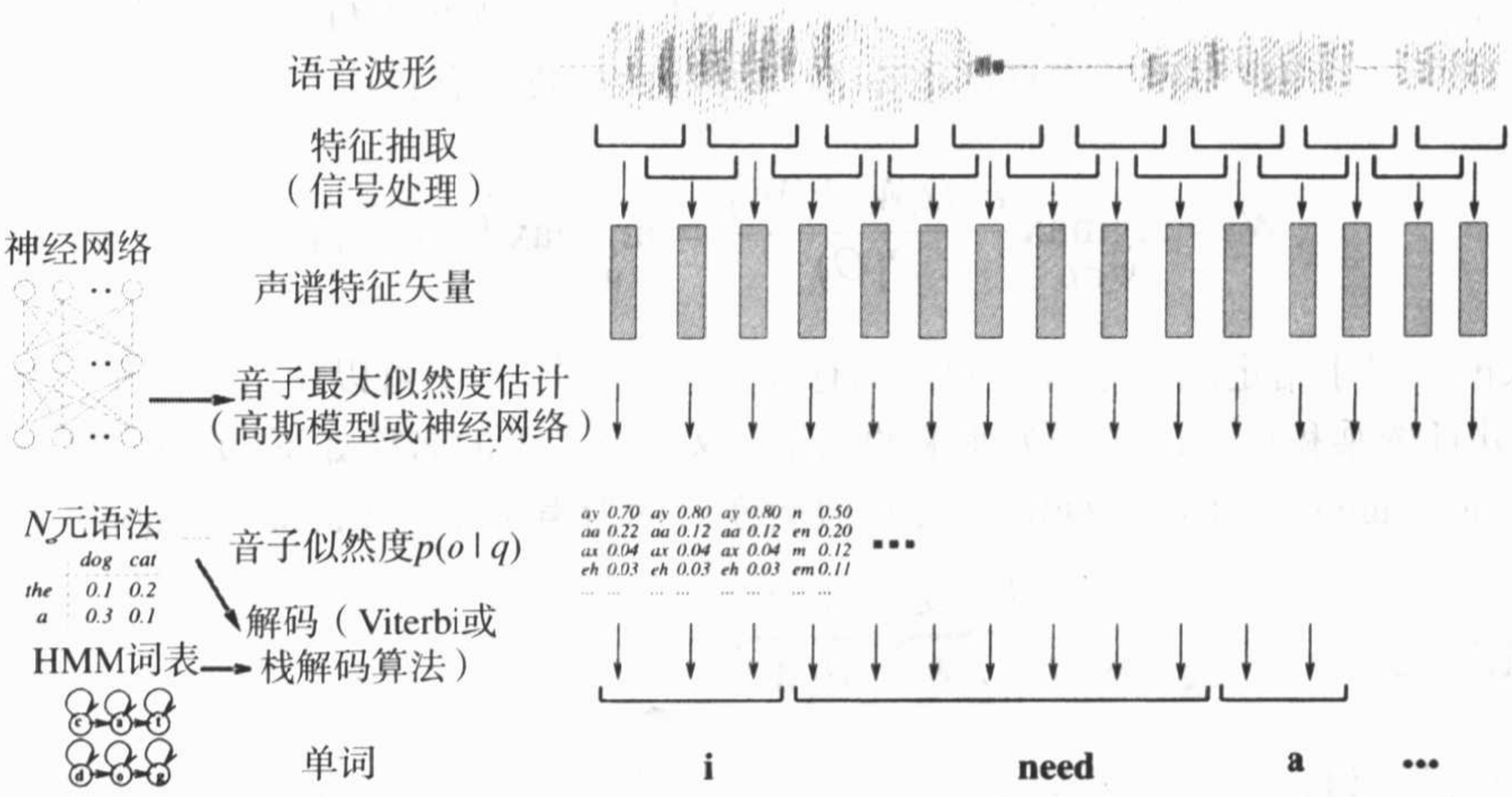


图 7.2 简化的语音识别系统的总体结构图

7.2 隐马尔可夫模型概述

在第5章中，我们曾经使用加权有限状态自动机或马尔可夫链来模拟单词的发音。自动机包括一个状态序列 $q = (q_0\, q_1\, q_2 \cdots\, q_n)$，每个状态对应于一个音子，状态之间的转移概率的集合 a_{01}, a_{12}, a_{13} 用一个音子跟随另一个音子的概率来编码。我们用结点来表示状态，用结点与结点之间的边来表示转移概率。如果在两个结点之间存在非零的转移概率，这两个结点之间就有边。我们知道，我们也可以使用向前算法来计算所观察的音子序列 $o = (o_1\, o_2\, o_3 \cdots\, o_t)$ 的似然度。图 7.3 是单词 *need* 的自动机，这个单词的观察序列在第 5 章中已经讨论过了。

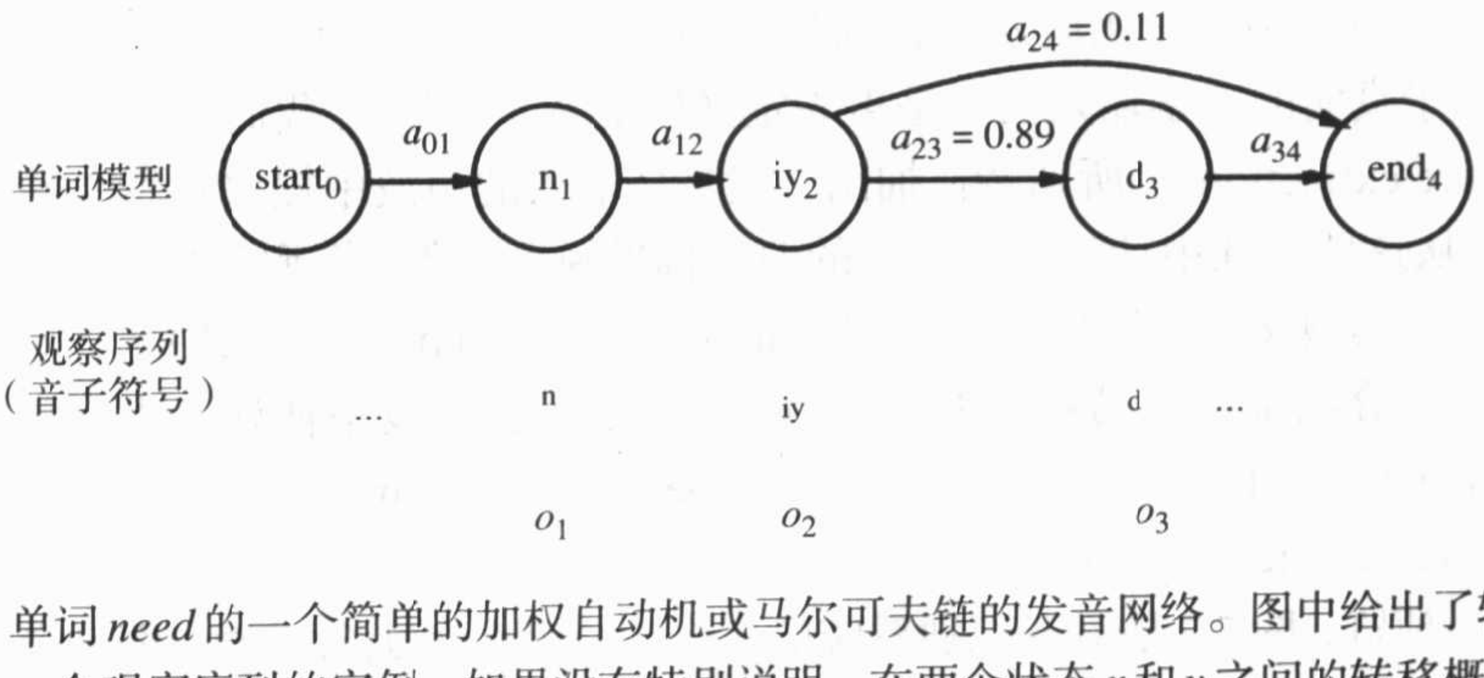


图 7.3 单词 *need* 的一个简单的加权自动机或马尔可夫链的发音网络。图中给出了转移概率和一个观察序列的实例。如果没有特别说明，在两个状态 x 和 y 之间的转移概率 a_{xy} 为 1.0

我们将会看到，这些模型在语音识别中是非常重要的，它们在两个方面简化了问题。首先，这些模型简单地假定输入是由符号序列组成的！显而易见，这样的假定在现实世界中是不对的，在现实世界中的语音输入实质上是由空气粒子的微小运动组成的。在语音识别中，输入是有歧义的，分成音片的输入信号实际值的表示称为特征或声谱特征。我们在后面将要仔细地研究这些特征；声学特征要表示不同的频度所含有的能量大小信息。另一方面的简化是对第5章中的加权自动机的简化，这种简化假定输入符号恰好对应于自动机的状态。因此，当我们看到输入符号[b]时，就知道可以把

标记[b]移动一个状态。与此相反，在隐马尔可夫模型（HMM）中，我们看不到输入符号，不知道要移动到什么状态。输入符号不能惟一地决定下一个状态[①]。

我们知道，加权自动机或马尔可夫链可以用状态集合 Q、转移概率集合 A、定义好的初始状态和终结状态、观察似然度集合 B 来描述。对于加权自动机，如果状态 i 和观察值 o_t 相匹配，则概率 $b_i(o_t)$ 为 1.0；如果不匹配，则概率 $b_i(o_t)$ 为 0。HMM 与马尔可夫模型的不同之处在于，HMM 要增加两个更多的要求。第一个要求是，HMM 有一个观察符号（observation symbol）的集合 O，这个集合中的符号不是从状态集合 Q 的字母中抽取的。第二个要求是，观察似然度函数 B 的值不只限于 1.0 或 0，在 HMM 中概率 $b_i(o_t)$ 可以取 0 ~ 1.0 的任何值。

图 7.4 给出了单词 *need* 和一个观察序列实例的 HMM。注意它和图 7.3 的不同之处。首先，观察序列现在是用于表示语音信号的声谱特征矢量。其次要注意到，我们现在还允许一个状态可以多次重复生成同一观察值，为此在相应的状态上可以加一个自反圈。这个圈使得 HMM 可以模拟音子的可变音延（duration）。音子越长，通过 HMM 时的圈就越多。

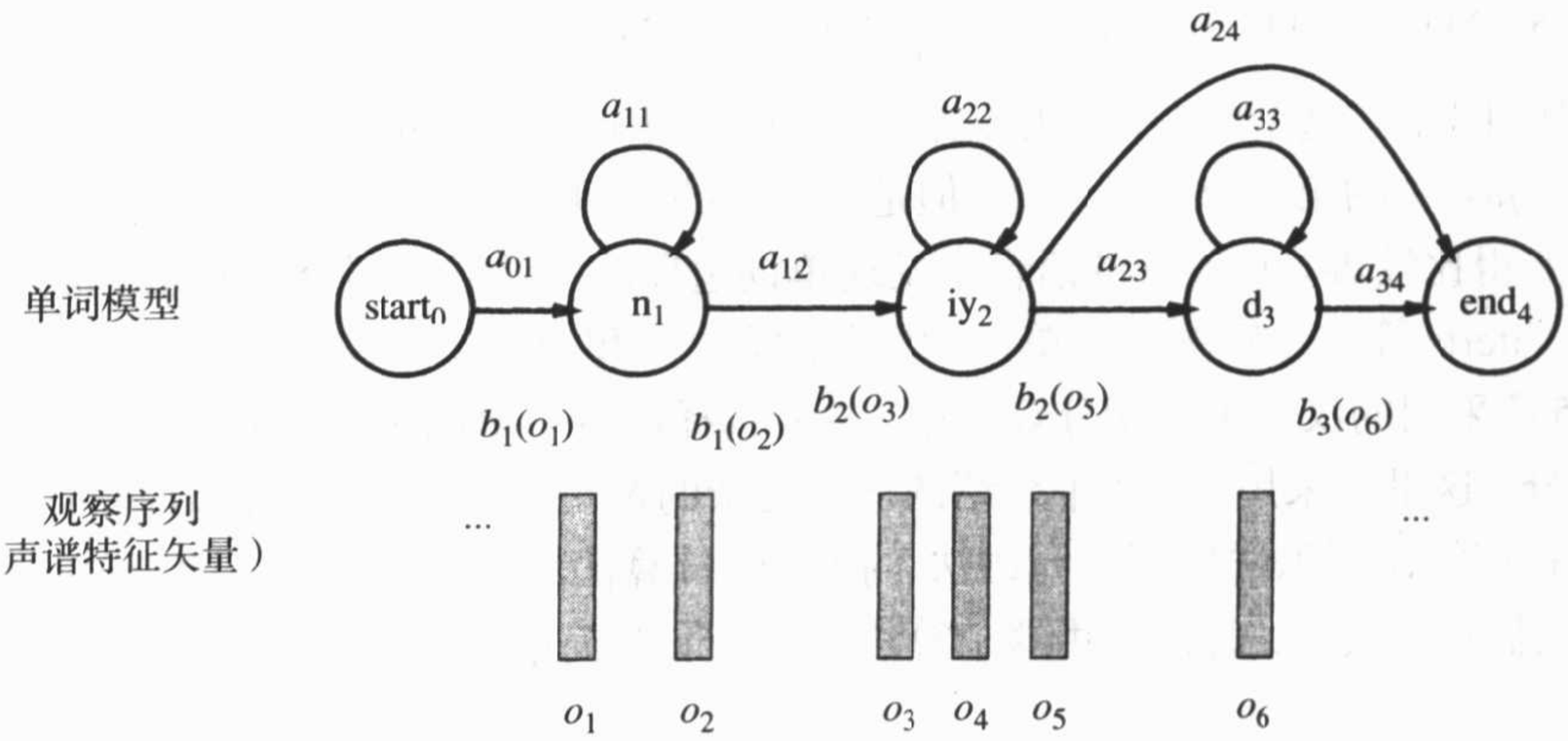


图 7.4　单词 *need* 的 HMM 发音网络，图中给出了转移概率和观察序列的一个实例。注意，这里增加了输出概率 B。在语音识别中使用的 HMM 通常还在状态上加了一个自反圈，以模拟音子的可变音延

总的来说，我们在定义一个 HMM 时，要求的参数如下：

- **状态**　状态的集合 $Q = q_1\, q_2 \cdots q_N$
- **转换概率**　概率集合 $A = a_{01}\, a_{02} \cdots a_{n1} \cdots a_{nn}$。每个 a_{ij} 表示从状态 i 到状态 j 的转换概率。转换概率集合称为转换概率矩阵。
- **观察似然度**　观察似然度集合 $B = b_i(o_t)$，每个观察似然度表示从状态 i 生成的观察值 o_t 的概率。

在前面的例子中，我们曾经使用过两个“特殊”状态（非发射状态），一个是初始状态，一个是终结状态。在第 5 章中讲过，可以避免使用这两个状态，但要说明两件事：

- **初始状态分布**　在状态上的初始概率分布 π，使得 π_i 是 HMM 在状态 i 开始时的概率。显而易见，对于某些状态 j，可能有 $\pi_j = 0$，意味着这些状态不是初始状态。
- **接收状态**　合法的接收状态集合。

① 实际上，正如我们过去指出的，根据第二个标准，对于在第 5 章中研究过的自动机，有的在技术上就是 HMM。这是因为，在输入符号串[n iy]中的第一个符号与单词 *need* 或 *an* 中的状态[n]都是相容的。看到符号[n]，我们不知道在底层状态中这个符号是由 *need-n* 生成的，还是由 *an-n* 生成的。

这个定义对于加权自动机也是适合的。一般来说，输入到模型中的符号序列（如果把它想像成识别机）或由模型产生的符号序列（如果把它想像成生成机）都称为观察序列，记为 $O = (o_1 o_2 o_3 \cdots o_T)$。

7.3 再谈 Viterbi 算法

在第5章中已经介绍过，对于给定的一个自动机，如何使用向前算法计算观察序列的概率，如何使用Viterbi算法发现通过自动机的最佳路径，以及这个最佳路径的观察序列概率。在第5章中，观察序列由单个词组成，但在连续的语音中输入由单词的序列组成，而我们没有单词边界的位置。在第5章中假定已知单词的边界，这样确实把发音问题大大地简化了，因为我们确信[ni]这个发音是从一个单词来的，我们只有7个候选可以进行比较。但是，在实际的话语中并不知道单词边界的位置。例如，我们来给下面的句子解码，这个句子来自 Switchboard 语料库（不要提前偷看答案！）：

[ay d ih s hh er d s ah m thihng ax b aw m uh v ih ng r ih s en l ih]

答案在本页脚注[①]中。这个问题之所以比较难，部分原因是由于协同发音（co-articulation）和快速发音（例如，*just* 中的第一个音子[d]）。但是，主要的原因在于缺乏空间来指出单词的边界，从而使这个问题变得比较难。在连续的语音中发现单词边界的问题称为切分（segmentation），在第5章中曾经使用 Viterbi 算法来解决汉语单词的切分问题，这里也将采用 Viterbi 算法来解决这样的切分问题。在第5章采用的汉语单词切分算法中，主要选择那些使单词序列具有最大频度的切分作为希望采用的切分。这里将采用第6章中介绍的比较复杂的 N 元语法模型来进行语音的切分。在本节的其他部分，我们将说明怎样使用 N 元语法，把 Viterbi 算法应用于观察音子的简单符号串的切分和解码问题。我们还将说明，怎样使用这个算法来切分非常简单的单词符号串。这里是我们的输入和输出：

输入	输出
[aa n iy dh ax]	*I need the*

图 7.5 是 *I*, *need* 和 *the* 的单词模型，为了稍微复杂一些，还加了一个单词 *on*。

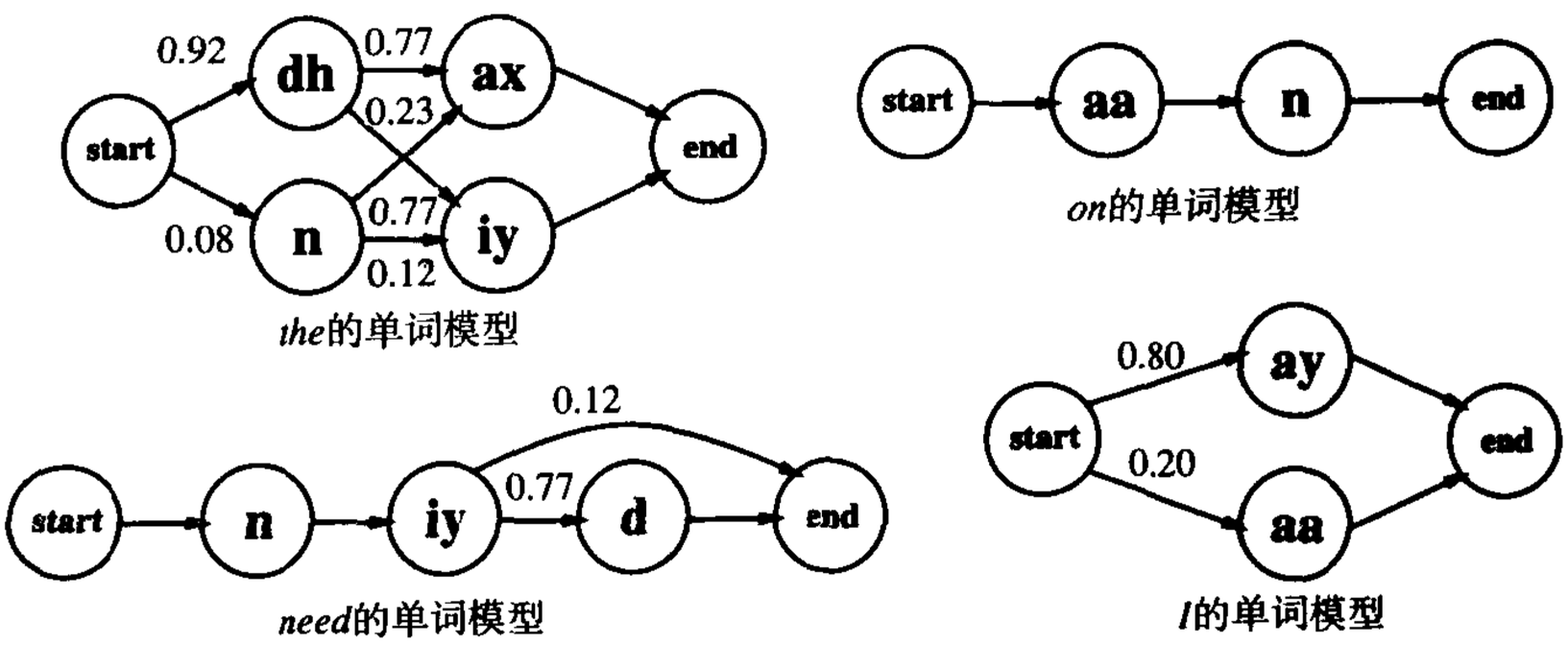


图 7.5 单词 *I*, *on*, *need* 和 *the* 的发音网络。所有的发音网络（特别是单词 *the*）都大大地简化了

① I just heard something about moving recently.（我刚才听到关于目前的电影的一些事情。）

Viterbi 算法的目标是，对于给定的观察音子的集合 $o = (o_1\ o_2\ o_3 \cdots o_t)$，发现最佳的状态序列 $q = (q_1\ q_2\ q_3 \cdots q_t)$。动态规划算法的输出如图 7.6 所示。沿着 y 轴是词表中的所有单词，在每个单词项目的内部是其状态。x 轴是按时间排列的，每个时间单位有一个观察音子[①]。在矩阵的每个单元中还包含在该状态的最佳序列结尾的概率。对于整个观察符号串，我们可以通过查找在最右的列中具有最大概率的单元的办法，发现最佳的状态序列，并跟踪产生它的序列。

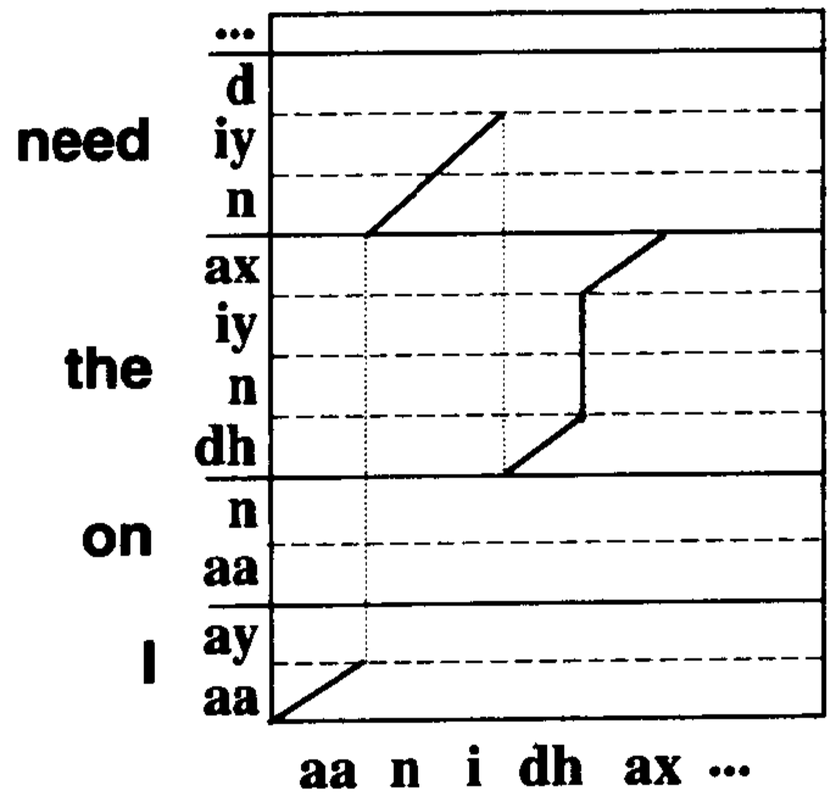


图 7.6　用于发现最佳音子序列的 Viterbi 算法的结果（这个算法也可以估计最佳的单词序列）

更形式化的说法是，对于给定的观察序列 $o = (o_1\ o_2 \cdots o_T)$和模型 λ（加权自动机或状态图），我们要搜索最佳状态序列 $q = (q_1\ q_2 \cdots q_T)$。矩阵中的每个单元 *viterbi*[*i*, *t*]包含着前 t 个观察和在 HMM 的状态 i 结束的最佳路径的概率。这是在长度为 t–1 的所有可能状态序列中的概率最大的路径：

$$viterbi[t,i] = \max_{q_1,q_2,\cdots,q_{t-1}} P(q_1\ q_2 \cdots q_{t-1}, q_t = i, o_1, o_2, \cdots, o_t|\lambda) \tag{7.8}$$

为了计算 *viterbi*[*t*, *i*]，Viterbi 算法假定“动态规划恒定”（dynamic programming invariant）。这是一个简化的（但并不正确）假定。这种假定认为，如果对于全部的观察序列，最终的最佳路径恰好通过了状态 q_i，这条最佳路径必定是在此之前包括状态 q_i 在内的所有路径中的最佳路径。这并不意味着在任何时刻 t 的最佳路径就是所有序列中的最佳路径。一条路径在开始时看起来可能并不好，但后来它可能会变成最佳路径。我们在后面将会看到，对于某些类别的语法（包括三元语法），Viterbi 假定会遭到失败，所以有些语音识别系统转而使用其他解码算法，如栈解码算法或 A* 解码算法等。在第 5 章中讨论最小编辑距离算法时就可以看出，使用 Viterbi 假定就是为了使我们能够用一种简单办法来分解最佳路径概率的计算；在时刻 t 的每条最佳路径就是在时刻 t–1 终结的每条路径的最佳扩充。换言之，对于在状态 j 和在时刻 t 结尾的最佳路径，*viterbi*[*i*, *j*]就是从时刻 t–1 到时刻 t 的前面每种可能路径的一切可能扩充的最大值：

$$viterbi[t,j] = \max_i (viterbi[t-1,i]\, a_{ij})\, b_j(o_t) \tag{7.9}$$

在图 7.9 描述的算法中，取一个观察序列，一个单独的概率自动机，然后通过自动机返回最佳路径。因为这个算法要求一个单独的自动机，所以我们需要把 *the*, *I*, *need* 和 *on* 等不同的概率音子

① 该模型的 x 轴中的组成成分在两个主要的方面被简化了，下一节将对此进行说明。首先，观察值不是音子而应该是被抽取的声谱特征；其次，每个音子不是由时间单位的观察组成的，而是由很多观察组成的（因为音子可以在时间上延长而变得比一个音子更长）。在这个例子中，y 轴也被简化了，因为我们将会看到，在大多数 ASR 系统中每个音子都使用了多个“次音子”（subphone）单位。

网络结合到一个自动机中。为了建立这个新的自动机，我们需要在每两个单词之间加上一个弧，而且在弧上标出概率（二元语法概率）。图7.7是把Brown语料库和Switchboard语料库结合起来计算出的一些简单的二元语法概率：

I need	0.0016	need need	0.000047	#Need	0.000018
I the	0.00018	need the	0.012	#The	0.016
I on	0.000047	need on	0.000047	#On	0.00077
I I	0.039	need I	0.000016	#I	0.079
the need	0.00051	on need	0.000055		
the the	0.0099	on the	0.094		
the on	0.00022	on on	0.0031		
the I	0.00051	on I	0.00085		

图7.7 单词*the*, *on*, *need*和*I*彼此相续形成的二元语法概率，以及它们作为句子开头时（也就是在#之后）的二元语法概率。计算时把Brown语料库和Switchboard语料库结合起来，并采用了加0.5的平滑

图7.8说明了把这4个单词彼此结合而成的二元语法的发音网络，网络中增加了一些新的弧。为了便于读这个图，大多数弧都没有画出来，读者可以根据图7.7中的各个概率来想像，把相应的概率插入到每两个单词之间。

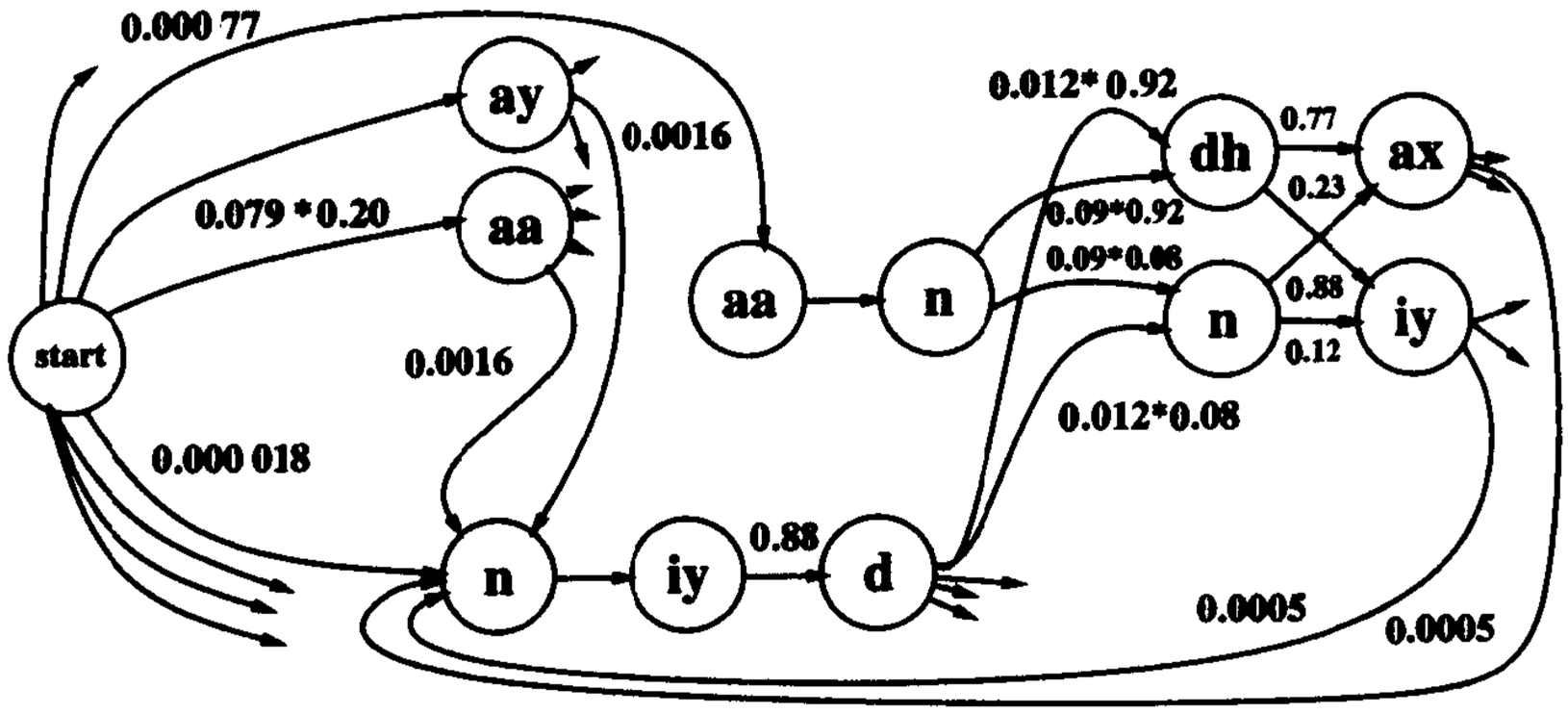


图7.8 表示单词*I*, *need*, *on*和*the*的简化的自动机。单词之间的弧上标出的概率来自图7.7。由于空间不够，本图中只画出了很少的几个弧来表示单词之间的转移概率

在图5.19中曾给出了一个算法，为方便起见，我们把它复制在图7.9中。我们在图7.9中可以看出，Viterbi算法建立了一个概率矩阵，矩阵的列表示每个时刻的索引号t，矩阵的行表示状态图中的每个状态。算法首先建立了T+2个列；图7.9中说明了前6列的情况。第一列是初始的伪观察值，下面的各个列依次对应于第一个观察音子[aa]，第二个观察音子[n]，依次类推。我们从第一列开始，置开始状态的概率为1.0，置其他概率为0（见图7.10）。为便于阅读，概率为0的单元都用空白表示。对于矩阵中的每个列，也就是对于每个时间索引t，每个单元*viterbi*[t, j]将包含通向该单元终点的最大可能路径的概率。我们采用在前面所有可能状态来的概率中求最大值的方法，递归地计算这个概率。然后转入到下一个状态，根据式（7.9）中的递归关系，对于列0中*viterbi*[0, i]的每个状态i，我们计算它转入列1中*viterbi*[1, j]的每个状态j的概率。在表示输入*aa*的列中，只有两个项目具有非零的值，因为除了两个标记为*aa*的状态之外，其他所有状态的$b_1(aa)$都是0。单词*I*的*viterbi*(1, *aa*)的值等于从#到*I*的转移概率与*I*的发音为元音*aa*的概率的乘积。

```
function VITERBI(observations of len T,state-graph) returns best-path

  num-states ← NUM-OF-STATES(state-graph)
  Create a path probability matrix viterbi[num-states+2,T+2]
  viterbi[0,0] ← 1.0
  for each time step t from 0 to T do
    for each state s from 0 to num-states do
      for each transition s' from s specified by state-graph
        new-score ← viterbi[s, t] * a[s,s'] * b_s'(o_t)
        if ((viterbi[s',t+1] = 0) || (new-score > viterbi[s', t+1]))
        then
              viterbi[s', t+1] ← new-score
              back-pointer[s', t+1] ← s
  Backtrace from highest probability state in the final column of viterbi[] and
return path.
```

图 7.9　用于发现连续语音识别中的最优状态序列的 Viterbi 算法，为简单起见，使用音子作为输入（从图5.19复制得来）。给出音子的观察序列和一个加权自动机（状态图），算法返回具有最大概率的自动机的路径，并接收观察序列。$a[s, s']$表示从当前状态 s 到下一个状态 s' 的转移概率，$b_{s'}(o_t)$表示对于给定的 o_t，s' 的观察似然度

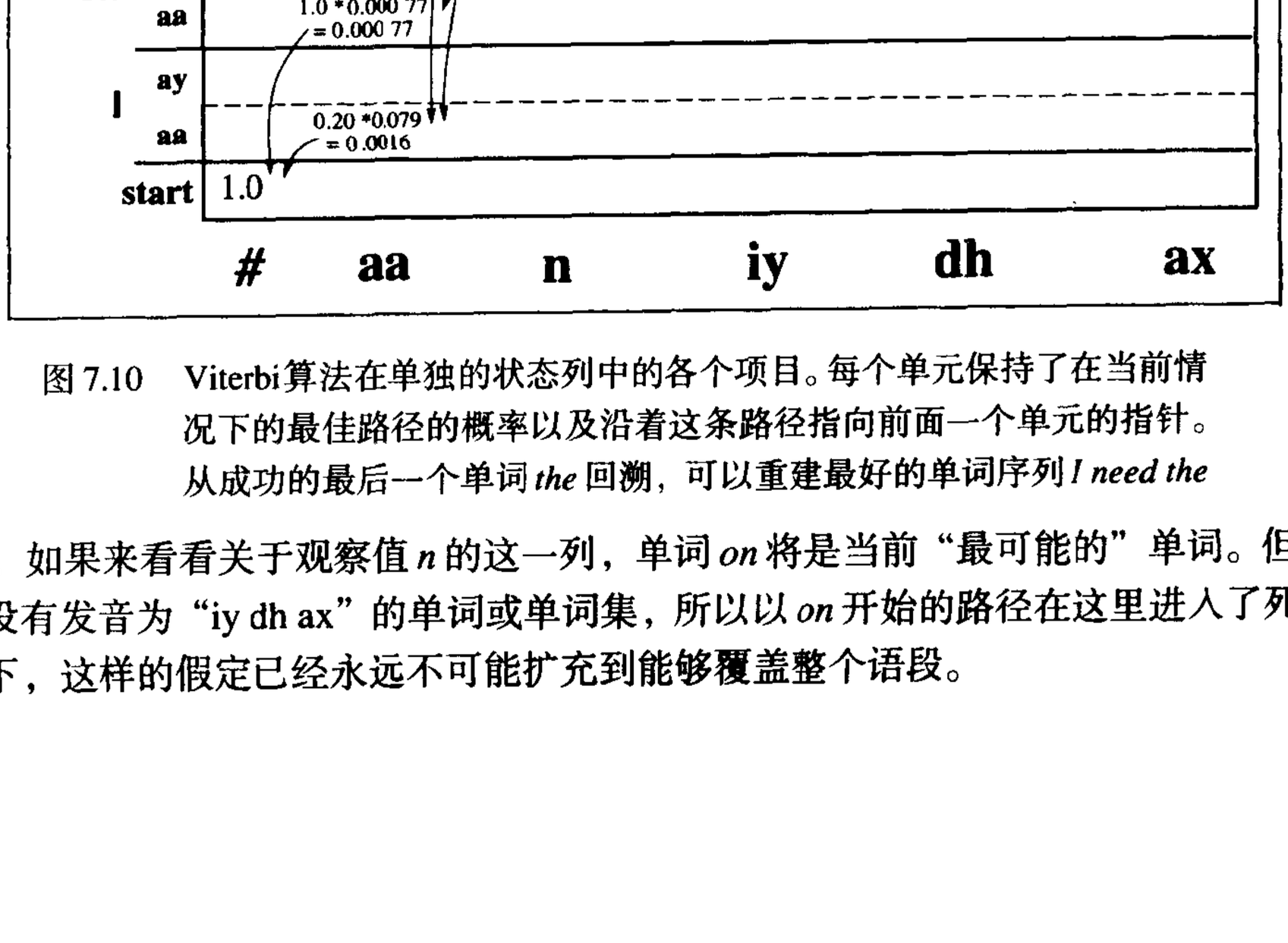


图 7.10　Viterbi算法在单独的状态列中的各个项目。每个单元保持了在当前情况下的最佳路径的概率以及沿着这条路径指向前面一个单元的指针。从成功的最后一个单词 *the* 回溯，可以重建最好的单词序列 *I need the*

注意，如果来看看关于观察值 *n* 的这一列，单词 *on* 将是当前“最可能的”单词。但是，由于在词表中没有发音为“iy dh ax”的单词或单词集，所以以 *on* 开始的路径在这里进入了死胡同。在这种情况下，这样的假定已经永远不可能扩充到能够覆盖整个语段。

当我们看到观察值*iy*时，存在着两条候选路径：*I need*和*I the*；*I need*将是当前最可能的路径。当达到观察值*dh*时，可能从表示*need*的*iy*达到*dh*，也可能从表示*the*的*iy*达到*dh*。在这种情况下，通过*I need*的路径进入表示*dh*的单元的概率，是这两条路径中的最大概率。

最后，导致最佳路径的概率将在最终的*ax*这一列出现。在这个例子中，最终的这个列只有一个单元是非零单元，这就是单词*the*的状态*ax*（实际的例子不会这么简单，其他很多单元将不是非零单元）。

如果句子到这里已经真正结束了，就必须再找出得到这个概率的路径。我们不能只在每列中挑出概率最高的状态。为什么不能呢？因为在前面找出的最可能的路径不一定就是整个句子的最可能的路径。我们还记得，在看到*n*之后，最可能的路径是单词on。但是，这个句子的实际上最可能的路径却是*I need the*。因此，在图7.10中还不得不依靠“Hansel和Gretel”方法（如果你希望比喻得更有经典色彩，也可以说“Jason和Minoitaur”方法）：“每当我们进入一个单元时，一定要保留回到我们所来的这个单元的指针。”读者应该相信，Viterbi算法也可以用来解决切分和解码的问题。

本节中介绍的Viterbi算法是简化过的；实现一个真正的Viterbi解码算法要复杂得多，我们曾经指出，有三个关键方法可以充实这个简化Viterbi算法。

首先，在一个实际的语音识别系统的HMM中，输入的不是音子，而是声谱特征或声学特征的特征矢量（feature vector）。因此，对于给定的状态i，观察值o_t的观察似然概率（observation likelihood probability）$b_i(t)$将不会简单地取值为0或1，而是要给出一个颗粒度更细的概率估计，这个概率估计可以通过高斯概率估计或神经网络混合方法来计算。下一节将介绍如何计算这样的概率。

其次，在大多数语音识别系统中，HMM的状态不是简单的音子，而是次音子（subphone）。在这些系统中，每个音子被分解成三个状态：该音子的开始部分、中间部分和最后部分。用这样的办法对音子进行切分更好地反映了我们的直觉，这是在声学输入方面的很有意义的改变，它比音子具有更细的颗粒度。例如，把塞辅音分解为成阻和除阻等部分。很多系统使用三音子上下文（triphone context）来分解次音子(Schwartz et al., 1985; Deng et al., 1990)。这样，就不会是大约60个音子单位，而可能会有60^3之多的三音子了。实际上许多可能的音子序列从来不会出现或很少出现，所以有的系统采用把可能的三音子聚类（clustering）的办法来建立数量比较小的三音子模型(Young and Woodland, 1994)。图7.11说明了一个完整音子的三音子模型b(ax, aw)。

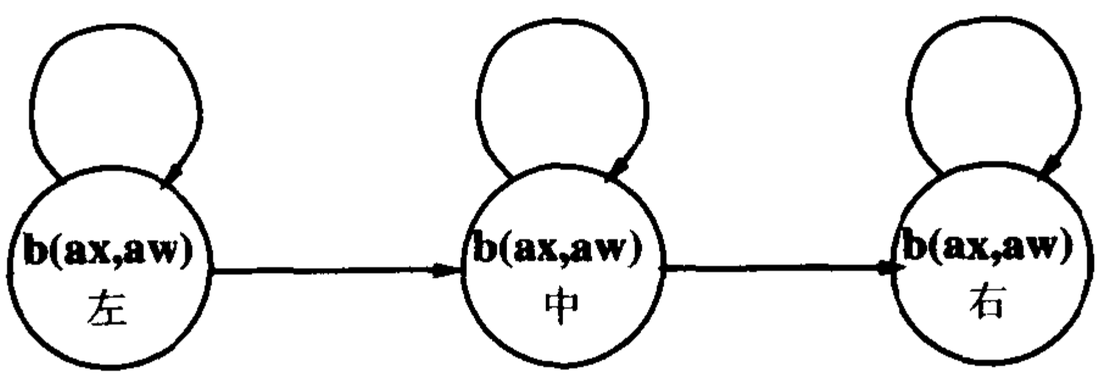


图7.11 依赖于上下文的三音子b(ax, aw)的一个例子，音子[b]前面是[ax]，后面是[aw]，正如*about*这个单词的开头那样，图中显示了它的左次音子、中次音子和右次音子

最后，在大容量词汇识别的实际应用中，当算法的延伸路径从一个状态列到另一个状态列时，如果考虑所有可能的单词，系统的成本将会很高。我们不用这样的办法。在每个时间步（time step）对低概率的路径进行修剪，而不让它延伸到下一个状态列。这通常可以使用定向束搜索（beam search）的方法来实现：对于每个状态列（时间步），算法要维护一个比较短的表，在这个短表（short list）中只记录路径概率处于最可能单词路径的某个百分比（定向束宽度）之内的高概率单词。当向下一个时间步转移时，只有从这些单词的转移才能够向前延伸。由于单词是按照路径概率的大小排

列的，在定向束内（激活）的单词，就可以从一个时间步向另一个时间步转移。采用这种定向束的近似方法，显著地加快了系统的速度，减轻了由于解码而降低系统性能的开销。这种定向束的策略是由Lowerre(1968)首先实现的。因为在大多数Viterbi算法的应用中都使用了定向束搜索的策略，所以在某些文献中使用定向束搜索或时间同步的定向束搜索（time-synchronous beam search）之类的术语来替代 Viterbi 算法这个术语。

7.4　先进的解码方法

Viterbi 解码算法有两个主要的限制。我们首先来讨论第一个限制。对于给定的声学输入，Viterbi解码算法实际上不计算具有最大概率的单词序列，而是计算与这样的单词序列近似的、对于给定的输入具有最大概率的状态序列（也就是音子序列或次音子序列）。这样的差别不总是重要的，概率最大的音子序列可能与概率最大的单词序列对应得非常好。但是，有时概率最大的音子序列并不对应于概率最大的单词序列。例如，在一个语音识别系统中，当词表中的每个单词都有多个发音时，就会出现音子序列与单词序列不对应的情况。假定正确的单词序列包含一个具有很多发音的单词，由于离开每个单词的开始弧的概率之和应该为 1，所以通过这个具有多个发音的HMM单词模型的发音路径的单词的概率，将小于通过只有一个发音路径的单词的概率。在这种情况下，因为Viterbi解码算法只能从这些不同的发音路径中选择一条，所以会忽略这个具有不同发音的单词，而支持只有一个发音路径的不正确的单词。

Viterbi 解码算法的第二个限制是，它不能用于所有可能的语言模型。在事实上，在我们定义Viterbi算法时，它就不能完全采用比二元语法模型更复杂的其他语言模型的优点。其原因在于前面已经提过的事实，例如，三元语法破坏了动态规划恒定的假定，而动态规划恒定才使动态规划算法成为可能。前面说过，动态规划恒定是一个简化的（但不正确的）假定。动态规划恒定假设，如果整个观察序列的最终的最佳路径恰好通过了状态q_i，那么这个最佳路径一定包含一直到状态q_i之前并且也包含状态q_i本身在内的最佳路径。由于在三元语法中一个单词的概率要根据前面两个单词来决定，一个句子的最佳三元语法概率的路径就可能会通过没有包括在该单词的最佳路径中的某个单词。对于单词w_y和w_z，当一个特定的单词w_x具有较高的三元语法概率，但w_y的最佳路径却不包含w_z时，即对于一切q，$P(w_y|w_q, w_z)$比较低时，就会发生这样的情况。

对于 Viterbi 解码算法的这些问题，存在着两类解决办法。一类解决办法是修改 Viterbi 解码算法，返回多个潜在的语段，然后再使用其他高水平的语言模型或发音模型算法，重新给多个输出排序。一般来说，这种多遍解码（multiple-pass decoding）的方法在计算上是有效的，但如果先使用二元语法这样的不太复杂的语言模型来进行第一遍粗解码，然后再使用更复杂但比较慢的解码算法继续工作，就可以减少搜索空间。

例如，Schwartz and Chow(1990)提出了一种类似于 Viterbi 的算法，称为 *N*- 最佳 Viterbi 算法。对于给定的语音输入，这种算法返回*N*个最佳（*N*-best）的句子（单词序列）。例如，假定我们使用二元语法，这种 *N*- 最佳Viterbi 算法将返回 10 000 个概率最高的句子，每个句子带有它们的似然度打分。然后，使用三元语法给每个句子指派一个新语言模型的先验概率。这些先验概率与每个句子的声学似然度相结合，生成每个句子的后验概率（posterior probability）。然后，使用这种更复杂的概率重新给句子打分。图 7.12 是这个算法的直觉说明。

第一类解决办法中还有一种方法，也是用*N*- 最佳（*N*-best）的办法来提升 Viterbi 算法，但返回的不是一个句子表，而是一个单词格（word lattice）。单词格是单词的有向图（directed graph），单词之间用单词格连接之后就可以对大量的句子进行紧致的编码。在格中的每个单词使用它们的

观察似然度来扩充，这样，通过格的任何路径都可以与更复杂的语言模型中推导出的先验概率结合起来。例如，Murveil et al.(1993)描写了一个在SRI语音识别系统Decipher中使用的算法，这个算法在第一遍粗解码中使用二元语法，产生单词格，然后使用更复杂的语言模型来精心地改进这个单词格。

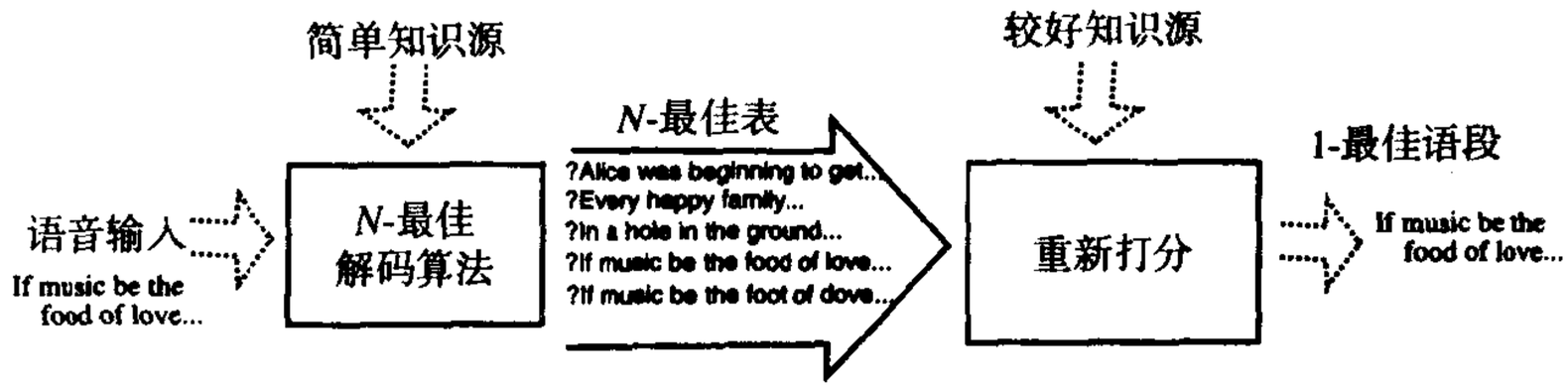


图 7.12 把*N*-最佳解码算法用来作为两阶段解码模型的一部分。使用有效但并不复杂的知识源返回*N*-最佳语段。这样就明显减小了第二遍解码模型的搜索空间，使它不至于太复杂，也不至于太慢

Viterbi解码算法问题的第二类解决办法是使用完全不同的解码算法。在这些不同的算法中，最常见的是栈解码算法，又称为A*解码算法(Jelinek, 1969; Jelinek et al., 1975)。我们将使用A*搜索（A* search)这个术语来描述这种算法，这是在人工智能文献中使用的术语；栈解码算法的研究来自通信理论的文献，它和人工智能中的最佳优先搜索（best-first search）的结合是后来才被注意到的(Jelinek, 1976)。

7.4.1 A*解码算法

为了理解A*解码算法是怎样工作的，我们需要再谈一谈Viterbi算法。我们知道，Viterbi算法计算的是向前算法的近似值。Viterbi算法计算通过HMM的一个最佳（MAX）路径的观察似然度，而向前算法则计算通过HMM的所有路径的总和（SUM）的观察似然度。我们之所以接受这样的近似值，是因为Viterbi算法在计算似然度时同时还搜索最佳路径。A*解码算法与Viterbi算法不同，它将依靠完全的向前算法而不依靠近似值。另外，A*解码算法还允许我们使用任何语言模型。

A*解码算法是对于格（lattice）和树（tree）的一种最佳优先搜索，而格和树隐含地定义了一种语言中可允许单词的序列。考虑图7.13中的树。这个树的根在左边的START结点上。这个树中的每条路径定义了该语言的一个句子；沿着从START到叶子的路径，把路径中所有的单词毗连起来，就可以形成一个句子。我们这里对于树的表示不很明显，但栈解码算法隐含地使用树作为构造解码搜索的一种手段。

算法从树的根开始向叶子进行搜索，查找概率最大的路径，而概率最大的路径就代表概率最大的句子。当我们从根向叶子进行搜索时，离开给定单词的结点的每个枝所表示的单词，可能跟随在这个当前给定的单词之后。每个这样的枝上都有概率，这个概率表示在我们前面所看到句子给定部分的条件之下，下面一个单词出现的条件概率。此外，我们将使用向前算法给每个单词指派一个产生所观察声学数据的某个部分的似然度。因此，A*解码算法必须找出从根到概率最大的叶子之间的路径（单词序列），而该路径的概率可以由语言模型的先验概率和它与声学数据匹配的似然度的乘积来确定。这可以通过保持部分路径优先队列（priority queue）的办法来实现。这个优先队列也就是句子中带有分数（score）标注的前面部分（prefix of sentence）。在一个优先队列中，每个成分都打了一个分数，上托（pop）操作返回分数高的成分。A*解码算法反复地选择最佳的句子前面部分，对于这个部分，计算它后面所有可能出现的下一个单词，把句子加以延伸，并把这些延伸了的句子加到优先队列中。图7.14给出了一个完全的算法。

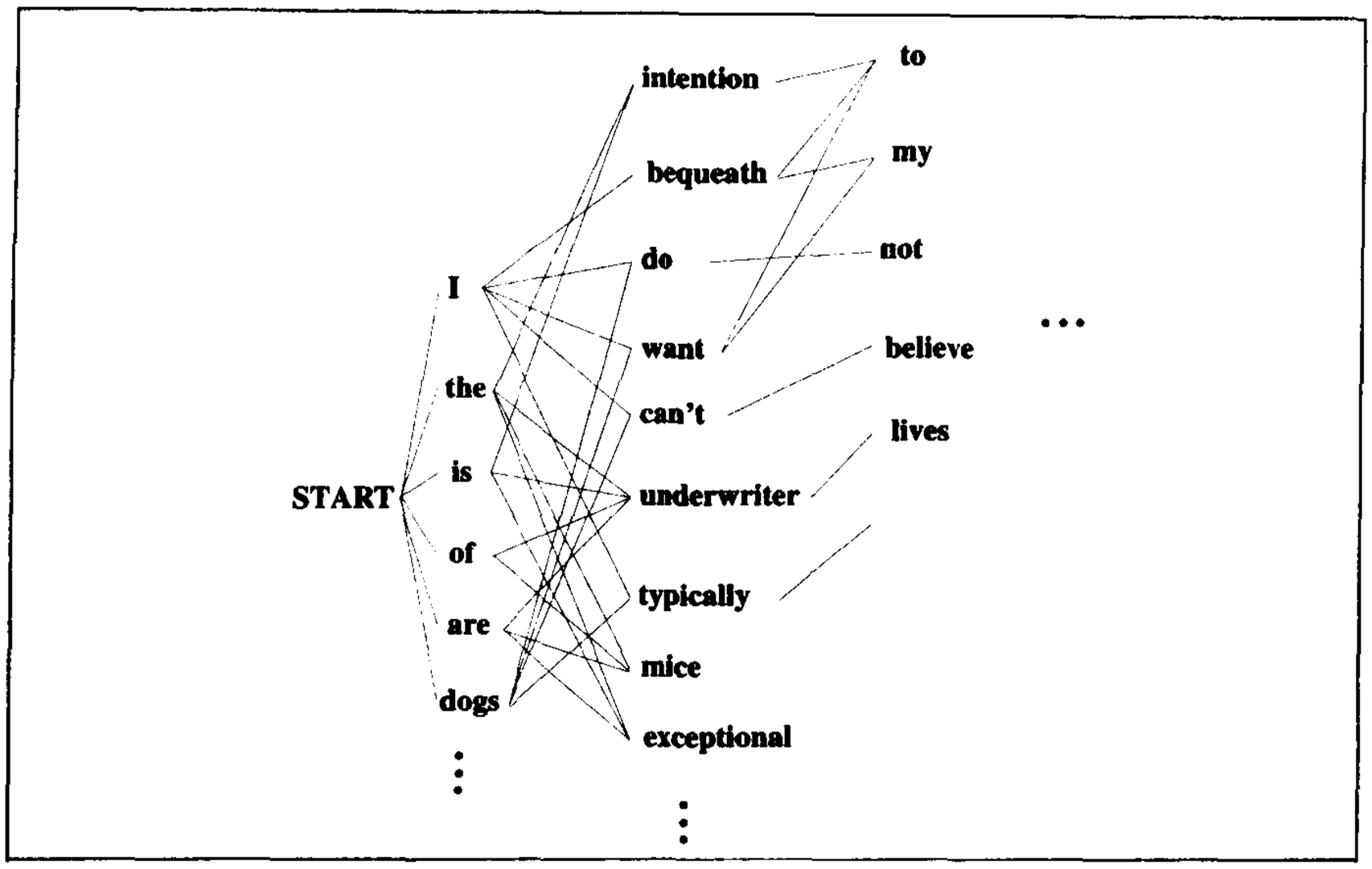


图 7.13　定义一种语言的可容许单词序列的隐含格的可视表示。一种语言中句子的集合很大，不可能明显地表示出来，但这个格作为一个比喻可以帮助我们探索这些句子的各种子符号串

function STACK-DECODING() **returns** *min-distance*

Initialize the priority queue with a null sentence.
Pop the best (highest score) sentence s off the queue.
If (s is marked end-of-sentence (EOS)) output s and terminate.
Get list of candidate next words by doing fast matches.
For each candidate next word w:
　　Create a new candidate sentence $s + w$.
　　Use forward algorithm to compute acoustic likelihood L of $s + w$
　　Compute language model probability P of extended sentence $s + w$
　　Compute "score" for $s + w$ (a function of L, P, and ???)
　　if (end-of-sentence) set EOS flag for $s + w$.
　　Insert $s + w$ into the queue together with its score and EOS flag

图 7.14　A*解码算法(Paul, 1991; Jelinek, 1997)修改得到。这里没有完全地定义用于计算句子分数的评估函数；可能的评估函数将在下面讨论

我们来研究A*解码算法的一个追求时尚的例子，这个例子处理的波形所对应的正确转写是半句时髦话："*If music be the food of love*"（如果音乐是爱情的食粮）。图 7.15 说明了解码算法检查了从根开始的第一段长度为 1 的路径之后的搜索空间的情况。我们使用快速匹配（fast match）的办法来选择下面一个或多个最可能的单词。快速匹配是一种试探性的方法，用于筛选下面可能的单词的数目。在通常的情况下，要计算出前面概率的近似值（参看后面对快速匹配的讨论）。

在例子中的这个点上，我们使用快速匹配的办法，从下面的可能单词中选出一个子集合，并且给每个选出的单词打分。我们还没有讲怎样来精确地计算分数。大致来说，对于给定的声学输入，这个分数应该是所假定的句子的概率的一个组成部分 $P(W \mid A)$，它本身是由语言模型的概率 $P(W)$ 和声学似然度 $P(A \mid W)$构成的。

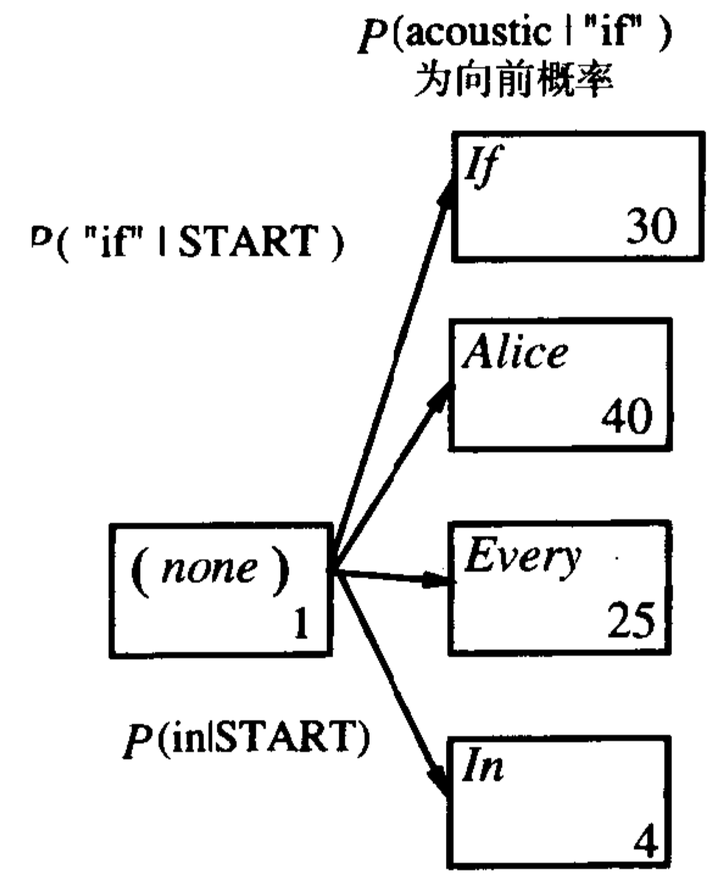


图 7.15　搜索句子 *If music be the food of love* 开始时的搜索空间。在这个开始阶段，*Alice* 是最可能的假设（与其他假设相比，它的分数最高）

图 7.16 说明了搜索的下一个阶段。我们把结点 *Alice* 向前延伸，这意味着 *Alice* 不再处于队列中，但是它的后继单词进入了队列。注意，这时标记为 *if* 的结点成为了分数最高的结点，所得的分数比 *Alice* 的所有后继结点都高。

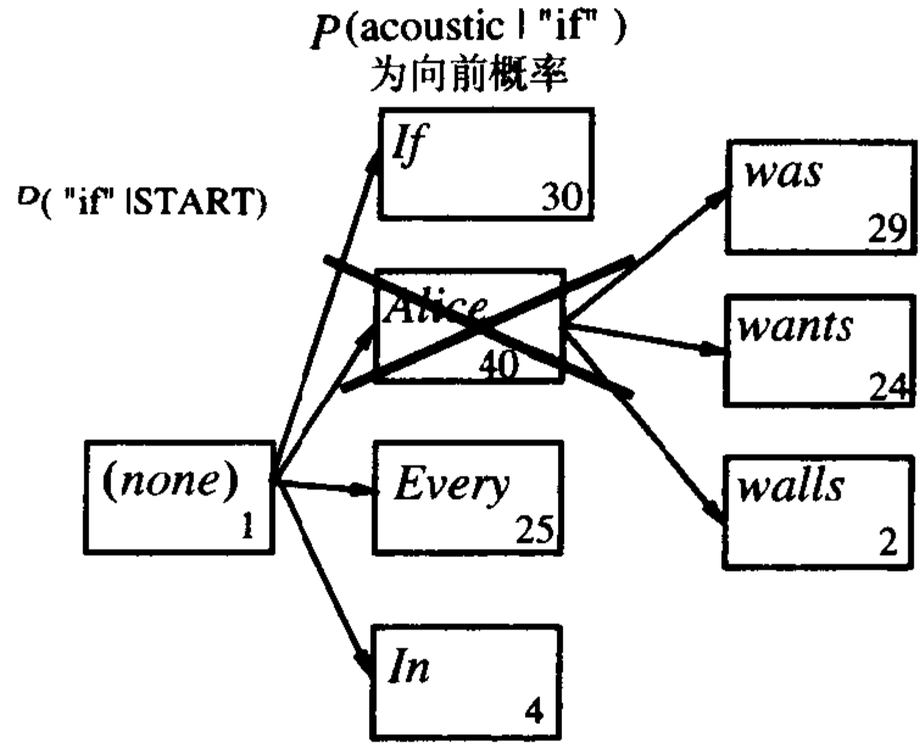


图 7.16　搜索句子 *If music be the food of love* 的下一个阶段。现在我们延伸结点 Alice，并且把它的三个分数比较高的后继（*was*, *wants* 和 *walls*）加入队列中，注意，现在分数最高的结点是 *START if*，顺着 *START Alice* 的这条路径已经不存在了

图7.17说明了在延伸了结点if之后的搜索状态，这时if被移走，队列中增加了 *if music*, *if muscle* 和 *if messy*。

前面提到，给一个假设打分的标准与其概率有关。我们现在来具体说明这个问题。对于给定的声学符号串 y_1^j，单词串 w_1^i 的分数似乎应该等于先验概率和似然度的乘积：

$$P(y_1^j|w_1^i)P(w_1^i)$$

遗憾的是，这样计算出来的概率不能作为打分的分数，因为如果这样计算，越长的路径概率会越小，而越短的路径概率会越大。这是出于概率和子符号串的简单事实；这样，符号串的任何前面的部分将会比符号串本身具有更大的概率，例如 P(START the ....)将会大于 P(START the book)。在

这种情况下，如果我们采用这个概率作为分数，在遇到单个词的假设时，A* 解码算法将会停滞不前，束手无策。

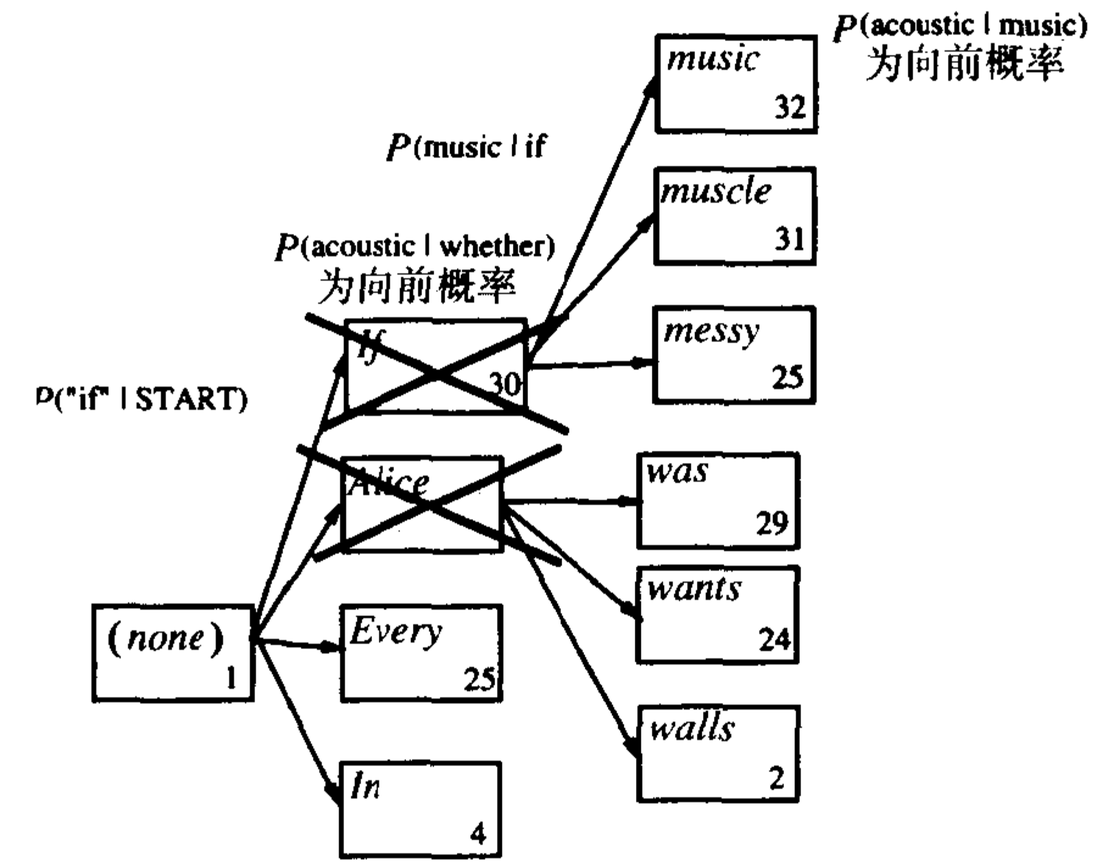


图 7.17　我们现在延伸结点 *if*，这时 *START if music* 这个假设的分数最高

我们不采用上面的办法，而是采用A*评估函数来计算。A*评估函数(Nilsson, 1980; pearl, 1984)称为 $f^*(p)$，对于给定的局部路径 p，有：

$$f^*(p) = g(p) + h^*(p)$$

$f^*(p)$是从部分路径 p 开始的最佳完全路径（完全句子）的估计分数。换言之，对于给定的部分路径 p，$f^*(p)$能够估计出，如果继续通过这个句子，这条路径的好坏程度。A* 算法使用两个部分来进行这样的估计：

- $g(p)$是从语段的起点到部分路径终点的分数。可以通过对于前面给定的声学符号串 p 的概率来很好地估计函数 g，也就是对于构成 p 的单词串 W 的 $P(A|W)P(W)$。
- $h^*(p)$是从部分路径延伸到语段终点的最佳分数的估计。

如何很好地估计 h^* 还是一个没有解决的问题，也是一个很有意思的问题。有一种方法是根据在句子中剩下的单词数来估计 h^* 的值(Paul, 1991)；进一步的讨论可参阅 Jelinek(1997)。

前面讲过，无论是 A* 解码算法还是其他两阶段解码算法，都要求使用快速匹配，以便很快地找出词表中哪些单词可以作为与声学输入中的某个部分相匹配的最佳候选。很多快速匹配算法都基于使用一种树结构词表（tree-structured lexicon），在词表中存储所有单词的发音，存储方式要使得在向前方推进计算概率时，能够与相同音子开头的单词共享，做到前后勾连。树结构词表是首先由 Klovstad and Mondshein(1975)提出的；Gupta et al.(1988)把树结构词表用于快速匹配中；Bahl et al.(1992)在 A* 解码算法中，Ney et al.(1992)和 Nguyen and Schwartz(1999)在 Viterbi 算法中，都使用过树结构词表。图7.18是在Sphinx-II的语音识别系统中使用的树结构词表的一个例子(Ravishankar, 1996)。每个树的根表示所有单词开头的第一个音子，单词开头的上下文与音子有关（音子上下文可以穿过单词的边界，也可以不穿过单词的边界），每个叶子与一个单词相关联。

还有很多其他种类的多阶段搜索算法，例如，不要把向前 – 向后搜索算法与HMM置参数的向前 – 向后算法相混淆(Austin et al., 1991)，前者的向前搜索比较简单，但紧接着的向后搜索（即时间反演）很细致。

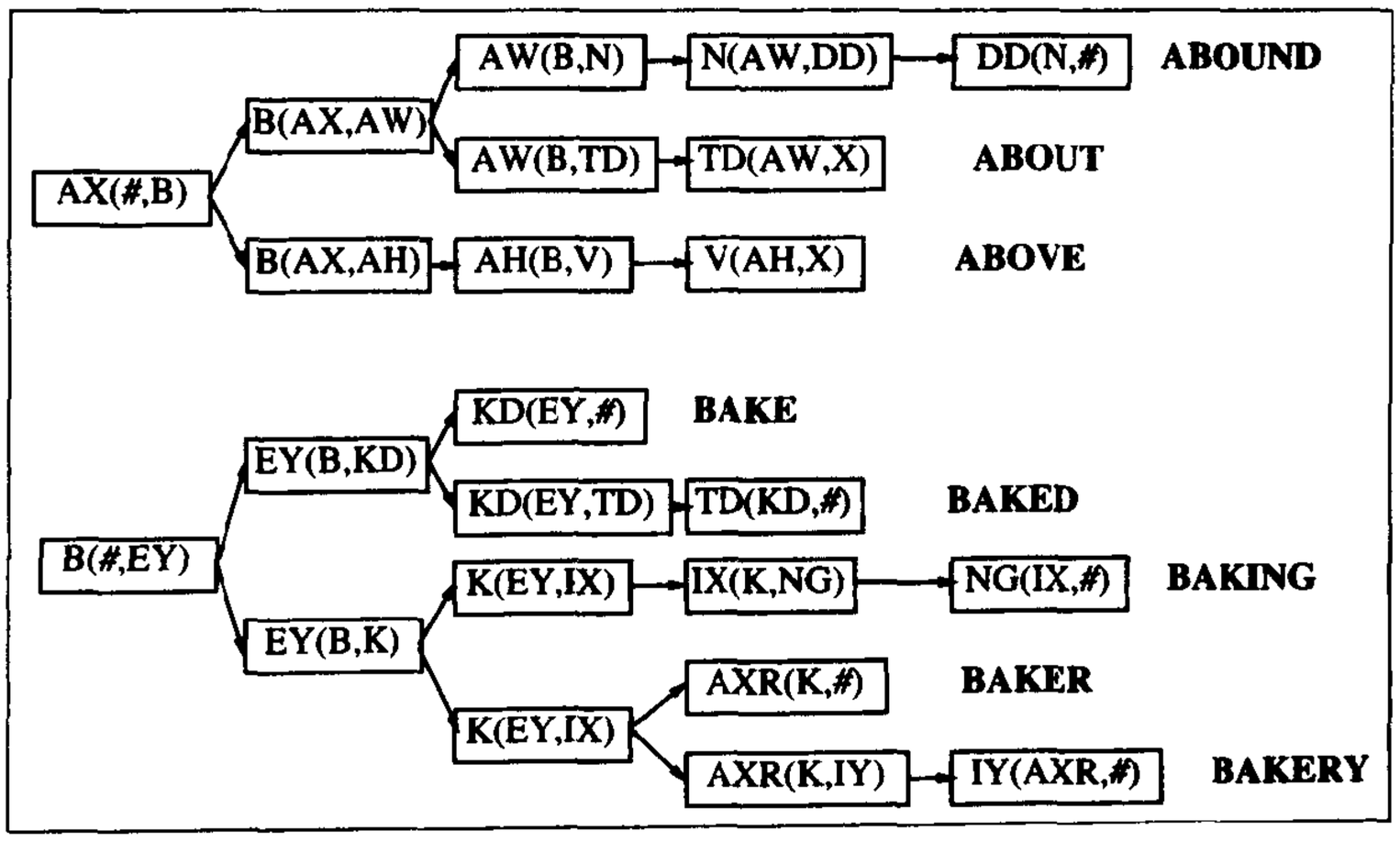


图 7.18 在Sphinx-II的语音识别系统中使用的一个树结构词表(Ravishankar, 1996)。每个结点对应于一个特定的三音子，三音子符号的记录方式基本上按照ARPAbet，只做了很小的修改，因此，EY(B, KD)表示前面为 B 后面以 K 结尾的音子 EY

7.5 语音的声学处理

在本节中，我们将对语音识别文献中的声学处理问题进行简短回顾，这种声学处理通常称为特征抽取（feature extraction）或信号分析（signal analysis）。特征（feature）这个术语是指表示语音信号的一个时间片（time-slice）的矢量。常用的特征类型有 LPC 特征和 PLP 特征等。所有这些特征都是声谱特征（spectral feature），使用形成波形（waveform）的不同频度的分布来表示这个波形；这种频度分布称为声谱（specturm）。我们首先简要地介绍声学波形和它的数字化方法，总结频度分析和声谱分析的一些研究思想，然后对于所抽出特征的类型进行大致的描述。这些都是非常简要的介绍；有兴趣的读者可以参阅其他声学语音学的有关语言方面的论著(Johnson, 1997; Ladefoged, 1996)，或参阅语音数字信号处理的有关工程方面的论著(Rabiner and Juang, 1993)。

7.5.1 声波

正如人耳的输入一样，语音识别系统的输入也是空气压力变化的一个复杂系列。这种空气压力变化显然来自说话者，说话者使用特定的方式使空气通过声门由口腔或鼻腔流出，就造成了空气压力变化。我们通过描画空气压力对于时间的变化情况的方法来表示声波。我们想像有一个垂直的薄片可以锁住空气压力的波形（大概就像说话者嘴巴前面的扩音器或听话者耳朵里的耳塞中的金属薄片），这样的比喻可以帮助我们理解这样的图形。这个图形可以测度这个薄片上的空气分子的压缩量或解压量。图7.19是Switchboard语料库中的一个波形图，它描述了一个人在打电话时说“she just had a baby”的波形中的某一段。

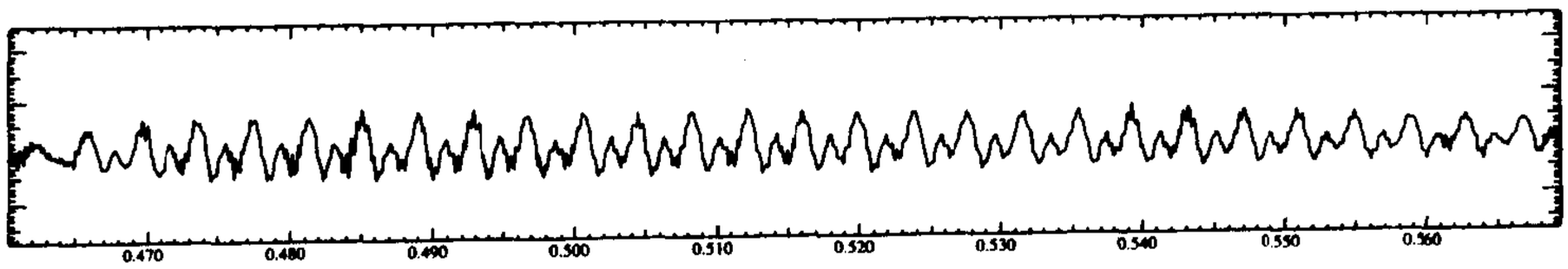


图 7.19 来自图7.20中所示的话段中元音[iy]的波形。*y*轴表示空气压力对于标准大气压的向上或向下的变化。*x*轴表示时间。注意，波形是有规律地反复变化的

波有两个重要的特征：频度（frequency）和振幅（amplitude）。频度是在 1 s 中之内波本身重复的次数，或者称为“周”（cycle）。注意，在图 7.19 中，我们捕捉到的波在 0.11 s 内重复了 28 次。所以，这个波的片断的频度是 28/0.11 或每秒 255 周。每秒的周数通常称为赫兹（Herz，缩写为 Hz），所以，图 7.19 中的波的频度就是 255 Hz。

图 7.19 中的纵轴表示空气压力变化的大小。纵轴上的值比较高（振幅较高），说明该时刻的空气压力比较高，零值表示空气压力正常（标准大气压），负值表示压力比标准大气压低（空气稀薄）。

还有两个感知特性与频度和振幅有关。音高（pitch）是与频度有关的感知特性。一般来说，如果语音具有较高的频度，我们会感觉到它的音高比较高，但是，频度和音高之间的关系不是线性的，因为人的听觉对于不同频度的敏锐程度是不同的。类似地，语音的响度（loudness）是音强（power）在感知上的对应，而音强与振幅的幅度有关。因此，振幅比较高的语音听起来会感觉响一些，当然，振幅与响度的关系也不是线性的。

7.5.2　怎样解释波形

只要给出了声波，人（和某些机器）就能够转写和理解语音，因此波形一定包含了使转写和理解语音成为可能的足够信息。在大多数场合下，这样的信息不是只观察波形就可以捕捉到的，但由于波形是可见的，而可见的观察已经足以使我们从中学习到很多东西。例如，元音和大多数辅音的不同在波形上是很明显的。我们说过，元音是浊音，发音比较长，比较响。语音的时间长度在波形图上直接表现为空间的长度。响度直接表现为振幅的高度。我们怎么来识别浊音呢？浊音是由声带有规律地开合所引起的。当声带振动时，在波形上就可以看到振幅出现有规则的波峰（peak），如图 7.19 所示。当发塞辅音时，例如，当[p], [t]或[k]处于成阻阶段时，在波形上就完全看不到波峰，事实上这时听到的是沉静。

在图 7.20 中，我们注意到，振幅出现有规则的波峰的地方就说明这里是浊音，从 0.46 s 到 0.58 s 是元音[iy]，从 0.65 s 到 0.74 s 是元音[ax}，等等。在没有振幅的地方说明是成阻时的沉静。例如，从 1.06 s 到 1.08 s 是第一个[b]的成阻，从 1.26 s 到 1.28 s 是第二[b]的成阻。

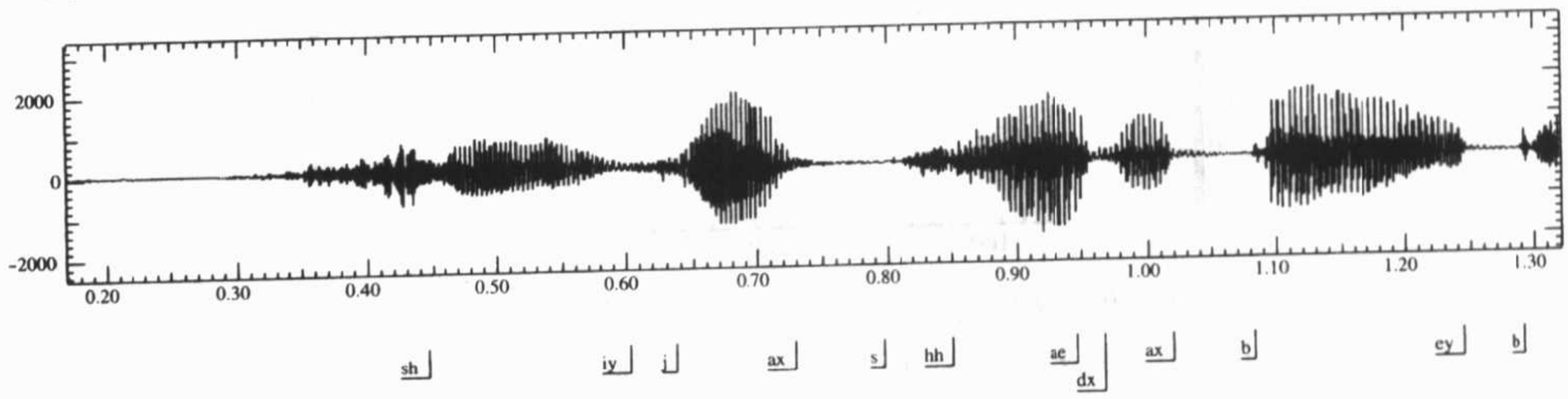


图 7.20　在 Switchboard 语料库中，句子“she just had a baby”的波形（4325 号会话）。发音者是一个女性，在 1991 年录音时，她大约有 20 岁，讲美国中南部方言。音子符号标在每个音子的末端。图中切掉了最后一个元音的最后一段

像[sh]这样的擦音也可以从波形上看出来。擦音产生的图形很不规则，从 9.33 s 到 0.46 s 的波形是典型的擦音波形。

7.5.3　声谱

某些比较宽泛的语音特征（表现为浊音，塞音的成阻和擦音）可以从波形上解释，但是更细致的分类（哪个元音？哪个擦音？）要求用声谱特征来对不同的输入进行不同的表示。声谱特征是根据 Fourier 对于复杂波的深入研究提出来的。Fourier 指出，每个复杂波都可以表示为很多频度不同

的简单波的总和。这可以与音乐中的和音（chord）相类比，一个和音可以由多个音符组合而成，任何波形也可以由相应于不同“音符”的波组合而成。

我们来研究图7.21，这个图显示了在句子中0.9 s时单词*had*中的元音[æ]的波形。注意，图中是一个重复了9次的复杂波形，而且在每个大的波中又包括4个小的重复波（注意，在每个重复波中有4个小的波峰）。整个波的频度是250 Hz（我们可以计算出来，因为在0.036 s内重复9次，所以频度为：9周/0.036 s = 250 Hz）。小波的频度大约等于大波频度的4倍，大约为1000 Hz。如果细心观察，还可以看到，在1000 Hz的波内，又包括两个小的波峰。这些很微小的波的频度应该是1000 Hz波的频度的两倍，因此它们的频度等于2000 Hz。

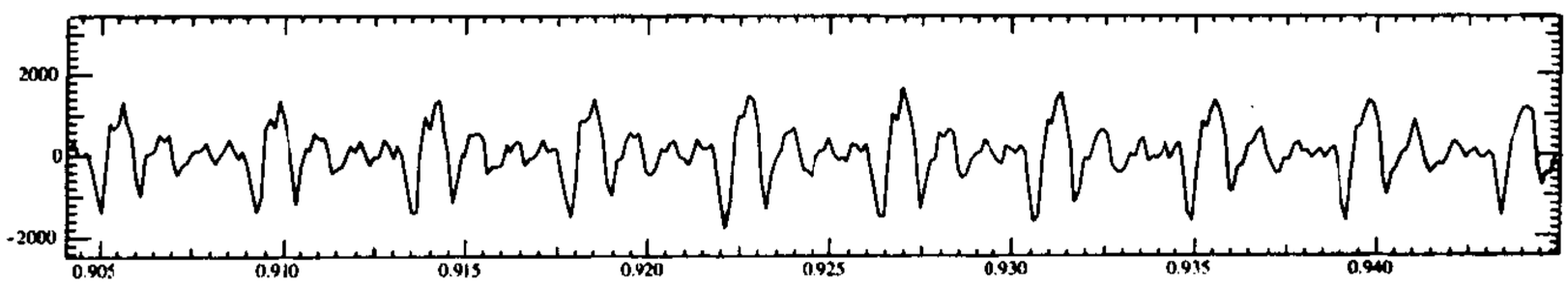


图7.21　从图7.20的波形中切出了单词*had*中的元音[æ]部分的波形

声谱是一个波的这些不同频度成分的表示。声谱可以使用傅里叶变换（Fourier transform）来计算。傅里叶变换是一种数学工具，可以把一个波的每个频度成分都分解开。但是，在很多语音技术的实际应用中，并不直接使用傅里叶变换声谱，而是使用称为LPC声谱的声谱平滑技术(Atal and Hanauer, 1971; Itakura, 1975)。

图7.22是图7.21中的波形的LPC声谱。LPC（Linear Predictive Coding）的意思是“线性预测编码”，这是声谱的一种编码方法，可以让我们比较容易地看到声谱峰（spectral peak）的位置。

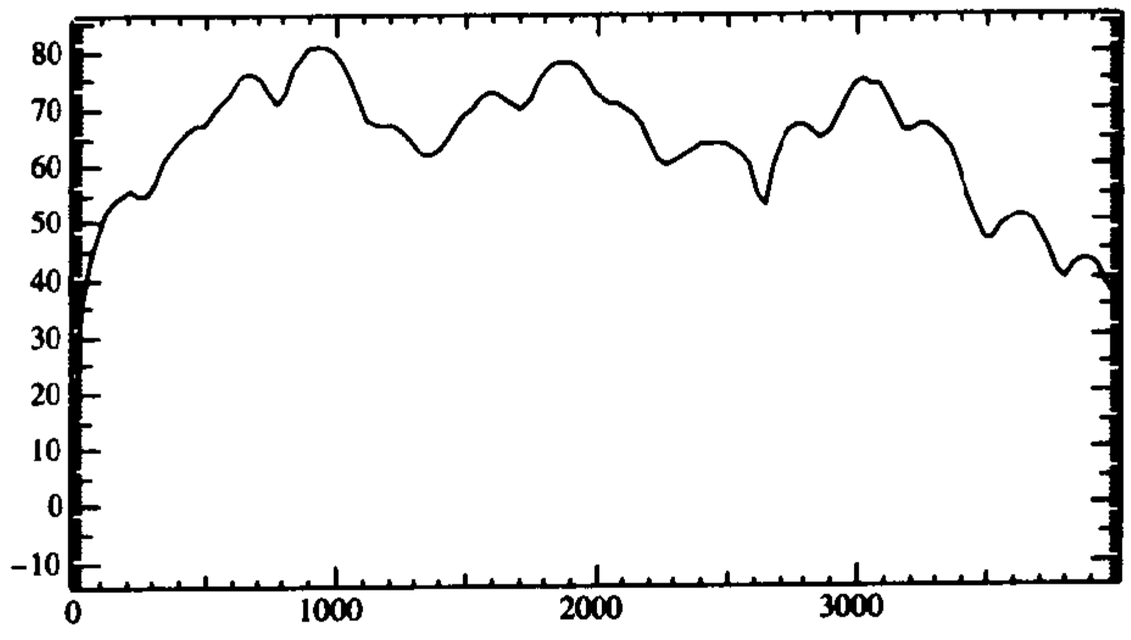


图7.22　*she just had a baby*的波形中[æ]元音在图7.21中所示的时间点上的LPC声谱。LPC使我们可以很容易地看到共振峰

声谱的x轴表示频度，y轴表示每个频度成分强度的大小，用振幅的对数即分贝（dB）来度量。图7.22说明了在930 Hz，1860 Hz和3020 Hz有三个重要的频度成分，其他频度成分的强度都比较低。这些重要的频度成分大致为1000 Hz和2000 Hz，这与我们观察图7.21中的声波所做出的预测是完全符合的。

为什么说声谱是有用的呢？在声谱中最容易看到的声谱峰是区别不同语音的最明显的特征，是音子的声谱特征的“签名”。例如，当化学元素加热时会显示出不同的光波长度，使得我们可以利用光波长度，根据光谱来探测远离我们若干个光年的星体上的元素。类似地，我们也可以通过观察波形的声谱，来探测不同音子的具有“签名”作用的区别性特征。无论对于人的语音识别还是对于机器的语音识别，利用声谱信息都是非常重要的。当人在听话的时候，耳蜗（cochlea）或内耳的功

能就是计算所接收的波形的声谱。与此类似，在语音识别中，作为HMM输入的各种特征都可以表示为声谱特征，通常表示为LPC声谱特征的变体，下面会讲到这个问题。

声谱表示在某一时刻的声波的频度成分，频谱（spectrogram）表示这些不同的频度怎样使波形随着时间的改变而改变，并把这种情况变成可见的东西。在频谱上，x 轴表示时间，y 轴表示频度的赫兹数。频谱的一个点上的暗度表示频度成分振幅的大小。例如，在图7.23的0.5 s处[①]，我们可以看到在1000 Hz附近有黑色的条纹。这意味着，单词 *she* 的[iy]在1000 Hz附近有一个重要的频度成分（1000 Hz大约处于音符B和C之间）。在频谱上的横向黑色条纹表示声谱峰，它们一般是元音，称为共振峰（formant）。

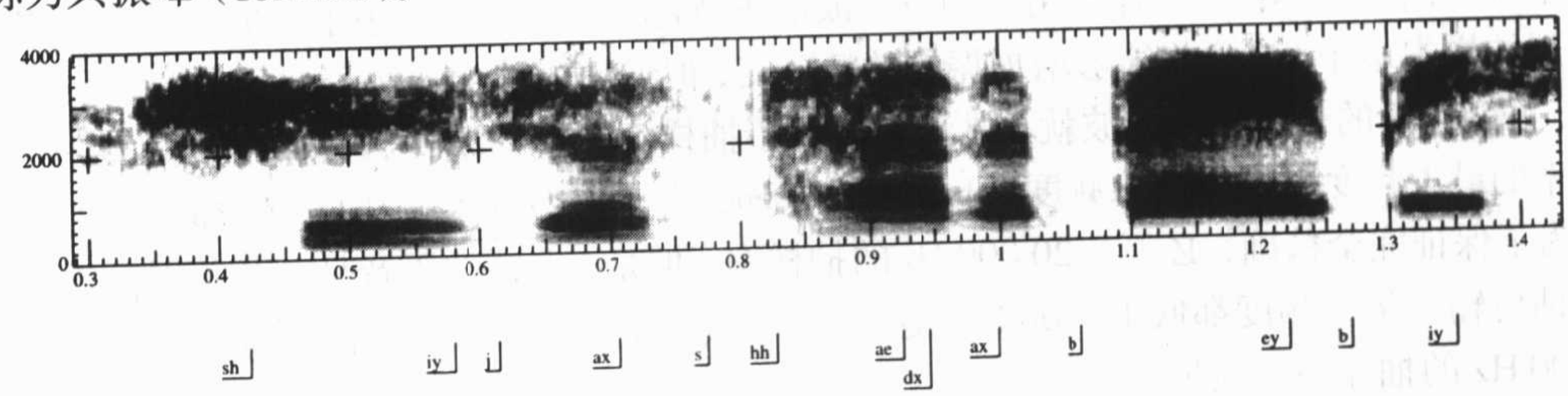


图 7.23　句子“She just had a baby”的频谱，这个句子的波形如图7.20所示。频谱可以想像成是图7.22中的声谱按时间片一段一段地结合而成的

声谱的这种表示对于音子的辨识有什么启示呢？首先，不同的元音在它们的特征位置具有不同的共振峰。我们已经说过，在波形样本中元音[æ]在930 Hz，1860 Hz和3020 Hz有共振峰。现在来看图7.20中语段开始时的元音[iy]。这个元音的声谱如图7.24所示。元音[iy]的第一个共振峰在540 Hz，它比元音[æ]的第一个共振峰的频度低得多，而第二个共振峰的频度（2581 Hz）又比[æ]的第二个共振峰的频度高得多。如果仔细观察就可以看出，这个共振峰的位置就在图7.23的0.5 s附近。

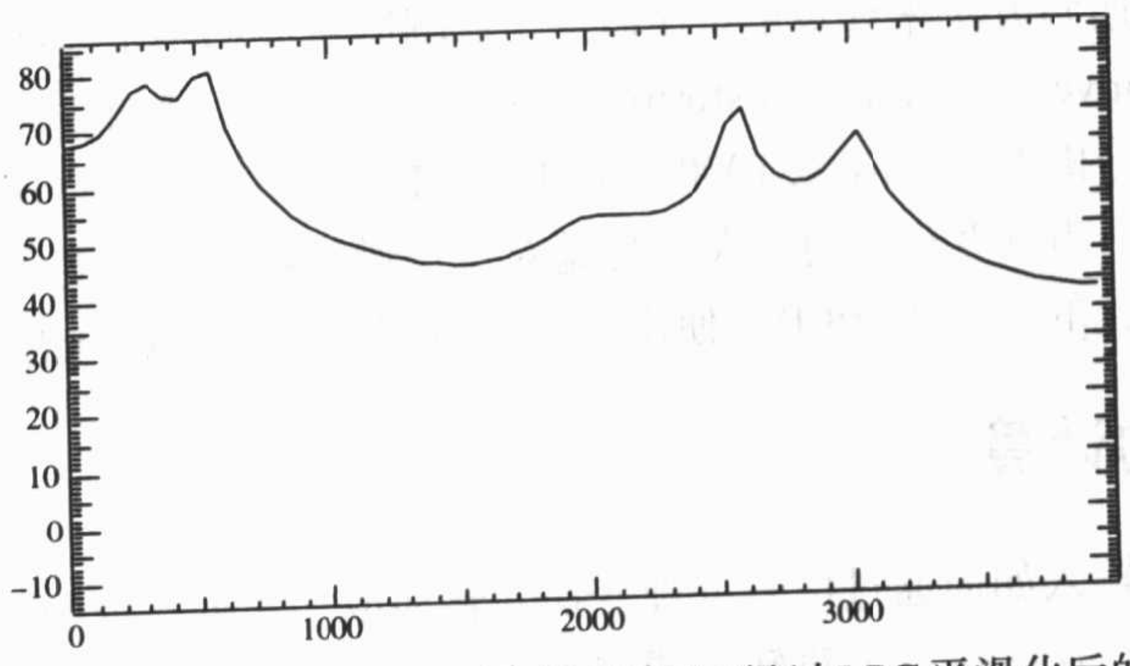


图 7.24　句子“*She just had a baby*”开头的元音[iy]经过LPC平滑化后的声谱。注意，它的第一个共振峰（540 Hz）比图7.22中元音[æ]的第一个共振峰的频度低得多，而它的第二个共振峰（2581 Hz）又比[æ]的第二个共振峰的频度高得多

前两个共振峰的位置（称为F1和F2）对于元音的辨别起着很大的作用，尽管不同说话人的共振峰不尽相同。共振峰还可以用于区分鼻音音子[n]，[m]以及[ŋ]，区分边音音子[l]和[r]。为什么不同的元音会有不同的声谱特征呢？共振峰是由口腔的共鸣引起的。我们可以把口腔想像成一个过滤器，它能够有选择地让声带振动产生的某些和谐音通过。舌头的运动改变口腔内空间的大小，这样就可以有选择地放大相应波长的声波，因此也就可以放大频带。

① 原文为0.9 s，可能有误。——译者注

7.5.4 特征抽取

对于波形特征和声谱的介绍虽然十分简短，但读者已经对于声谱特征的重要性以及它们与原来的波形的关系有了基本的了解，现在我们来总结性地介绍声谱特征的抽取过程。抽取过程从声波本身开始，以特征矢量（feature vector）结束[①]。

首先，输入的声波要进行数字化，也就是进行模拟-数字转换。模拟-数字转换的过程分为两步：抽样（sampling）和量化（quantization）。一个信号通过度量它在特定时刻的振幅来进行抽样；抽样率就是每秒提取的样本数目。通常的抽样率是8000 Hz和16 000 Hz。为了精确地测量声波，每周需要至少有两个样本：一个样本用于测量声波的正侧部分，一个样本用于测量声波的负侧部分。如果每周的样本多于两个，将能够增加振幅的精确度，但如果样本少于两个，就可能完全遗漏声波的频度。可能测量的最大频度的波就是那些频度等于抽样率一半的波（因为每周需要两个样本）。给定抽样率的最大频度称为Nyquist频度（Nyquist frequency）。大多数人类语音的频度都低于10 000 Hz；因此，为了保证完全精确，必须有20 000 Hz的抽样率。但是，电话的语音是由开关网络过滤过的，所以电话传输的语音频度都低于4000 Hz。这样，对于像Switchboard语料库这种电话带宽的语音来说，8000 Hz的抽样率已经足够了。

对于8000 Hz的抽样率，要求每秒测量8000个振幅，因此，重要的问题是把振幅的测量结果有效地进行存储。它们一般是以整数来存储的，或者是8比特（值为-128~127），或者是6比特（值为-32 768 ~ 32 767）。这个把实数值表示为整数的过程称为量化，因为存在着最小的颗粒度（量化范围），所有接近于这个量化范围的值都可以等同地表示。

当一个波形数字化之后，它就被转换成声谱特征的集合。LPC声谱用特征矢量来表示，每个共振峰用两个特征来表示，此外还要再加上两个表示声谱斜移的附加特征。所以，5个共振峰就可以用12个特征来表示（$12 = 5 \times 2 + 2$）。我们可以使用这样的LPC特征直接作为HMM的观察符号。但是，进一步的处理通常是针对特征进行的。有一种常用的特征称为倒谱系数（cepstral coefficient），倒谱系数是通过有效的递归方法来计算的。在概念上，倒谱系数相当于与线性预测系数对应的对数声谱的傅里叶逆变换（inverse Fourier transform）。另一种特征称为感知线性预测（Perceptual Linear Predictive，简称PLP）分析(Hermansky, 1990)。PLP取LPC特征并采用使其与人的听觉一致的方法来改进LPC特征。例如，当频度很高时，人的听觉对于声谱的辨别力就变得很差，而语音的感知响度与它的强度是成立方比的。所以，PLP就使用各种过滤器来过滤LPC声谱，并且取特征的立方根。

7.6 声学概率的计算

上一节说明了语音输入如何通过信号处理变换并转换成一系列特征矢量，每个矢量代表语音输入的一个时间片。怎样把这些特征矢量转变成概率呢？

特征矢量的概率的一种计算方法是，首先进行聚类（cluster），把它们聚类为能够计数的离散符号，然后计算它们在某个训练集中出现的次数。只要对次数进行计数，就可以计算出给定聚类的概率。这种方法一般称为矢量量化（vector quantization）。矢量量化在早期的语音识别算法中使用得非常普遍，但后来这种方法被一种更直接而且计算能力更强的方法所替代，即计算输入矢量实数值的（连续的）观察概率。这种方法要在一个连续的空间内计算概率强度函数（probability density function，简称pdf）。

---

① 读者必须记住，Picone(1993)曾经提醒我们，在使用单词“抽取”（extraction）时，不鼓励把它比喻成在“信号中”存在着某种特征等待着我们去抽取，这样的比喻是不恰当的。

有两种普遍使用的连续方法。这两种方法中使用得最广泛的是高斯概率强度函数（Gaussian pdf）。在最简单的高斯概率强度函数中，每个状态有一个单独的高斯函数，把观察矢量o_t映射为概率。另一种方法是使用神经网络或多层感知器，这种方法也能够通过训练给概率指派一个实数值的特征矢量。带有高斯观察概率估计器的HMM可以通过简单地扩充向前－向后算法（在附录D中讨论）进行训练而得到。带有神经网络观察概率估计器的HMM可以通过一种完全不同的、称为错误反向传播（error back-propagation）的算法进行训练而得到。

最简单地使用高斯概率强度函数时，我们假定观察特征矢量o_t的可能的值是按照正态分布的，这样就可以把观察概率函数$b_j(o_t)$表示为带有平均矢量μ_j和协方差矩阵Σ_j的高斯曲线；下标数表示矢量转置。为了保持完整性，这里写下了整个等式，但我们并不想深究这个等式的数学细节：

$$b_j(o_t) = \frac{1}{\sqrt{(2\pi)|\Sigma j|}} e^{[(o_t-\mu_j)'\Sigma_j^{-1}(o_t-\mu_j)]} \tag{7.10}$$

在通常情况下，我们简单地假定，协方差矩阵Σ_j是一个对角矩阵，也就是说，它包含倒谱特征1的简单方差，倒谱特征2的简单方差，等等，而不考虑倒谱特征1对倒谱特征2方差的影响。这意味着，在实际工作中，对于特征矢量的每个特征，我们只是简单地保持它的平均值和方差。

大多数语音识别系统的做法更复杂一些。对于每个状态，它们使用多重的高斯函数，这样，在计算观察矢量的每个特征的概率时，就要加上高斯曲线的一个变体。这种技术称为高斯混合法（Gaussian mixture）。另外，在一种称为参数约束（parameter tying）或约束混合法（tied mixture）的技术中，状态之间可以共享高斯函数(Huang and Jack, 1989)。例如，在声学上相似的音子的某些特征就可以共享高斯函数。

怎样计算在高斯估计中的平均值和方差呢？再次考虑只有一个状态i的非隐藏的马尔可夫模型会对我们有所帮助。特征平均矢量μ和方差矢量Σ可以通过平均的办法来估计：

$$\hat{\mu}_i = \frac{1}{T}\sum_{t=1}^{T} o_t \tag{7.11}$$

$$\hat{\Sigma}_i = \frac{1}{T}\sum_{t=1}^{T}[(o_t-\mu_j)'(o_t-\mu_j)] \tag{7.12}$$

但是，由于存在着多个隐藏的状态，我们不知道哪个观察矢量o_t是由哪个状态产生的。附录D将说明怎样修改向前－向后算法，把在时间t和状态i的HMM的概率分摊给每个可能的状态i，从而对每个可能的状态i指派一个观察矢量o_t。

模拟连续值特征的另一种方法是神经网络，又称多层感知器（MLP），或人工神经网络（Artificial Neural Network，简称ANN）。神经网络很复杂，用一两页篇幅不可能介绍清楚，但我们可以直观地说明，作为一种与高斯估计不同的方法，神经网络如何应用于概率估计。有兴趣的读者可以阅读有关神经网络基础知识的教材(Anderson, 1995; Hertz et al., 1991)以及一些与神经网络语音识别有关的专业文献(Bourlard and Morgan, 1994)。

神经网络是用加权链（weighted link）连接起来的一些小的计算单元的集合。给神经网络一个输入值矢量，它就可以计算出一个输出值矢量。在计算过程中，神经网络的每个计算单元计算输入单元的某个非线性函数，并把结果值传给其输出单元。

这里要介绍神经网络使用的一种方法，称为“混合隐马尔可夫－多层感知器方法”（简称混合HMM-MLP方法）。这种方法使用了HMM的一些成分（例如单词发音的状态图表示），但观察概率的计算使用了MLP，而不使用高斯混合法。这些MLP输入是在时间t的信号表示以及包围该信

号的一个窗口框架（又称为“帧”）；例如，这可以是在时间 t 的一个声谱特征矢量再加上在时间 $t+10$ ms, $t+20$ ms, $t+30$ ms, $t+40$ ms, $t-10$ ms 等的 8 个附加矢量。这样，神经网络的输入是 9 个矢量，对于一个时间片，每个矢量有一整套实数值声谱特征。网络对于每个音子有一个输出单元，作为对输出值的限制，所有输出单元的值的总和应该为 1，这样的网络可以用来计算对于给定的观察矢量 o_t，状态 j 的概率 $P(j|o_t)$。图 7.25 是这种网络的一个样本。

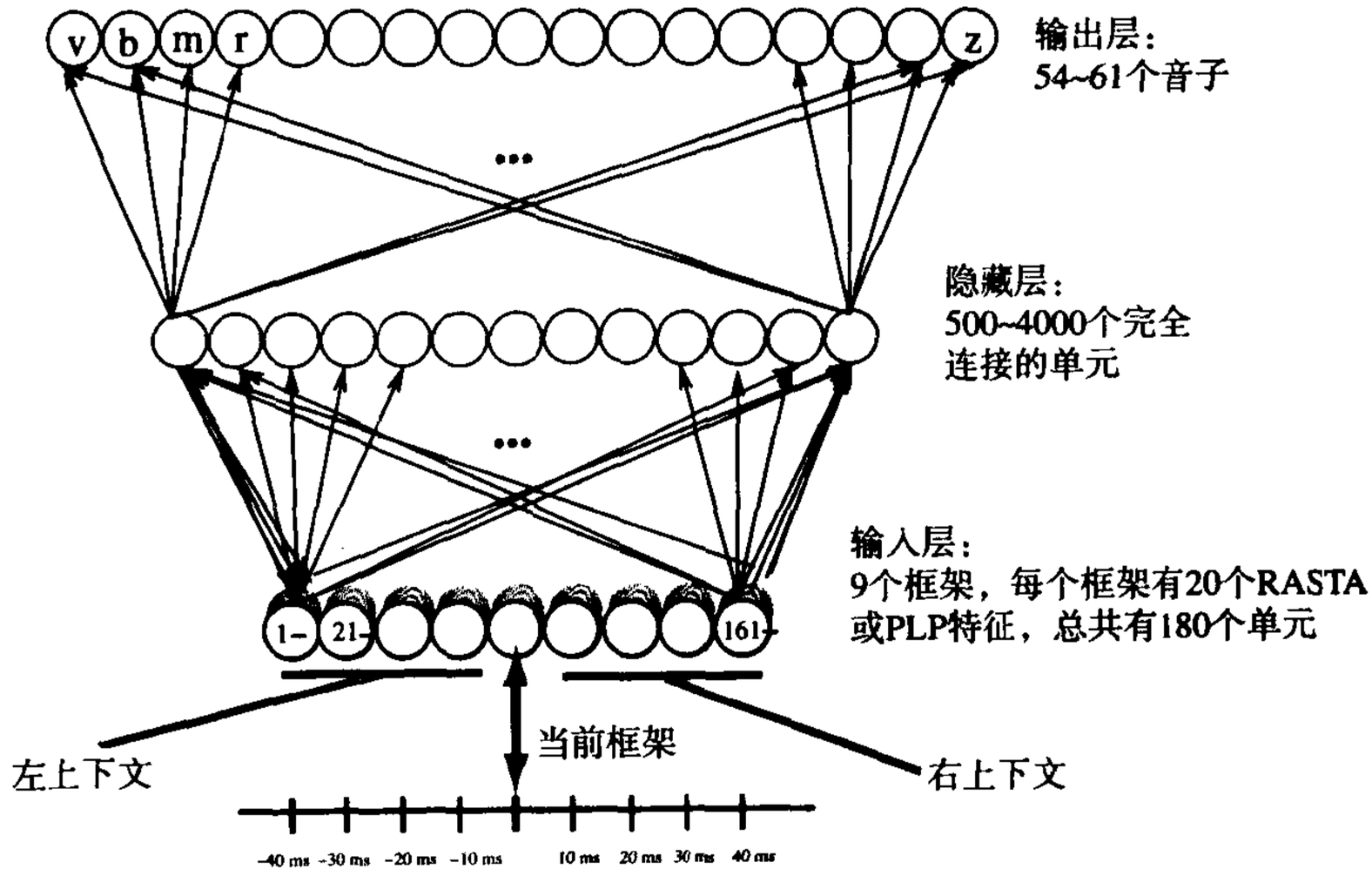


图 7.25 用于估计音子状态概率的神经网络。这个网络可以在不同于高斯模型的HMM模型中使用，这个特定的网络来自 Bourlard and Morgan(1994)描写的 MLP 系统。给定一个框架的特征矢量以及分别在它两侧的特征矢量（每侧有 4 个框架），可以估计出概率 $p(q_j|o_t)$。应用贝叶斯公式，可以把这个概率转化为观察似然度的估计 $b=p(o_t|q_j)$。这些网络是使用错误反向传播算法训练出来的，而这个算法是用于高斯模型的嵌入训练算法的一部分

对于给定的一个观察 o_t，MLP 计算 HMM 状态 j 的概率 $P(q_j|o_t)$。但是，对于 HMM 的观察概率函数 $b_j(o_t)$，我们需要的观察似然度是 $P(o_t|q_j)$。贝叶斯公式可以帮助我们认识到怎样从 $P(q_j|o_t)$ 来计算 $P(o_t|q_j)$。网络要计算的是：

$$p(q_j|o_t)=\frac{P(o_t|q_j)p(q_j)}{p(o_t)} \tag{7.13}$$

我们可以把各项重新安排如下：

$$\frac{p(o_t|q_j)}{p(o_t)}=\frac{P(q_j|o_t)}{p(q_j)} \tag{7.14}$$

式（7.14）右边的两项可以直接通过 MLP 来计算；分子就是 MLP 的输出，分母是给定状态的全部概率，也就是所有的观察加起来的概率，即 $\sigma_j(t)$ 的所有 t 的总和。这样，尽管我们不能直接计算 $P(o_t|q_j)$，但可以使用式（7.14）计算 $\frac{p(o_t|q_j)}{p(o_t)}$，根据计算结果就得出了“按比例的似然度”（scaled likelihood），它等于似然度被观察概率来除所得的值。实际上，这个按比例的似然度就像正规的似然度一样好，因为在识别过程中观察概率 $p(o_t)$ 是一个常数，它不会妨碍我们得到应得的结果。

训练 MLP 的误差反向传播算法要求我们知道对于每个观察 o_t 的正确音子标记 q_j。给定一个很大的观察训练集和正确的标记，算法反复地调整 MLP 中的权值，以减少训练集的错误。

在下一节中将会看到这个带标记的训练集来自哪里以及这个训练集与 HMM 使用的嵌入训练（embedded training）算法的适合程度。神经网络能够取得与高斯模型大体上相同的效果，它的优点是使用的参数较少，缺点是训练所需的时间较长。

方法论探讨：词错误率

语音识别系统的标准评测指标是词错误率（word error rate）。词错误率的计算是根据语音识别系统返回的单词串，通常称为假定单词串（hypothesized word string），与正确的转写或参照转写（reference transcription）的差别大小来进行的。给定一个正确的转写，计算词错误率的第一步是计算在假定单词串和正确单词串之间的单词的最小编辑距离。这个计算结果将是在正确单词串和假定单词串之间映射时的单词替代、单词插入和单词脱落的最小数目。这样，词错误率可以定义如下（注意，由于等式中包含插入，错误率可能会大于 100%）：

$$\text{词错误率} = 100 \times \frac{\text{单词插入数} + \text{单词替代数} + \text{单词脱落数}}{\text{正确转写的单词总数}}$$

下面是来自 CALLHOME 语料库的一个例子，参照语段和假定语段上下对齐，同时还给出了用于计算词错误率的计数：

参照语段：	i	***	**	UM	the	PHONE	IS	i	LEFT	THE	portable
假定语段：	i	GOT	IT	TO	the	*****	FULLEST	i	LOVE	TO	portable
评估：		I	I	S		D	S		S	S	

参照语段：	****	PHONE	UPSTAIRS	last	night	so	the	battery	ran out
假定语段：	FORM	OF	STORES	last	night	so	the	battery	ran out
评估：	I	S	S						

这个语段有 6 个替代，3 个插入，1 个脱落：

$$\text{词错误率} = 100 \times \frac{6+3+1}{18} = 56\%$$

在写作本书时，用美国国立标准和技术研究所（NIST）的广播新闻语料 Hub4 测试集来测试自然语音，最新的语音识别系统的词错误率达到 20%(Chen et al., 1999)，如果用 Hub5 测试集再加上 Switchboard，Switchboard-II 和 CALLHOME 语料库来测试自然语音，词错误率为 40%(Hain et al., 1999)。

7.7　语音识别系统的训练

现在我们已经介绍了为建立图 7.2 所示的标准语音识别系统所需的全部算法。我们已经知道了怎样建立一个 Viterbi 解码器，Viterbi 解码器怎样采用三个输入（观察概率、HMM 词表和 N 元语法的语言模型），并产生出概率最大的单词符号串。其中，观察概率是通过高斯估计或 MLP 估计从声谱特征得到的。但是，我们还不知道怎样来训练概率模型，以建立一个语音识别系统。

本节将扼要地介绍大多数 ASR 系统使用的嵌入训练（embedded training）过程，不考虑这些系统是基于高斯模型的，基于 MLP 的，还是基于矢量量化的。我们把算法的一些细节（例如，训练 HMM 概率的向前 - 向后算法）放在附录 D 中供读者查阅。

首先，为了训练一个基本的语音识别系统，我们需要的概率模型共有 4 个：

- **语言模型概率**　$P(w_i|w_{i-1}, w_{i-2})$
- **观察似然度**　$b_j(o_t)$

- **转移概率** a_{ij}
- **发音词表** HMM状态图结构

为了训练语音识别系统中的这些组成部分，一般来说需要有：

- 一个带有单词转写的语音声波文件的训练语料库
- 一个用于训练语言模型的大规模文本语料库，这个语料库中应该包含来自语音语料库的单词转写以及其他许多类似的文本
- 通常还需要一个小规模的、用于训练的语音语料库，这个语料库要有语音标记（也就是说，声学信号的框架要手工进行音位标注）

让我们从N元语法的语言模型开始。N元语法的语言模型使用第6章中描述的方法来训练；对大规模语料库中N元语法的出现单位进行计数，然后对这些计数进行平滑化和归一化。用于训练语言模型的语料库一般比用于训练HMM参数a和b的语料库的规模大得多。这是因为，训练语料库越大，模型的精确度越高。由于N元语法模型训练起来比训练HMM观察概率快，又由于文本需要的空间比语音需要的空间少，实践证明，使用5000万单词的大型语料库来训练语言模型是可行的。在一般情况下，用于训练HMM参数的语料库也是语言模型数据的一部分；重要的是保持声学模型和语言模型的一致性。

HMM 词表结构用手工来建立，工作时要使用一部非定制发音词典，如第 4 章中讲过的PRONLEX词典(LDC, 1995)或CMUdict词典。在有些系统中，词典中的每个音子都要映射到HMM模型的状态中。这样，单词*cat*对应于[k], [ae]和[t]三个状态。但是，很多系统使用我们前面描述的次音子结构，其中每个音子被分割为三个状态：音子的开始部分、中间部分和最后部分。每个这样的次音子还可以进一步在三音子上下文中出现不同的实例。

HMM参数的嵌入训练细节各不相同，这里介绍最简单的一种。首先，我们需要转移概率a_{ij}和观察概率$b_j(o_t)$的某些初始估计值。对于转移概率，起初假定对于任何状态的所有可能的下一个状态的值都是等概率的。观察概率可以从一个手工标注的小语料库开始，一步一步地使用自举（bootstrapped）的方法训练得到。例如，TIMIT语料库或Switchboard语料库各有4小时的经过语音标注的口语语料。它们可以为每个语音框架提供一个“正确”的音子状态标记q。这些都可以送到MLP中，或者求其平均给出初始高斯平均值和方差。对于MLP，初始的估计是很重要的，所以手工标注的自举结果（bootstrap）就成为了其后的规范。对于高斯模型，参数的初始值看来不那么重要，所以高斯模型的初始平均值和方差通常只是一个集合，在使用整个训练集的平均值和方差时，这个集合对于所有的状态都是等同的。

现在，对于所有的概率a和b，我们有了初始的估计。在算法的下一个阶段，高斯模型和MLP系统各不相同。对于MLP系统，我们使用一种称为“强制Viterbi对齐”（forced Viterbi alignment）的方法。强制Viterbi对齐取语段中正确单词的序列作为输入，将其与声谱特征矢量对齐。这样能够产生HMM状态的最佳序列，其中的每个状态与一个特征矢量对齐。因此，强制Viterbi对齐是正规Viterbi解码算法的一种简化，因为它只需要列出正确的音子序列，而不必找出单词序列。之所以称之为“强制”的，是因为我们给算法强加了某种约束，要求最佳路径必须通过一个特定的单词序列。它还要求使用Viterbi算法，因为单词有多个发音，并且每个音子的音延是不固定的。强制Viterbi对齐的结果是带有“正确”音子标记的特征矢量的集合，这个集合可以用来对神经网络进行再训练。强制对齐中的转移计数可以用来估计HMM的转移概率。

对于高斯 HMM 模型，我们不使用强制 Viterbi 对齐，而使用附录 D 中描述的向前 – 向后算法。对于给定的初始概率 a 和 b，我们对每个句子计算它的向前概率和向后概率。正如在 MLP 中的情况，向前 – 向后算法也需要使用我们对于正确单词的知识来加以约束。对于给定的模型 λ，用向前 – 向后算法来计算它的概率。在被转写的句子中，使用“已知”的单词序列的知识，从而能够知道，对于所有的符号串，什么样的单词模型可以使我们得到模型 λ，然后用这个模型来对每个句子的向前和向后概率进行计算。

7.8　用于语音合成的波形生成

前面我们已经讨论了声学处理，现在来讨论文本 – 语音转换系统（text-to-speech system，简称 TTS）中的声学问题。从第 4 章中已经知道，TTS 系统中语言处理部分的输出是一个音子序列，每个音子要带有它的音延和表示音高的 F0 曲拱。这些技术要求通常称为“目标”（target），目标是我们希望语音合成系统产生的结果。

最为普遍使用的算法类型是波形毗连（waveform concatenation）。这样的毗连合成（concatenative synthesis）的基础是语音数据库，它是一个单独的说话人的录音。然后，这个数据库被切分为一些短的单元，这样的单元包括音子、双音子、音节和单词以及其他单元。最简单的语音合成系统只有音子单元，在数据库的音子库中，每个音子只有一个单独的单元。为了选择适合的单元，我们可以生成一系列单元，把这些单元与输入中的音子序列相匹配。使用信号处理技术来平滑单元边缘的交接点，就可以简单地把这些单元的波形毗连起来，形成一个单独的合成语音波形。

实验证明，单独的音子毗连系统不能产生高质量的语音输出。正如在语音识别中那样，音子的上下文在音子的声学模式中起着重要的作用，因此在 /a/ 前面的 /t/ 与在 /s/ 前面的 /t/ 听起来有很大的差别。

我们前面在图 7.11 中描述的三音子模型被普遍选择作为语音识别的单元，因为三音子模型既覆盖了音子左边的上下文，又覆盖了音子右边的上下文。遗憾的是，一种语言的三音子数量一般非常大（有若干万之多），目前在语音合成中收集这么多的单元是不可能的。因此，在语音合成中通常使用双音子（diphone），因为双音子很好地平衡了依靠上下文和音子数量（一种语言有 1000~2000 个双音子）之间的矛盾。在语音识别中，双音子单元通常是从第一个音子后半部开始，在第二个音子的前半部结束。其原因在于音子的中间部分比边缘部分更稳定，所以在双音子模型中大多数 /a/ 音子的中间部分是非常相似的，尽管它的声学模型的开始点实际上与此不同。如果双音子在音子的中间部分毗连，那么相邻的单元之间的不连续情况经常可以忽略。

7.8.1　音高和音延的修正

刚才描述的双音子语音合成系统将产生与所要求的音子序列相应的质量不错的语音波形。但是，当双音子录音时，在毗连的波形中每个音子的音高和音延（也就是韵律）将是相同的，这将与输入所要求的音高和音延不对应。因此，语音合成处理的第二个阶段就要使用信号处理技术来改变毗连波形的韵律。

当来自声谱的音高被清楚地区分开以后，就可以用前面描述的线性预测模型（LPC）来修改韵律。如果毗连波形用线性预测系数的序列来表示，就可以生成相应于所希望音高的一套脉冲，并用这些脉冲再次激发线性预测系数，以再次产生语音波形。通过压缩和扩充系数的框架就可以改变时延。在线性预测（LPC）产生正确 F0 和时延的时候，系统会发出某种“蜂鸣”的语音信号。

另一种技术也可以达到同样的目标，这种技术称为“时域音高同步覆盖添加”（time-domain pitch-synchronous overlap and add technique，简称 TD-PSOLA）技术。TD-PSOLA 按照音高同步的原则工作，每个框架的中心处于语音中的音高标志的周围，而不是像一般的语音信号处理那样集中于一个正规的区间内。相互毗连的波形被分解为一些框架，每个框架以音高标志的周围为中心，并且向音高标志的两侧延伸。根据输入所要求的音高和音延来确定新的音高的集合，并把这些新的音高的框架结合起来，从而改变韵律。只要简单地覆盖和添加这样的框架，就可以产生出合成波形。当把新的音高标志彼此紧密地结合在一起时（音高的区间越短，音高的频度越大），音高就增加；而当把这些音高相互分开时，音高就减少。因此，只要把这些框架彼此覆盖起来，语音就会变长，把这些框架分开，语音就会变短。TD-PSOLA 的操作可以与可变速磁带录音机相比较，如果你用比一般录音情况下更快的速度倒带，音高的区间就会彼此接近，因此音高就会增加。但是，当给一个磁带录音机加速时，将会使语音的所有组成部分（包括表示元音特征的共振峰）的频度明显增加，因此会发出一种短促刺耳的不自然的声音。TD-PSOLA 与此不同，因为它首先把每个框架都分离开来，然后再逐渐减少框架之间的距离。由于每个框架的间隔不变，所以非音高部分的频度很难发生改变，最后形成的语音，除了音高不同之外，听起来基本上与原来的语音一样。

7.8.2 单元选择

信号处理和双音子毗连可以产生质量不错的语音，但是其结果还不算很理想。原因有很多，归根结底是由于这样的事实：每个双音子只有一个例子是不够的。首先，信号处理不可避免地会发生失真，当信号处理系统必须处理大量的音高和音延时，语音的质量就会变差。另外，在大多数语音信号处理算法中，还有很多细微的因素影响着语音的质量。例如，在说一个语段的时候，发音的强度会随着时间的延伸而减少，因此在语段的结尾产生的语音就比较弱。这样，如果双音子来自语段的开始部分，当这些语段处于短语后面的位置时，听起来就显得不自然。

单元选择合成的方法试图解决这个问题。这种方法对于每个单元收集若干个具有不同音高、不同音延和不同语言环境的实例，使得这个单元非常接近于目标，而信号处理所需的工作量也就少得多。单元选择的一种技术(Hunt and Black, 1996)的工作情况如下：

算法的输入与其他的毗连合成系统的输入相同，只是每个音子的曲拱F0要用三个F0的值来说明，而不是只用一个曲拱来说明。这种技术使用音子时要考虑它们的单元，在大规模的自然语音数据库中给音子做索引。在数据库中，每个音子要标记出它们的音延和三个音高的值。算法的工作分为两步。首先，对于目标单词中的每个音子，要从数据库中选择出它们的候选单元，使得它们的音延和 F0 尽可能地与该因子相匹配。这些候选单元使用一个目标成本（target cost）函数，按照它们与实际目标的接近程度来进行排序。算法工作的第二部分是测量每个单元的每个候选与相邻候选结合的好坏程度。各种结合位置都要进行评估，使得潜在可能的单元在中部进行结合，这种做法与我们在双音子模型中的做法相同。这些潜在的结合使用毗连成本（concatenation cost）函数来进行排序。最后一步工作是在整个句子中挑选那些目标成本和毗连成本最小的单元作为最佳单元。这一步使用非常简单的 Viterbi 算法来进行，比我们在 HMM 语音识别中使用 Viterbi 算法简单得多。这里，目标成本相当于观察概率，毗连成本相当于转移概率。

在通常的情况下，如果使用一个大规模的、包含每个单元的很多实例的语料库，单元选择合成系统能够产生的语音就会比纯粹的双音子合成系统产生的语音更自然。有些系统使用信号处理来保证韵律与目标彼此匹配，而其他一些系统只是单纯地把各个单元毗连起来，这些系统的想法是：只粗略地与目标匹配的语段，比那些尽管与目标匹配得很精确但同时带有某些信号处理失真的语段要好一些。

7.9　人的语音识别

人的语音识别有很多特征与我们介绍过的语音自动识别模型的很多特征是共同的。前面说过，像PLP分析这样的信号处理算法(Hermansky, 1990)的设计，实际上明显地受到人的听觉系统特性的启发。此外，人的词汇存取（lexical access，也就是人从心理词典中检索单词的过程）的四个特性在 ASR 中也是存在的，这四个特性是：频度（frequency）、平行性（parallelism）、邻近效应（neighborhood effect）和基于提示的处理（cue-based processing）。例如，与*N*元语言模型的ASR一样，人的词汇存取也是对单词频度敏感的。与低频度的口语单词相比，高频度的口语单词存取的速度比较快，或者存取时需要的信息比较少。在噪声环境下，或者在单词中只有局部部分呈现出来时，高频度的单词比低频度的单词更容易被成功识别(Howes, 1957; Grosjean, 1980; Tyler, 1984)。正如ASR一样，人的词汇存取是并行的：在同样的时刻可以激活多个单词(Marslen-Wilson and Welsh, 1978; Salasoo and Pisoni, 1985)。人对单词的存取还显示出邻接效应（一个单词的邻近词是一些与它类似的单词）。带有较高频度权值邻近词的单词，它的存取速度比那些邻近词比较少的单词慢一些(Luce et al., 1990)。Jurafsky(1996)指出，存取时的邻近效应可以用 ASR 中的贝叶斯模型来解释。

最后，人的语音感知还是基于提示的（cue-based）：语音输入要结合很多不同层次的提示来解释。例如，业已证明，人对于单个音子的感知要把许多不同的提示结合起来进行，包括声学提示，例如共振峰的结构或发音的确切时间(Oden and Massaro, 1978; Miller, 1994)，视觉提示，例如嘴唇的运动(Massaro and Cohen, 1983; Massaro, 1998)，词汇提示，例如音子所在单词的同一性(Warren, 1970; Samuel, 1981; Connine and Clifton, 1987; Connine, 1990)。还有一个通常称为音位复原效应（phoneme restoration effect）的例子。Warren(1970)取一个语音样本并且在咳嗽声的背景下替换其中的一个音子（例如，替换 *legislature* 中的[s]这个音子）。Warren 发现，受试者在听到这样的录制磁带时，在典型的情况下听到的仍然是包含[s]在内的整个单词 *legislature*，而且还感觉到了咳嗽声背景的存在。在人的语音感知中，其他的提示还有语义方面的单词联想（word association）和重复优先（repetition priming）。所谓“单词联想”指的是，如果同时还听到一个语义上相关的词，单词的存取会比较快。所谓“重复优先”指的是，当再听刚才已听到的单词时，其存取速度会比较快。这两种与语义有关的研究结果，已经被应用于第 6 章所讨论的一些新近的语言模型中。例如，Kuhn and Mori(1990)的存储模型就使用了重复优先的原理；Rossenfeld(1996)的触发器模型，Coccaro and Jurafsky(1998)的 LSA 模型以及 Bellegarda(1999)都使用了单词联想的原理。值得注意的是，这些卓越的思想绝不是现在才提出的，Cole and Rudnicky(1983)在他们一篇引人入胜的评述文章中指出，对于单词和音子处理的上下文效应这种深刻的关系，事实上早就被 William Bagley(1901)发现了。Bagley的成就，除了单词和音子的上下文效应之外，还包括对于音位复原效应的早期研究，这些成果在爱迪生（Edison）留声机滚筒的录音中得到了应用，后来又进行过修改，并且把它们公之于众。Bagley 的这些成果被遗忘了，很久以后才被再次发现[①]。

现代ASR模型和人的语音识别之间的一个差别是ASR模型的时间导向性（time-course）。在执行 ASR 算法时，解码搜索的过程是在整个语段上进行优化的，这一点很重要。这意味着，解码器在句子的结尾时返回的最佳句子假设可能与在句子中途返回的当前最佳句子假设有很大的差别。与此不同的是，有充分的证据表明，人的语音处理是在线的（on-line）。人们把一个语段一步步地切分成若干个单词，当他们听到相应单词的时候就指派给该单词一个解释，这个过程是递增进行的。例如，Marslen-Wilson(1973)曾经研究过所谓的“紧密背影”（close shadower）：当人们听到一个语

① 参见 1.6 节中讲过的科学中的多重独立发现。

音片断时，会在250 ms的短时间内留下该语音片断的背影（即向后重复）。Marslen-Wilson还发现，当这些背影出现错误时，它们会根据上下文，利用句法和语义特征来进行校正，在这250 ms之内进行单词的切分、剖析以及解释。Cole(1973)和Cole and Jakimik(1980)发现，在关于错误发音检查的研究中也存在着类似的效应。在这些研究成果的基础上，学者们研制了一些关于人类语音感知的心理模型，例如队列模型（Cohort model）(Marslen-Wilson and Welsh, 1978)和TRACE计算模型(McClelland and Elman, 1986)。TRACE计算模型重点研究了单词选择和切分的时间导向。例如，TRACE模型是一个连接主义的模型或者神经网络的交互式激活模型。在这个模型中，独立的计算单元被组织为三个平面：特征平面、音位平面和单词平面。每个单元表示关于它在输入中出现的一个假设。输入时，各个单元被并行地激活，单元之间的激活可以流动；不同平面的单元之间的连接是可激发的，而同一平面上的单元之间的连接是抑制的。所以，一个单词被激活之后，就可能稍微抑制所有其他单词的激活。

前面重点讨论了人的语音识别和机器的语音识别之间的相似性，但它们之间还有很多差别。特别是，许多事实表明，很多其他方面的提示在人的语音识别中也在发挥作用，而且这样的提示也被成功地结合到ASR系统中。这些被忽视的提示中最重要的是韵律。这里我们只举一个简单的例子，Cutler and Norris(1988)和Cutler and Carter(1987)都注意到，英语多音节单词有很多词的“例”的重音都在开始的音节，因此在“矩阵切分策略”（metrical segmentation strategy，简称MSS）中提出把重音用来作为单词切分的一个提示。

7.10 小结

本章在第4章到第6章的基础上，介绍了大词汇量连续语音识别（Large Vocabulary Continuous Speech Recognition）和文本－语音合成（Text-To-Speech synthesis）的基本算法。

- 语音识别系统的输入是一系列声波。波形、频谱和声谱等可视化工具用于理解信号中的信息。
- 在语音识别的第一步，声波被抽样和量化，并转化为某种声谱的表示；常见的声谱表示是LPC声谱特征，它对输入中的每个时间片提供一个特征矢量。
- 这些特征矢量被用来估计语音的似然度（又称为观察似然度），估计的方法有高斯估计混合法或神经网络法。
- 解码或搜索是用于发现与输入观察序列相匹配的模型状态的最佳序列的过程。在这个过程中使用解码或搜索这两个术语，这个事实表明，语音识别本质上就是一个交叉学科，它从一个以上的领域借用不同的比喻：解码来自信息论领域，搜索来自人工智能领域。
- 我们介绍了两种解码算法：时间同步Viterbi解码算法（这种算法通常要进行剪枝，又称为定向束搜索）和栈解码算法或A*解码算法。这两种算法都以特征矢量的系列作为输入并采用两个辅助算法，一个算法用于指派似然度（例如，高斯算法或MLP算法），另一个算法用于指派先验概率（例如，N元语言模型）。两者的输出都是单词串。
- 嵌入训练是训练语音识别系统的正规方法。给出一个带有手工建立的发音结构的初始词表，用这个词表来训练HMM的转移概率以及HMM的观察概率。这个HMM的观察概率估计可以通过高斯算法或MLP得到。
- 建立TTS系统的声学部分的一个途径是使用毗连合成，在毗连合成中通过毗连方法合成语段，然后从一个记录单独说话人语音的大规模数据库中取出双音子，并对双音子进行平滑处理。

7.11　文献和历史说明

世界上能够识别语音的第一台机器大概应该是一个名字称为“Radio Rex”的玩具，这种玩具在 20 世纪 20 年代开始在市场上出售。Rex 是一个赛璐珞的狗，这只狗（通过一个弹簧）会动，弹簧在500 Hz的声音能量的作用下放松，当弹簧一放松，狗就动起来。由于500 Hz粗略地相当于“Rex”中的元音的第一个共振峰的频度，所以当人们说“Rex”时，就好像是人在叫唤狗，狗就会在人的叫唤声的控制下走过来(David and Selfridge, 1962)。

在 20 世纪 40 年代末 50 年代初，建立了一系列机器语音识别系统。早期的贝尔实验室的系统可以识别一个单独说话人说出的 10 个数字中的任何一个数字(Davis et al., 1952)。这个系统存储了不依赖于说话人的 10 个模式，每个数字对应一个模式，每个模式代表数字中的头两个元音的共振峰。通过选择与输入存在最高相关系数的模式的方法，识别正确率达到了 97%~99%。Fry(1959)和 Denes(1959)在伦敦大学建立了一个音位识别系统，根据一个类似的模式识别原则，该系统能够识别 4 个元音和 9 个辅音。Fry 和 Denes 的系统首次使用音位转移概率来对语音识别系统进行约束。

在 20 世纪 60 年代末 70 年代初，产生了一些重要的创新性研究成果。首先，出现了一系列的特征抽取算法，包括高效的快速傅里叶变换（Fast Fourier Transform，简称 FFT）(Cooley and Tukey, 1965)，倒谱处理在语音中的应用(Oppenheim et al., 1968)以及用于语音编码的 LPC 的研制(Atal and Hanauer, 1971)。其次，提出了一些处理翘曲变形（warping）的方法，在与存储模式匹配时，通过展宽和收缩输入信号的方法，来处理说话速率和切分长度的差异。解决这些问题的最自然的方法是动态规划，正如我们在第 5 章中看到的，在研究这个问题时，同样的算法被多次重新发现。首先把动态规划应用于语音处理技术的是 Vintsyuk(1968)，尽管该成果没有被其他研究人员提及，但是 Velichko and Zagoruyko(1970)和 Sakoe and Chiba(1971)以及(1984)都再次重复了他的发现。稍后，Itakura(1975)把这种动态规划的思想和LPC 系数相结合，并首先在语音编码中使用，所建立的系统可以抽取输入单词中的 LPC 特征，并使用动态规划的方法把这些特征与存储的 LPC 模板相匹配。

在这个时期的第三项创新是HMM的兴起。1972年前后，分别在两个实验室中应用隐马尔可夫模型独立进行了语音问题研究。一方面的应用是由一些统计学家的工作开始的，Baum 和他的同事们在普林斯顿大学的国防分析研究所研究HMM，并把它应用于解决各种预测问题(Baum and Petrie, 1966; Baum and Eagon, 1967)。James Baker 在 CMU 做研究生期间，他研究了 Baum 他们的工作，并把这样的算法应用于语音处理(Baker, 1975)。与此同时，在 IBM Thomas J. Watson 研究中心，Frederick Jelinek，Robert Mercer 和 Lalit Bahl 独立地把 HMM 应用于语音研究(Jelinek et al., 1975)，他们在信息论模型方面的研究受到Shannon(1948)的影响。IBM的系统和Baker的系统非常相似，特别是都使用了本章中描述的贝叶斯方法。两个系统的早期工作之间的一个不同之处是解码算法。Baker的 DRAGON 系统使用了 Viterbi（动态规划）解码，而 IBM 系统则应用 Jelinek 的栈解码算法(Jelinek, 1969)。Baker在建立语音识别公司Dragon Systems之前，曾经短期参加过IBM小组的工作。IBM的语音识别方法在20世纪末期完全地支配了这个领域。IBM实验室确实是把统计模型应用于自然语言处理的推动力量，他们研制了基于类别的N元语法模型，研制了基于HMM的词类标注系统，研制了统计机器翻译系统，还使用熵和困惑度作为评测的度量。

HMM 逐渐在语音处理界流传开来。这种流传的原因之一是由于美国国防部高级研究计划署（Advanced Research Projects Agency of the U.S. Department of Defense，简称 ARPA）发起了一系列的研究和开发计划。第一个五年计划开始于 1971 年，Klatt(1977)对此做了评述。第一个五年计划的目标是建立基于少数说话人的语音理解系统，这个系统使用了一个约束性的语法和一个词表（1000

单词），要求语义错误率低于10%。ARPA资助了四个系统，而且对它们进行了比较。这四个系统是：系统开发公司（System Development Corporation，简称SDC）的系统，Bolt, Beranek & Newman（BBN）的HWIM系统，卡内基-梅隆大学的Hearsay-II系统和Harpy系统(Lowerre, 1968)。其中，Harpy系统使用了Baker的基于HMM的DRAGON系统的一个简化版本，在评测系统时得到了最佳的成绩。根据Klatt的评述，这是惟一达到了ARPA计划原定目标的系统（对于一般的任务，这个系统的语义正确率达到了94%）。

从20世纪80年代中期开始，ARPA资助了一些新的语音研究计划。第一个计划的任务是“资源管理”（Resource Management，简称RM）(Price et al., 1988)。这个计划的任务与ARPA早期的课题一样，主要是阅读语音（说话人阅读的句子的词汇量有1000个单词）的转写（也就是语音识别），但这个系统还包括一个不依赖于说话人的语音识别装置。其他的任务包括华尔街日报（WSJ）的句子阅读识别系统，这个系统开始时的词汇量限制在5000个单词之内，最后的系统已经没有词汇量的限制了（事实上，大多数系统已经可以使用大约60 000个单词的词汇量）。后来的语音识别系统识别的语音已经不再是照本宣科的语音，而是可以识别更加自然的语音了，其中有识别广播新闻的系统（称为Hub-4）(LDC, 1998; Graff, 1997)，能够转写广播新闻，包括转写那些非常复杂的广播新闻（例如，街头现场采访的新闻等），还有CALLHOME和CALLFRIEND等系统，其中的一部分称为Hub-5(LDC, 1999)，可以识别朋友之间在电话里的自然对话。空中交通信息系统（Air Traffic Information System, 简称ATIS）这个课题(Hemphill et al., 1990)是一个语音理解的课题，可以帮助用户预定机票，回答用户关于可能乘坐的航班、飞行时间、日期等方面的问题。

ARPA课题大约每年进行一次汇报，参加汇报的课题除了ARPA资助的课题之外，还有来自北美和欧洲的其他“志愿者”系统，在汇报时，彼此测试系统的单词错误率和语义错误率。在早期的测试中，那些营利性质的公司一般都不进行比赛，但后来很多公司却开始进行比赛（特别是IBM公司和ATT公司）。ARPA比赛的结果促进了各个实验室之间广泛地彼此借鉴和交流技术，因为在比赛中很容易看出在过去的一年中什么样的思想有助于减少错误，而这后来就成为了HMM模型传播到每个语音识别实验室的重要因素。ARPA的计划也造就了很多有用的数据库，这些数据库原来都是为了评估而设计的训练系统和测试系统（如TIMIT, RM, WSJ, ATIS, BN, CALLHOME和Switchboard），但后来都在各个总体性的研究中得到了使用。

在语音识别方面，有很多教材可供那些想进一步更深刻地理解本章所讲的内容的读者选择。Jelinek(1997)，Gold and Morgan(1999)以及Rabiner and Juang(1993)都是很全面的教材。后两本教材广泛地讨论了这个领域的发展历史。书中还有Levinson(1995)的综述性文章，这篇文章对于本章在历史方面的介绍是有影响的。我们对于向前-向后算法的描述来自Rabiner(1989)。其他有用的可以作为教材的文章是Knill and Young(1997)。关于语音识别领域的研究经常发表在两年举行一次的EUROSPEECH会议的论文集中，也发表在隔年举行的口语处理国际会议（International Conference on Spoken Language Processing，简称ICSLP）以及一年一次的IEEE声学、语音和信号处理国际会议（International Conference on Acoustics, Speech, and Signal Processing，简称ICASSP）的论文集中。关于这个领域的杂志有*Speech Communication*（语音通信），*Computer Speech and Language*（计算机语音和语言），*IEEE Transactions on Pattern Analysis and Machine Intelligence*（模式分析和机器智能IEEE学报）以及*IEEE Transactions on Acoustics, Speech, and Signal Processing*（声学、语音和信号处理IEEE学报）。

第二部分　句法的计算机处理

如果单词是语音和语言处理的基础，那么句法就是它的骨架。句法研究单词之间的形式关系。第二部分共有6章，我们将研究单词怎样类聚为称为词类（part-of-speech）的类别，怎样与相邻的单词组合成短语，以及一个句子的单词与单词之间彼此依赖的方式。本书的这个部分还要研究所有这些知识的计算机模型，包括上下文无关语法、词汇化语法、特征结构以及像Chomsky层级（Chomsky hierarchy）这样的元理论。我们还要介绍与这些理论有关的算法，如用于剖析（parsing）的Earley算法、CYK算法以及用于特征组合的合一算法。这个部分还包括这些句法知识的概率模型，如HMM词类标注系统和概率上下文无关语法。最后，我们还要研究人类句法处理的心理模型。

第8章　词的分类与词类标注

Conjunction Junction, what's your function?
Bob Dorough, *Schoolhouse Rock, 1973*

There are ten parts of speech, and they are all troublesome.
Mark Twain, *The Awful German Language*

The definitions [of the parts of speech] are very far from having attained the degree of exactitude found in Euclidean geometry.
Otto Jespersen, *The Philosophy of Grammar*, 1924[①]

Alexandria（公元前100年）的Dionysius Thrax或者另一位学者（我们可以理解，根据同一时代的作品的文本来判断准确的作者名是非常困难的，因此，我们不敢肯定学者的名字）曾经勾画出希腊语法（称为"technē"）的轮廓，对于他那个时代的语法知识进行了总结。这部著作是现代语言学词汇的直接来源，例如*syntax*（句法），*diphthong*（双元音），*clitic*（附着形式）和*analogy*（类推）等，都是从这部著作中来的，令人叹为观止。这部著作还描写了8个词类（part of speech）：名词、动词、代词、介词、副词、连接词、分词和冠词。尽管早期的其他学者（包括Aristotle和Stoics）也各自提出了他们的词类表，但Thrax的这8个词类成为了其后2000年来希腊语、拉丁语和大多数欧洲语言词类描写的基础。

《Rock学校》是一部3分钟的音乐动画系列片，首次于1973年在电视上放映。这部系列片的设计是为了激发儿童学习乘法表、语法、基础科学和历史的兴趣。读者大概已经开始明白为什么我们在上面要引这部系列片中的那段话。事实上，《Rock学校》中所讲的语法是按照非常传统的方式来描述的，它恰好也有8首歌来描述词类（虽然它的词类表对Thrax原来的词类进行了很小的修改，用形容词和叹词来替代原来的分词和冠词）。

词类又称为POS（Part Of Speech）、词的分类、形态类或词汇标记。在最近的词类表中，词的分类的数目比Thrax的词类数目多得多。例如，Penn树库有45类(Marcus et al., 1993)，Brown语料库有87类(Francis, 1979; Francis and Kučera, 1982)，C7标记集有146类(Garside et al., 1997)。

词类对于语言信息处理的意义在于能够提供关于单词及其邻近成分的大量有用信息。这不但对于主要的范畴是有用的（例如，动词和名词的区别），对于很多细致的区别也是很有用的。例如，这些标记集区别了主有代词（*my*, *your*, *his*, *her*, *its*）和人称代词（*I*, *you*, *he*, *me*）。如果知道了一个词是主有代词还是人称代词，就能知道什么词会出现在它的近邻（主有代词后面会出现名词，人称代词后面会出现动词）。这些信息在语音识别的语言模型中是非常有用的。

词类还能告诉我们关于单词发音的信息。在第4章中讨论过，单词*content*可以是名词，也可以是形容词，其发音是有区别的（作为名词发音为CONtent，作为形容词发音为conTENT）。因此，如果知道了单词的词类，在语音合成系统中就可以产生出更自然的发音。在语音识别系统中，就可

① 三句话的译文如下，其中第一句话翻译成中文后已经看不出原文的俏皮之处。
连接词的连接，什么是你的功能？
这里共有10个词类，它们都很令人棘手。
词类的定义还远远没有达到欧几里得几何学那样精确的程度。

以更精确地进行识别。类似的单词对还有 *OBject*（名词）和 *obJECT*（动词），*DIScount*（名词）和 *disCOUNT*（动词），参见 Cutler(1986)。

词类还可以用来在信息检索（Informational Retrieval，简称 IR）中分割词干（stemming），因为如果知道了单词的词类，就可以帮助我们了解单词会有什么样的形态词缀（第 3 章曾经讲过这个问题）。词类也可以帮助信息检索的应用，帮助我们从文献中选择出名词或其他的重要单词。自动词类标注也可以帮助我们建立自动词义排歧算法，还可以用于诸如基于类别的 *N* 元语法（class-based N-gram）这样的先进 ASR 语言模型中，在 8.7 节中将要讨论这个问题。词类标注经常用于文本的“局部剖析”（partial parsing）。例如，在信息抽取应用系统中，当需要快速查找名字或其他短语时，就需要词类的信息；我们将在第 15 章中讨论这个问题。最后，进行过词类标注的语料库对于语言研究是非常有用的。例如，词类标记可以帮助我们在大规模语料库中查找实例或特定结构的频度。

在本章的其余部分中，我们将首先对英语的词的分类进行总结，然后描述不同的标记集（tagset），以便对这些词的类别进行编码。然后，接着介绍 3 种标注算法：基于规则的标注（rule-based tagging）、随机标注（stochastic tagging）和基于转换的标注（transformation-based tagging）。

8.1　大多数英语词的分类

Well, every person you can know,
And every place that you can go,
And anything that you can show,
You know they're nouns. ①

Lynn Ahrens, Schoolhouse Rock, 1973

在此之前，我们使用诸如“名词”和“动词”之类的词类术语都是非常自由的。在传统上，词类是根据形态和句法功能来定义的；如果单词的词缀具有相似的功能（这是单词的形态特征），或者单词能够出现在相似的环境中（这是单词的“分布特征”），就把它们归为一类。同一类的单词会有语义一致性（semantic coherence）的倾向（名词事实上经常描写“人、地点或事物”，形容词经常描写“性质”），但语义的一致性并不总是必要的，所以一般来说我们不使用语义一致性作为定义词类的标准。

词类可以分为两个大类：封闭类（closed class）和开放类（open class）。封闭类是那些包含的单词成员相对固定的词类。例如，介词就是一个封闭类，因为英语中的介词是一个固定的集合，很少创造或杜撰新的介词。与此相反，名词和动词则是开放类，因为新的名词和动词不断地被创造出来，或者从其他语言中借用过来，新名词和新动词是层出不穷的，例如 *to fax*（发传真）是一个新的动词，*futon*（蒲团）则是一个借用的名词。任何给定的说话人或语料库的开放类单词可能是不同的，但是，所有说一种语言的人以及各种大规模的语料库，都可以共享相同的封闭类的集合。封闭类的单词又称为虚词（function word），虚词是语法词，如 *of*, *it*, *and*, *or*, *you* 等，它们一般都很短，并且出现频度很高，在语法中起着重要的作用。

世界语言中有四个大的开放类：名词、动词、形容词和副词。事实证明，这四个开放类在英语中都存在，尽管并不是在所有语言中都如此。很多语言没有形容词。例如，在美洲的土著语言 Lakhota 中就没有形容词；在汉语中也可能没有形容词，汉语中与英语中的形容词对应的许多单词的功能相当于动词的一个次类（subclass）。

① 这是音乐动画系列片中关于名词性质的顺口溜，其内容很通俗，故不翻译，原文照录，供读者玩味。——译者注

众所周知，人类语言至少应该包括两个范畴：名词和动词（尽管在某些语言中，如Nookta语，名词和动词的区别还难以捉摸）。名词就是名称，在名词所给出的词汇类别中，大多数出现的词都是表示人、地点或事物的。但是，由于像名词这样的词的类别是从功能上定义的（根据形态和句法功能来定义），而不是从语义上来定义的，因此某些表示人、地点或事物的单词不一定是名词；反之，某些名词不一定就是表示人、地点或事物的词。这样，名词可以包含具体的词，如 *ship*（船）和 *chair*（椅子），也可以包含抽象的词，如 *bandwidth*（带宽）和 *relationship*（关系），还可以包含某些类似动词的词，如在句子 *His pacing to and fro became quite annoying*（他踱来踱去的步调使人心烦）中的 *pacing*（步调）。怎样来定义英语中的名词呢？名词是具有如下能力的词：与限定词同时出现，如 *a goat*（一只山羊），*its bandwidth*（它的带宽）和 *Plato's Republic*（柏拉图的理想国），可以受主有代词修饰，如 *IBM's annual revenue*（IBM的年收入），并且大多数名词（不是所有的名词）都能够以复数形式出现，如 *goats*（一些山羊）和 *abaci*（算盘）。

在传统上，名词可分为专有名词（proper noun）和普通名词（common noun）。专有名词，如 *Regina*, *Colorado* 和 *IBM*，是某个特定的人或实体的名称。在英语中，它们前面一般不受冠词限制，例如 *the book is upstairs*（书在楼上）中的名词 *book* 前面有冠词，但是 *Regina is upstairs*（Regina在楼上）中的专有名词 *Regina* 前面没有冠词。在书面英语中，专有名词的第一个字母一般要大写。

在很多语言中，包括英语在内，普通名词又分为可数名词（count noun）和物质名词（mass noun）。可数名词是那些可以在语法上计数的名词；也就是说，它们能够以单数和复数的形式出现（*goat/goats*, *relationship/relationships*），并且能够计数（*one goat*, *two goats*）。当某种东西能够在概念上按照同质来分组时，就使用物质名词。诸如 *snow*（雪），*salt*（盐）和 *communism*（共产主义）这样的词是不可数的（不能说 **two snows* 或 **two communisms*）。物质名词出现时也可以没有冠词，而单数可数名词出现时不能没有冠词（可以说 *Snow is white*，但是不能说 **Goat is white*）。

动词包括大多数表示行为和过程的词。例如 *draw*（拉）、*provide*（提供）、*differ*（区别于）、go（去）等都是主要的动词。正如我们在第3章中看到的，英语动词有若干个形态：非第三人称单数（*eat*）、第三人称单数（*eats*）、进行时（*eating*）和过去分词（*eaten*）等。英语动词还有一个次类，称为助动词（auxiliary），我们在讨论封闭类形式时再研究助动词。

英语中的第三个开放类的形式是形容词；从语义上讲，这个封闭类包括很多描写性质和质量的单词。大多数语言都有表示颜色（*white*, *black*），年龄（*old*, *young*）和价值（*good*, *bad*）等概念的形容词，但是也存在着没有形容词的语言。我们在上面讨论过，有的语言学家主张，汉语族（Chinese family）的语言使用动词来描写英语形容词中的颜色和年龄等概念。

最后一个开放类形式是副词。无论从语义上讲还是从形式上讲，副词包含的词比较杂，就像一个矮胖子。例如，Schachter(1985)指出，在下面的句子中，所有斜体字的单词都是副词：

Unfortunately, John walked *home extremely slowly yesterday*
（遗憾的是，John 昨天回家时走得非常慢）

如果一定要说这一类词在语义上有什么一致性，那么我们只能说这些词都可以看成是要修饰某种东西的词，它们通常修饰的词是动词（verb），因此英语中把它们命名为adverb，但是这类词也可以修饰其他副词或者修饰整个动词短语。方位副词或地点副词，如 *home*, *here* 和 *downhill*（向下），说明某个行为的方向或地点；程度副词，如 *extremely*（非常），说明某个动作、过程或性质延伸的程度；方式副词，如 *slowly*，*slinkily*（鬼鬼祟祟地）和 *delicately*（微妙地），描述某个行为或过程的

方式；时间副词，如 *yesterday* 和 *Monday*，描述某个行为或事件发生的时间。由于这一类词具有异质（heterogeneous）的特性，在一些词类标注方案中，某些副词被标注为名词（例如，类似 Monday 这样的时间副词）。

与开放类不同，各种语言的封闭类差别很大。这里列举了英语中的某些重要的封闭类，并附有例子：

- **介词**　on, under, over, near, by, at, from, to, with
- **限定词**　a, an, the
- **代词**　she, who, I, others
- **连接词**　and, but, or, as, if, when
- **助动词**　can, may, should, are
- **小品词**　up, down, on, off, in, out, at, by
- **数词**　one, two, three, first, second, third

介词（preposition）出现在名词短语之前；从语义上讲，它们是表示关系的，通常表示空间或时间的关系，无论是字面上的关系（*on it*, *before then*, *by the house*），还是比喻的关系（*on time*, *with gusto*, *beside herself*）。但是，它们常常也要指出其他关系（"*Hamlet was written by Shakespeare, and [from Shakespeare]*"，"*And I did laugh sans intermission an hour by his dial*"（由于他的电话，我不间断地笑了一个小时）。图 8.1 是英语中的介词，材料来自 CELEX 联机词典(Celex, 1993)，根据它们在 1600 万词的英语语料库 COBUILD 中的频度排序。注意，不能认为这个表已经是定论。使用不同的词典和不同的标记集，词类标注的结果可能会有差别。这个表把介词和小品词结合在一起。参阅下面对小品词的论述。

of	540 085	through	14 964	worth	1563	pace	12
in	331 235	after	13 670	toward	1390	nigh	9
for	142 421	between	13 275	plus	750	re	4
to	125 691	under	9525	till	686	mid	3
with	124 965	per	6515	amongst	525	o'er	2
on	109 129	among	5090	via	351	but	0
at	100 169	within	5030	amid	222	ere	0
by	77 794	towards	4700	underneath	164	less	0
from	74 843	above	3056	versus	113	midst	0
about	38 428	near	2026	amidst	67	o'	0
than	20 210	off	1695	sans	20	thru	0
over	18 071	past	1575	circa	14	vice	0

图 8.1　CELEX 联机词典中的英语介词（以及小品词）。频度计数根据 1600 万词的 COBUILD 语料库

小品词（particle）与介词或副词相似，经常与动词结合形成一个更大的单位，称为短语动词（phrasal verb）。下面的例子来自 Thoreau：

So I *went on* for some days cutting and hewing timber...

Moral reform is the effort to *throw off* sleep...

我们可以看出，*went on* 中的 *on* 和 *throw off* 中的 *off* 都是小品词而不是介词。在第一个例子中，在 *on* 后面的不是一个名词短语，而是一个真正的介词短语。在第二个例子中，*throw off* 是一个及物短语动词，我们可以说 *off* 是一个小品词而不是一个介词，因为小品词也可以出现在其宾语的后面（可以说 *throw sleep off*，也可以说 *throw off sleep*），而介词不能这样用（*The horse went off its track* 不能说成 **The horse went its track off*）。

Quirk et al.(1985)给出了下面的单个的小品词表。由于很难自动区分介词和小品词，某些标记集就不区分它们（例如，CELEX 使用的一个标记集）。尽管在语料库中（如 Penn 树库）进行了这样的区分，自动处理时也很不容易做到可靠地区分，所以我们没有给出它们的计数。

aboard	aside	besides	forward(s)	opposite	through
about	astray	between	home	out	throughout
above	away	beyond	in	outside	together
across	back	by	inside	over	under
ahead	before	close	instead	overhead	underneath
alongside	behind	down	near	past	up
apart	below	east, etc.	off	round	within
around	beneath	eastward(s),etc.	on	since	without

图 8.2 英语中的单个小品词(Quirk et al., 1985)

英语中的一个特别小的封闭类是冠词（article）。英语有三个冠词：*a*, *an*, *the*，不过 *this* 和 *that* 也经常包括在冠词中，如 *this chapter* 中的 *this* 和 *that page* 中的 *that*。冠词通常位于一个名词短语的开头。*A* 和 *an* 表示名词短语是无定的，而 *the* 则表示名词短语是有定的。我们将在第 18 章中讨论有定性（definiteness）问题。冠词在英语中的出现频度很高；在大多数英语语料库中，*the* 是当之无愧的具有最大频度的单词。这里是COBUILD语料库的统计结果，语料库的规模是1600万单词：

the 1 071 676
a 413 887
an 59 359

连接词（conjunction）用来连接两个短语、分句或句子。并列连接词（coordinating conjunction），如 *and*, *or* 或 *but*，用于连接地位平等的两个成分。当一个成分处于某种嵌入的地位时，就要使用从属连接词（subordinating conjunction）。例如，在句子 *I thought that you might like some milk*（我想你大概喜欢喝牛奶）中的 *that* 是一个从属连接词，它把主要分句 *I thought* 和从属分句 *you might like some milk* 连接起来。这个分句之所以称为从属分句是因为整个分句是主要动词 *thought* 的“内容”。像 that 这样把动词和它的论元（argument）以这种方式联系起来的从属连接词又称为标补语（complementizer）。在第 9 章和第 11 章中，我们将更详细地讨论“补足”（complementation）的概念。图 8.3 中的表列出了英语中的连接词。

代词（pronoun）是简短地援引某些名词短语、实体或事件的一种形式。人称代词（personal pronoun）涉及人或实体（*you*, *she*, *I*, *it*, *me* 等）。主有代词（possessive pronoun）是人称代词的一种形式，说明实际的所属关系，而更常见的是只说明人和某种客体之间的抽象关系（*my*, *your*, *his*, *her*, *its*, *one's*, *our*, *their*）。Wh 代词（Wh-pronoun）可以在某种问题形式中使用（*what*, *who*, *whom*,

whoever ），也可以像标补语那样起作用（*Frieda, who I met five years ago...* ）。图 8.4 中的表列举了 CELEX 联机词典中的英语代词。

and	514 946	yet	5040	considering	174	forasmuch as	0
that	134 773	since	4843	lest	131	however	0
but	96 889	where	3952	albeit	104	immediately	0
or	76 563	nor	3078	providing	96	in as far as	0
as	54 608	once	2826	whereupon	85	in so far as	0
if	53 917	unless	2205	seeing	63	inasmuch as	0
when	37 975	why	1333	directly	26	insomuch as	0
because	23 626	now	1290	ere	12	insomuch that	0
so	12 933	neither	1120	notwithstanding	3	like	0
before	10 720	whenever	913	according as	0	neither nor	0
though	10 329	whereas	867	as if	0	now that	0
than	9511	except	864	as long as	0	only	0
while	8144	till	686	as though	0	provided that	0
after	7042	provided	594	both and	0	providing that	0
whether	5978	whilst	351	but that	0	seeing as	0
for	5935	suppose	281	but then	0	seeing as how	0
although	5424	cos	188	but then again	0	seeing that	0
until	5072	supposing	185	either or	0	without	0

图 8.3　CELEX 联机词典中的并列连接词和从属连接词。频度计数来自 1600 万个单词的 COBUILD 语料库

助动词（auxiliary）是英语动词中的一个封闭类。从语言学各学科交叉的角度来看，助动词是标志主要动词的某些语义特征的词（通常是动词）。这些语义特征包括：一个行为是现在发生的、过去发生的、还是将来发生的（时态）；一个行为是不是完成了（体）；一个行为是不是否定的（极性对立）；一个行为是必须的、可能的、建议性的，还是所期望的，等等（情态）。

英语中的助动词包括系动词 *be*，两个动词 *do* 和 *have* 以及它们的各种屈折变化形式，另外还包括情态动词（modal verb）。之所以把 *be* 称为系动词，是由于它把主语和某种类型的谓词性名词成分和形容词联系起来，例如 *He is a duck*（他是一个奇特的人）。助动词 *have* 用于标志完成时态，例如 *I have gone*（我去了）或 *I had gone*（我曾经去了）。*be* 用做被动形式的一部分，例如 *We were robbed*（我们被抢劫了），或者用在进行时的结构，例如 *We are leaving*（我们正在离开）。情态动词用于标志与事件或主要动词描述的行为相联系的情态。因此，*can* 表示能力或可能性，*may* 表示允许或可能性，*must* 表示必要性，等等。图 8.5 是英语情态动词的频度计数。除了上面提到的系动词 *have* 之外，还有情态动词 *have*，例如 *I have to go*（我只得走了）。这个情态动词在英语口语中使用得非常普遍。不过，无论是情态动词 *have*，还是情态动词 *dare*，在 CELEX 词典中的频度计数都很稀少，其原因在于 CELEX 词典不区分主要动词的意义［例如 *I have three oranges*（我有三只橘子）和 *He dared me to eat them*（他向我挑战要吃掉它们）］和情态动词的意义［例如 *There has to be some mistake*（这里必定有些错误）和 *Dare I confront him?*（我敢与他对抗吗？）］，以及非情态助动词的意义［例如 *I have never seen that*（我从来没有见过这种东西）］。

英语中还有很多单词不多不少地只有一种功能，它们是叹词（*oh, ah, hey, man, alas*）、否定词（no, not）、礼貌标志词（*please, thank you*）、问候词（*hello, goodbye*）以及表示存在的 there［*there are two on the table*（桌子上有两个）］。是把这些类别的词都指派一个特定的名字，还是合在一起（合并为叹词，或者甚至合并成副词），取决于标记的目的。

it	199 920	how	13 137	yourself	2437	no one	106
I	198 139	another	12 551	why	2220	wherein	58
he	158 366	where	11 857	little	2089	double	39
you	128 688	same	11 841	none	1992	thine	30
his	99 820	something	11 754	nobody	1684	summat	22
they	88 416	each	11 320	further	1666	suchlike	18
this	84 927	both	10 930	everybody	1474	fewest	15
that	82 603	last	10 816	ourselves	1428	thyself	14
she	73 966	every	9788	mine	1426	whomever	11
her	69 004	himself	9113	somebody	1322	whosoever	10
we	64 846	nothing	9026	former	1177	whomsoever	8
all	61 767	when	8336	past	984	wherefore	6
which	61 399	one	7423	plenty	940	whereat	5
their	51 922	much	7237	either	848	whatsoever	4
what	50 116	anything	6937	yours	826	whereon	2
my	46 791	next	6047	neither	618	whoso	2
him	45 024	themselves	5990	fewer	536	aught	1
me	43 071	most	5115	hers	482	howsoever	1
who	42 881	itself	5032	ours	458	thrice	1
them	42 099	myself	4819	whoever	391	wheresoever	1
no	33 458	everything	4662	least	386	you-all	1
some	32 863	several	4306	twice	382	additional	0
other	29 391	less	4278	theirs	303	anybody	0
your	28 923	herself	4016	wherever	289	each other	0
its	27 783	whose	4005	oneself	239	once	0
our	23 029	someone	3755	thou	229	one another	0
these	22 697	certain	3345	'un	227	overmuch	0
any	22 666	anyone	3318	ye	192	such and such	0
more	21 873	whom	3229	thy	191	whate'er	0
many	17 343	enough	3197	whereby	176	whenever	0
such	16 880	half	3065	thee	166	whereof	0
those	15 819	few	2933	yourselves	148	whereto	0
own	15 741	everyone	2812	latter	142	whereunto	0
us	15 724	whatever	2571	whichever	121	whichsoever	0

图 8.4 CELEX 联机词典中的代词。频度计数来自 1600 万个单词的 COBUILD 语料库

can	70 930	might	5580	shouldn't	858
will	69 206	couldn't	4265	mustn't	332
may	25 802	shall	4118	'll	175
would	18 448	wouldn't	3548	needn't	148
should	17 760	won't	3100	mightn't	68
must	16 520	'd	2299	oughtn't	44
need	9955	ought	1845	mayn't	3
can't	6375	will	862	dare	??
have	???				

图 8.5 CELEX 联机词典中的情态动词。频度计数来自 1600 万单词的 COBUILD 语料库

8.2　英语的标记集

前一节对英语单词所归入的词的类别进行了大致的描述。这里将描述在词类标注中实际使用的标记集（tagset），以便有血有肉地刻画出标记集的梗概，为下面各节描述各种标注算法做好准备。

通行的英语标记集有几种，多数都是从Brown语料库所用的包含87个标记的标记集发展而来的(Francis, 1979; Francis and Kučera, 1982)。英语中最常用的标记集有 3 个：Penn Treebank 的标记集包含 45 个标记，是小标记集(Marcus et al., 1993)；Lancaster 大学 UCREL 计划的成分似然性自动词性标注系统 CLAWS（the Constituent Likelihood Automatic Word-tagging System）使用的标记集 C5 包含 61 个标记，是中型的标记集，用于标注英国国家语料库（the British National Corpus，简称 BNC）(Garside et al., 1997)；第三个标记集是包含 146 个标记的大型标记集 C7(Leech et al., 1994)。C5 和 C7 标记集收录在本书附录 C 中，也可以参阅 Sampson(1987)和 Garside et al.(1997)，这两份文献对于这些标记集和其他标记集的来源和置标情况进行了详细描述。本节将介绍这些标记集中最小的一个，即 Penn Treebank 的标记集，然后讨论其他标记集的一些特殊的附加标记。引入这些标记对于某些系统是有用的。

图8.6中的Peen Treebank标记集应用于Brown语料库和一些其他的语料库。这个例子是Brown语料库的 Penn Treebank 版本中的一个标注了的句子（一个展开的 ASCII 文件，标记通常标在每个单词之后，中间用斜线隔开，但也可以用其他方式来表示）：

The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

标记	说明	例子	标记	说明	例子
CC	Coordin. Conjunction	*and, but, or*	SYM	Symbol	+,%, &
CD	Cardinal number	*one, two, three*	TO	"to"	*to*
DT	Determiner	*a, the*	UH	Interjection	*ah, oops*
EX	Existential 'there'	*there*	VB	Verb, base form	*eat*
FW	Foreign word	*mea culpa*	VBD	Verb, past tense	*ate*
IN	Preposition/sub-conj	*of, in, by*	VBG	Verb, gerund	*eating*
JJ	Adjective	*yellow*	VBN	Verb, past participle	*eaten*
JJR	Adj., comparative	*bigger*	VBP	Verb, non-3sg pres	*eat*
JJS	Adj., superlative	*wildest*	VBZ	Verb, 3sg pres	*eats*
LS	List item marker	*1, 2, One*	WDT	Wh-determiner	*which, that*
MD	Modal	*can, should*	WP	Wh-pronoun	*what, who*
NN	Noun, sing. or mass	*llama*	WP$	Possessive wh-	*whose*
NNS	Noun, plural	*llamas*	WRB	Wh-adverb	*how, where*
NNP	Proper noun, singular	*IBM*	$	Dollar sign	*$*
NNPS	Proper noun, plural	*Carolinas*	#	Pound sign	*#*
PDT	Predeterminer	*all, both*	"	Left quote	(' or ")
POS	Possessive ending	*'s*	"	Right quote	(' or ")
PP	Personal pronoun	*I, you, he*	(	Left parenthesis	([, (, {, <)
PP$	Possessive pronoun	*your, one's*	)	Right parenthesis	( ],), }, >)
RB	Adverb	*quickly, never*	,	Comma	,
RBR	Adverb, comparative	*faster*	.	Sentence-final punc	(. ! ?)
RBS	Adverb, superlative	*fastest*	:	Mid-sentence punc	(: ; ... – -)
RP	Particle	*up, off*			

图 8.6　Penn Treebank 的词类标记（包括标点符号）

Penn Treebank的标记集是从Brown语料库原有的87个标记的标记集中挑选出来的。这个小标记集去掉的标记主要是那些表示单词条目本身可以包含的信息的标记。例如，在原来的Brown语料库的标记集以及像C5这样的比较大的其他标记集中，对于动词*do*，*be*和*have*的不同形式都有不同的标记（例如，C5中用VDD表示*did*，用VDG表示*doing*），而这样的标记，在Penn Treebank的标记集中都略去了。

在Penn Treebank的标记集中，有些句法的区别没有表示出来，因为树库中的句子都是剖析过的，而不仅仅只是做了标记，所以某些句法信息已经在短语结构中表示出来了。例如，介词和从属连接词结合为一个单独的标记IN，这是因为在句子的树结构中，它们之间的歧义已经消解了（从属连接词总是位于分句之前，而介词总是位于名词短语之前或处于介词短语中）。

但是，在大多数进行标注的场合，并不要求对语料库进行剖析。正是由于这个原因，Penn Treebank的标记集在很多应用中显得并不够用。例如，C7标记集中就区分介词（II）和从属连接词（CS），并且还区分介词（II）和动词不定式的标志（TO）。

对于特定的应用目的来说，使用什么样的标记集取决于应用中需要多少信息，参见附录C中的标记集C5和C7。

8.3 词类标注

词类标注（Part-of-Speech tagging或POS tagging）可简称为标注，这是给语料库中的每个单词指派一个词类或者词汇类别标记的过程。这些标记通常也用来标注标点符号，因此自然语言的标注过程与计算机语言的词例还原（tokenization）过程是一样的，尽管自然语言的标记具有更多的歧义性。正如我们在本章开始时指出的，词性标注在语音识别、自然语言剖析和信息检索中都起着越来越重要的作用。

标注算法的输入是单词的符号串和前一节描述过的标记集。算法的输出要让每个单词都标上一个单独的而且是最佳的标记。例如，这里给出了ATIS语料库中的一些样本句子。ATIS语料库是一个关于航空旅行订票对话的语料库，我们将在第9章中讨论它。对于每个单词，我们给出了一个潜在的标记输出，标记集采用在图8.6中定义的Penn Treebank标记集：

```
VB      DT    NN    .
Book    that  flight .

VBZ     DT    NN    VB    NN    ?
Does that flight serve dinner ?
```

尽管这是一些非常简单的例子，但是要给每个单词都自动指派一个标记也并不非易事。例如，*book*这个单词就是有歧义的（ambiguous）。也就是说，*book*有一个以上的用法和一个以上的词类。*book*可以是动词，例如*book that flight*（订那种飞机票）或*book the suspect*（控告嫌疑人），也可以是名词，例如*hand me that book*（把那本书交给我）或*a book of matches*（一本关于比赛的书）。类似地，that可以是限定词，例如*Does that flight serve dinner*（这个航班供应晚餐吗），也可以是标补语，例如*I thought that your flight was earlier*（我认为，你的飞机早一些）。POS标注的问题就是消解这样的歧义，在一定的上下文中选择恰如其分的标记。我们在本书中将会看到，词类标注是歧义消解（disambiguation）中的一个重要方面。

词类标注的难度究竟有多大呢？英语中的大多数单词都是没有歧义的，也就是说，这些单词只有一个单独的标记。但是，英语中的最常用的单词很多都是有歧义的，例如*can*可以是助动词，表

示“能够”（to be able），也可以是名词，表示“罐头”（a metal container），还可以是动词，表示“把某个东西装进罐头”（to put something in such a metal container）。事实上，DeRose(1988)报告说，在 Brown 语料库中只有 11.5% 的英语词型（word type）是有歧义的，40% 以上的词例（word token）是有歧义的。根据 Francis and Kučera(1982)，DeRose 给出了如图 8.7 所示的标记歧义表。

无歧义（只有1个标记）	**35 340**	
歧义（有2~7个标记）	**4100**	
2个标记	3760	
3个标记	264	
4个标记	61	
5个标记	12	
6个标记	2	
7个标记	1	（“still”）

图 8.7　在 Brown 语料库中按歧义程度排列的词型数目(DeRose, 1988)

幸运的是，在占 40% 的歧义词例（word token）中，有不少是很容易消解歧义的。这是因为与一个单词相关联的不同标记的使用情况并不完全等同。例如，*a* 可以是一个限定词，也可以是字母 *a*（作为首字母缩写词的一部分或处于开头），但 *a* 作为限定词更常见。

大多数的标注算法可以归纳为两类：一类是基于规则的标注算法（rule-based tagger），一类是随机标注算法（stochastic tagger）。基于规则的标注算法一般都包括一个手工制作的歧义消解规则的数据库，这些规则要说明歧义消解的条件。例如，当一个歧义单词的前面是限定词时，就可以判断它是名词，而不是动词。下一节将根据 Karlsson 的约束语法构架描写一个基于规则的标注系统的样本 ENGTWOL[Karlsson et al.(1995)]。

在解决标注歧义问题时，随机标注算法一般会使用一个训练语料库来计算在给定的上下文中某一给定单词具有某一给定标记的概率。8.5 节将描写一个随机标注系统，该系统称为 HMM 标注系统，又称为最大似然度标注系统或马尔可夫模型标注系统。这个系统是建立在第 7 章中介绍过的隐马尔可夫模型（HMM）的基础上的。

最后，8.6 节还要描写一种标注算法，该算法称为基于转换的标注算法（transformation-based tagger），或者根据 Brill(1995)称为 Brill 标注算法（Brill tagger）。Brill 标注算法享有上述两种标注算法的特点。与基于规则的标注算法相似，这种算法要根据规则来决定一个有歧义的单词应该具有什么样的标记。与随机标注算法相似，这种算法有一个部分是用于机器学习的，规则可以由前面已标注好的训练语料库自动推导出来。

8.4　基于规则的词类标注

早期的词类自动标注算法的基础是使用两阶段的体系结构技术(Harris, 1962; Klein and Simmons, 1963; Greene and Rubin, 1971)。第一个阶段使用一部词典给每个单词指派一个潜在词类表。第二个阶段使用一个很大的手工书写的歧义消解规则表来筛选原来的潜在词类表，使每个单词得到一个单独的词类标记。

ENGTWOL 标注系统(Voutilainen, 1995)也是建立在与上面同样的两阶段体系结构技术的基础上的，不过它的词典和歧义消解规则都比早期的系统复杂得多。ENGTWOL 的词典是根据第 3 章中介绍的双层形态学来建立的，约有 56 000 个英语单词词干的词条(Heikkilä, 1995)。如果一个单词有多个词类（例如，*hit* 的名词意义和动词意义）就分别立条，而且计算时不计屈折形式和派生形式。

每个条目要标出它的形态特征和句法特征。图8.8列出了词典中选出的一些单词以及它们的经过简化的特征表。

单词	POS	其他 POS 特性
smaller	ADJ	COMPARATIVE
entire	ADJ	ABSOLUTE ATTRIBUTIVE
fast	ADV	SUPERLATIVE
that	DET	CENTRAL DEMONSTRATIVE SG
all	DET	PREDETERMINER SG/PL QUANTIFIER
dog's	N	GENITIVE SG
furniture	N	NOMINATIVE SG NOINDEFDETERMINER
one-third	NUM	SG
she	PRON	PERSONAL FEMININE NOMINATIVE SG3
show	V	IMPERATIVE VFIN
show	V	PRESENT -SG3 VFIN
show	N	NOMINATIVE SG
shown	PCP2	SVOO SVO SV
occurred	PCP2	SV
occurred	V	PAST VFIN SV

图 8.8 Voutilainen(1995)和 Heikkilä(1995)描述的 ENGTWOL 词典中的词条样本

图 8.8 中的大多数特征都可以根据符号的字面来解释；SG 表示单数（singular），-SG3 表示非第三人称单数（other than third-person-singular），ABSOLUTE 表示非比较级和非最高级的形容词，NOMINATIVE 只表示非所有格，PCP2 表示过去分词（past participle）。PRE，CENTRAL 和 POST 表示限定词的顺序位置（前限定词 *all* 处于限定词 *the* 之前，例如 *all the president's men*）。NOINDEFDETERMINER 表示像 *furniture*（家具）这样的单词，它出现时前面不带非确定性的限定词 *a*。SV，SVO 和 SVOO 表示动词的次范畴化（subcategorization）或补语化（complementation）模式。我们将在第 9 章和第 11 章中讨论次范畴化，这里只简单地解释一下 SV，SVO 和 SVOO 的含义。SV 表示只带主语的动词（例如 *nothing occurred*），SVO 表示带主语和宾语的动词（例如 *I showed the film*），SVOO 表示带一个主语和两个补语的动词（例如 She showed her the ball）。

在标注算法的第一个阶段，每个单词都要通过双层的词表转录机，然后该单词的所有可能的词类都返回到它的条目上。例如，短语 *Pavlov had shown that salivation...* 将返回如下的表（每一行标出一类可能的标记，正确的标记用粗体标出）：

Pavlov	**PAVLOV N NOM SG PROPER**
had	**HAVE V PAST VFIN SVO**
	HAVE PCP2 SVO
shown	**SHOW PCP2 SVOO SVO SV**
that	ADV
	PRON DEM SG
	DET CENTRAL DEM SG
	CS
salivation	**N NOM SG**
...	

然后，把大约 1100 个约束应用于输入的句子，以便除掉那些不正确的词类；在上面的这个表中，粗体的条目表示所期望的结果，其中 *had* 的正确标记为过去时标记（不是分词），*that* 的正确标记为标补语（CS）。这样的约束也可以按否定的方式来使用。例如，一个约束可以把 *that* 的除了 ADV

（强调副词）的意义［在句子 *It isn't <u>that</u> odd*（这不是那么古怪）中的 *that* 的意义］之外的一切内容都清除了。这里是一个简单的约束的实例：

```
ADVERBIAL-THAT RULE
Given input: "that"
if
   (+1 A/ADV/QUANT); /* 如果下一个单词是adj，adverb或quantifier */
   (+2 SENT-LIM);      /* 并且它的后面是句子的边界 */
   (NOT -1 SVOC/A); /* 并且它的前一个单词不是像 */
                /*'consider'这样的动词。这种动词允许adjs作为宾语补足语 */
then eliminate non-ADV tags
else eliminate ADV tag
```

在这个规则中，*if* 后面的前两个语句查看 *that* 是否直接处于句子末尾的形容词、副词或数量词之前。在其他的所有场合，副词读法都将被清除。最后一个语句清除 *that* 前面是 *consider* 或 *believe* 等动词的情况，这些动词能够带一个名词和一个形容词；这是为了避免把下面的实例中的that标注为副词：

I consider that odd

另外的规则用于对 *that* 的标补语用法的约束。当前面的单词是要求补语的动词时（如 *believe, think, show*），并且 *that* 后面跟着的是一个名词短语和一个定式动词的开头，*that* 用做标补语。

我们这里的描述把 ENGTWOL 的体系结构大大地简化了。在 ENGTWOL 中还包括概率约束，并且还使用了我们现在还没有讨论过的其他句法信息。有兴趣的读者可以参阅Karlsson et al.(1995)。

8.5　随机词类标注

很久以前就有人使用概率方法来进行标注了，首先使用概率来进行标注的是Stolz et al.,(1965)。使用 Viterbi 解码的完全概率标注系统出自 Bahl and Mercer(1976)。在 20 世纪 80 年代，各种随机标注系统纷纷建立起来(Marshall, 1983; Garside, 1987; Church, 1988; DeRose, 1988)。本节将描述一种特定的随机标注算法，这种算法一般称为隐马尔可夫模型或 HMM 标注算法。从直觉上来说，所有的随机标注算法都可以简单地概括为我们在前面讨论过的“对这个单词选取最可能的标记”这种方法，这种方法的基础是在第 5 章中介绍过的贝叶斯的框架。

对于一个给定的句子或单词序列，HMM标注算法选择使得下面的公式为最大值的标记序列：

$$P(\text{word}|\text{tag}) * P(\text{tag}|\text{previous } n \text{ tags}) \tag{8.1}$$

后面将解释这个特殊的等式，并且通过实例来说明它。HMM标注算法一般是针对一个句子而不是一个单词来选择标记序列的，但为了教学上的方便，我们首先来看一看HMM标注算法怎样把一个标记指派给一个单词。我们首先给出基本的等式，然后通过一个例子来使用这个等式，最后再给出为什么要使用这个等式的说明。

这种类型的二元语法HMM标注算法对于单词 w_i 选择标记 t_i，使得对于给定的前面的标记 t_{i-1} 和当前单词 w_i，其概率最大：

$$t_i = \underset{j}{\text{argmax}}\, P(t_j|t_{i-1}, w_i) \tag{8.2}$$

我们在下面还要讨论某些简化的马尔可夫假定，现在我们根据等式（8.2），给出对于一个单独标记的如下的 HMM 等式：

$$t_i = \underset{j}{\operatorname{argmax}} P(t_j|t_{i-1})P(w_i|t_j) \tag{8.3}$$

8.5.1 说明问题的一个例子

让我们通过例子来进行说明。在下面的例子中，使用一个HMM标注算法给单词*race*指派恰当的标记（两个例句都取自 Brown 语料库，但稍微进行了简化）：

Secretariat/NNP is/VBZ expected/VBN to/TO **race**/VB tomorrow/NN[1] (8.4)

（秘书处希望明天进行比赛）

People/NNS continue/VBP to/TO inquire/VB the/DT reason/NN
for/IN the/DT **race**/NN for/IN outer/JJ space/NN (8.5)

（人们继续询问外层空间竞赛的理由）

在第一个例子中 *race* 是一个动词（VB），在第二例子中 *race* 是一个名词（NN）。

为了解释这个例子，我们假定 *race* 周围的单词都已经由某种机制进行了最好的标注作业，它们都得到了恰如其分的标记，而只有单词*race*是没有标记的。HMM标注算法的二元语法简单地假定标注问题可以通过观察周围的单词和标记来解决。我们在考虑给race指派一个标记的问题时，只给出如下的子序列：

to/TO race/???
the/DT race/???

现在来看看如何把等式应用于这个例子，以求出 *race* 的标记。式（8.3）说明，如果试图在序列 *to race* 中对 *race* 的标记在 NN 和 VB 之间进行选择，应该选择下面的两个概率中比较大的一个：

$$P(\text{VB}|\text{TO})P(\text{race}|\text{VB}) \tag{8.6}$$

和

$$P(\text{NN}|\text{TO})P(\text{race}|\text{NN}) \tag{8.7}$$

等式（8.3）以及它的实例式（8.6）和式（8.7）中都有两个概率，一个概率是标记序列概率 $P(t_i|t_{i-1})$，一个概率是单词的似然度 $P(w_i|t_j)$。对于 *race* 来说，标记序列概率 $P(\text{NN}|\text{TO})$和 $P(\text{VB}|\text{TO})$就是“对于给定的前面的标记，期望*race*是动词（名词）的概率有多大？”这个问题的答案。通过从一个语料库中进行计数和归一化可以计算这些概率。我们可以预期，动词比名词更多地跟随在TO之后，因为不定式动词（*to race*, *to run*, *to eat*）在英语中很普遍。名词也可能跟随在TO之后（*walk to school*, *related to hunting*），但是这种情况不太普遍。

为了更充分地确信这种预期，我们把Brown语料库和Switchboard语料库结合起来观察，得到如下的概率（概率表明，在TO之后动词出现的概率是名词的15倍）：

$$P(\text{NN}|\text{TO}) = 0.021$$
$$P(\text{VB}|\text{TO}) = 0.34$$

等式（8.3）以及实例等式（8.6）和等式（8.7）中的第二部分是词汇的似然度，即单词 *race* 对于给定标记的似然度，也就是 $P(\text{race}|\text{VB})$和 $P(\text{race}|\text{NN})$。注意，这个似然度项目不是问“对于这个单词，哪一个是它最可能的标记？”也就是说，这个似然度项目不是 $P(\text{VB}|\text{race})$。我们应该计算的

① 在某些语料库中（包括 Penn Treebank 这样的语料库），有时把诸如 tomorrow 和 Monday 这样的副词标注为 NN。

似然度是 $P(\text{race}|\text{VB})$。这个概率与我们的直觉有些相左，它回答的问题是："如果我们期望一个动词，那么这个动词是 *race* 的可能性是多少？"

这里是把 Brown 语料库和 Switchboard 语料库结合起来计算出的词汇似然度：

$$P(\text{race}|\text{NN}) = 0.000\,41$$
$$P(\text{race}|\text{VB}) = 0.000\,03$$

如果把词汇似然度与标记序列概率相乘，可以看出，尽管是HMM标注算法的简单二元语法，也能够把 *race* 的标记正确地确定为 VB，即使 *race* 的含义为 VB 的可能性比较小：

$$P(\text{VB}|\text{TO})P(\text{race}|\text{VB}) = 0.000\,01$$
$$P(\text{NN}|\text{TO})P(\text{race}|\text{NN}) = 0.000\,007$$

8.5.2　实际的 HMM 标注算法

我们说过，一个真正的HMM标注算法不应该只针对一个单独的单词选择最好的标记，而应该针对整个句子选择最好的标记序列。我们已经有了针对一个单词的HMM标注算法的直觉，现在给出其完全等式。一般来说，我们使用 Viterbi 近似方法，为每个句子选择概率最大的标记序列。因此，这种方法假定，对于句子中给定的单词序列(W)，我们来计算每个句子中概率最大的标记序列 $T = t_1, t_2, \cdots, t_n$：

$$\hat{T} = \underset{T \in \tau}{\operatorname{argmax}} P(T|W)$$

根据贝叶斯定理，$P(T|W)$可以表示为：

$$P(T|W) = \frac{P(T)P(W|T)}{P(W)}$$

因此，我们试图选择标记序列，使得$\frac{P(T)P(W|T)}{P(W)}$最大：

$$\hat{T} = \underset{T \in \tau}{\operatorname{argmax}} \frac{P(T)P(W|T)}{P(W)}$$

因为对于给定单词序列，我们要找出一个句子的最可能的标记序列，所以单词序列的概率$P(W)$对于每个标记序列都是相同的，我们可以忽略它，有：

$$\hat{T} = \underset{T \in \tau}{\operatorname{argmax}} P(T)P(W|T)$$

从如下的概率的链规则出发，我们来进行 N 元语法假设：

$$P(T)P(W|T) = \prod_{i=1}^{n} P(w_i|w_1 t_1 \cdots w_{i-1} t_{i-1} t_i) P(t_i|w_1 t_1 \cdots w_{i-1} t_{i-1})$$

正如在等式(6.9)中为了给单词序列的概率建立模型所做的那样，现在再做N元语法假设。由于三元语法模型使用得最为广泛，我们来定义三元语法模型。首先，我们简单地假定单词的概率是独立于它们的标记的：

$$P(w_i|w_1 t_1 \cdots w_{i-1} t_{i-1} t_i) = p(w_i|t_i)$$

其次，假定标记的历史能够用最近的两个标记来近似表示：

$$P(t_i|w_1 t_1 \cdots w_{i-1} t_{i-1}) = P(t_i|t_{i-2} t_{i-1})$$

这样，选择标记序列的最大值为：

$$P(t_1)P(t_2|t_1)\prod_{i=3}^{n}P(t_i|t_{i-2}t_{i-1})[\prod_{i=1}^{n}P(w_i|t_i)]$$

通常可以使用最大似然度估计从相对频度来估计这些概率：

$$P(t_i|t_{i-2}t_{i-1}) = \frac{c(t_{i-2}t_{i-1}t_i)}{c(t_{i-2}t_{i-1})}$$

$$P(w_i|t_i) = \frac{c(w_i,t_i)}{c(t_i)}$$

这个模型也可以进行平滑（例如，使用第6章中的回退或删除插值算法来进行平滑），以避免零概率。

使用第7章中介绍的Viterbi算法可以找出概率最大的标记序列。

Weischedel et al.(1993)和DeRose(1988)报告，使用这种算法的准确率可达大约96%。

过去我们看到的HMM标注系统很多都是使用手工标注的数据来训练的。Kupiec(1992)和Cutting et al.(1992a)以及其他一些文献都表明，也可以在没有标记的数据上使用第7章和附录D中介绍的EM算法来训练HMM标注系统。这些标注系统仍然从词典开始，词典中要指出什么样的单词可以指派什么样的标记；然后使用EM算法对于每个标记学习单词似然度的功能以及标记转换概率。不过，Merialdo(1994)的实验表明，尽管只用少量的训练数据，用手工标注训练出的标注系统也比通过EM训练出的标注系统的工作情况更好。因此，EM训练出的“纯粹的HMM”标注系统大概只有在没有可用的训练数据的情况下才是最适用的，例如，当前面没有手工标注的数据来对语言进行标注时，就可以使用EM算法来训练。

方法论探讨：标注系统的评价

评价标注系统时，通常可以把标注系统和人工标注的黄金标准（gold standard）的测试集相比较，评价的基础是正确百分比（percent correct），也就是标注系统测试集中和黄金标准中的所有标记一致的百分比。对于诸如Penn Treebank这样的简单标记集，当前大多数标注算法的正确率（正确百分比）大约为96%~97%；然后，人工标注者可以使用手工方式对标注过的语料库进行后加工。

96%的正确率好不好呢？因为标记集和任务不同，标记的使用情况可以根据下界底线（baseline）和上界顶线（ceiling）来进行比较。确定顶线的一个办法是看人完成这个任务能达到什么程度。例如，Marcus et al.(1993)发现，Brown语料库中的Penn Treebank的标记与人工标注者的一致程度为96%~97%。这意味着，黄金标准允许错误的幅度为3%~4%，不可能达到100%的精确度。但是，Voutilainen(1995, p.174)的两个实验发现，如果允许人讨论标记，他们可以达到与标记100%一致的程度。

关键概念#6　人的顶线　当使用人的黄金标准来评价一个分类算法时，检查人与标准的一致率。

Gale et al.(1992)提出的标准底线（在词义排歧这个稍微不同的背景下提出的）是对每个歧义单词选择一元语法的最可能的标记。每个单词的最可能的标记可以根据手工标注的语料库来计算（这个语料库可能与所评价的标注系统的训练语料库相同）。

关键概念#7　一元语法底线　当设计一个新的分类算法时，始终要把它与一元语法底线相比较（给每个词例指派在训练集中出现频度最高的类别）。

Charniak et al.(1993)说明，这个底线算法稍微加以平滑，就可以达到90%~91%的精确率！自从Harris(1962)以来，标注算法就已经考虑到了关于标记频度的问题。

8.6　基于转换的标注

基于转换的标注有时又称为 Brill 标注，它是机器学习中的基于转换的学习（Transformation-Based Learning，简称 TBL）方法的一个实例(Brill, 1995)，并且也从基于规则的标注算法和随机标注算法中得到了启示。与基于规则的标注算法相似，TBL是基于规则的，它要指出什么样的标记可以指派给什么样的单词。但是，TBL 又与随机标注算法相似，TBL是一种机器学习技术，其中的规则是从数据自动推导出来的。与某些但不是全部的HMM标注算法相似，TBL是一种有指导的学习技术，在标注之前需要有一个训练语料库。

为了理解TBL的整个构架，Samuel et al.(1998a)曾经给我们提供过一个很有用的类比，并指出这个类比是从Terry Harvey那里借用过来的。想像一下，一位女艺术家要以蓝天为背景画一间白色的房子，房子上有绿色的装饰。假定这幅画的大部分都是天空，因此这幅画的大部分都应该是蓝色的。开始时，这位女艺术家使用很粗的画笔把整块油画布涂成蓝色。然后，她用较小的白色画笔来调整画面上的东西，并且把整个房子涂成白色。这时，她只是给整个房子着色，用不着担心棕色的屋顶、蓝色的窗子或者绿色的山墙。然后，她才取一只更小的棕色画笔来给屋顶着色。接着，她把蓝色的颜料蘸到一只小画笔上，在谷仓上画出蓝色的窗子。最后，她拿一只很细的绿色画笔给山墙做装饰。

这位画家开始时用粗画笔覆盖了大块的油画布，而不是首先给各个区域分别着色，这些区域是要以后重新着色的。下一层的颜色占油画布的区域较小，所造成的“错误”也较小。每个新的层使用的画笔越来越细，修改图画的区域越来越小，因而产生的错误也越来越小。TBL 所用的方法与这位女画家的方法在某种意义上是相同的。TBL 算法有一套标注规则。语料库首先用比较宽的规则来标注，这些规则也就是在大多数场合使用的规则。然后，再选择稍微特殊的规则来修改原来的某些标记。接着，再使用更窄的规则来修改数量更少的标记（其中某些标记可能是前面已经修改过的标记）。

8.6.1　怎样应用 TBL 规则

让我们来看一看Brill(1995)的标注算法使用的一些规则。在使用这些规则之前，标注系统已经给每个单词标上了最可能的标记。我们可以从标注语料库中得到这些最可能的标记。例如，在Brown语料库中，*race* 的最可能的标注为名词：

$$P(\text{NN}|\text{race}) = 0.98$$
$$P(\text{VB}|\text{race}) = 0.02$$

这意味着，在上面看到的关于 *race* 的两个例子中，两个 *race* 的编码都是 NN。在第一种情况下，这是错误的，因为 NN 是不正确的标记：

is/VBZ expected/VBN to/TO race/NN tomorrow/NN　　(8.8)

在第二种情况下，这个 *race* 被正确地标注为 NN：

the/DT race/NN for/IN outer/JJ space/NN　　(8.9)

在选择了最可能的标记之后，Brill标注算法应用它的转换规则。当应用转换规则时，Brill的标注系统学习到一个正好应用于改正 *race* 的错误标记的规则，这条规则是：

Change **NN** *to* VB *when the previous tag is* TO
（当前面的标记为 TO 时，把 NN 改成 VB）

这条规则正好满足条件，它将把 *race/NN* 改成 *race/VB*，因为 *race* 的前面是 *to/TO*：

expected/VBN to/TO race/NN → expected/VBN to/TO race/VB （8.10）

8.6.2 怎样学习 TBL 规则

Brill的TBL算法包括三个阶段。在第一个阶段，首先把每个单词标上最可能的标记。然后，在第二个阶段，检查每个可能的转换，并选择那个能够最大程度地改善标注的转换。最后，在第三个阶段，根据这个规则，对数据进行重新标注。后面的两个阶段重复进行，直到达到某个标准，使得不能再继续充分地改善前一轮的结果为止。注意，在第二个阶段，要求 TBL 知道每个单词的正确标记是什么，这意味着 TBL 是一种有指导的学习算法。

TBL过程的输出是一个转换的有序表；这些转换组成一个“标注过程”，并可应用于新的语料库。从原则上讲，可能的转换这个集合是无限的，因为我们能够想像存在着这样的转换：transform NN to VB if the previous word was 'IBM' and the word 'the' occurs between 17 and 158 words before that（如果前一个单词是 'IBM'，并且单词 'the' 出现在前面 17 至 158 个单词之间，则把 NN 转换成 VB）。但是，TBL需要考虑每个可能的转换，以便找出在整个算法的每一轮中最好的转换。这样，这种算法就需要一种办法来限制这个转换集合。这个办法就是设计一个称为“模板”（template）的小集合，这个模板也就是转换的摘要。每个可允许的转换就是模板的一个实例。图8.9列出了Brill的模板集合。图8.10给出了学习转换的算法。

The preceding (following) word is tagged **z**.
The word two before (after) is tagged **z**.
One of the two preceding (following) words is tagged **z**.
One of the three preceding (following) words is tagged **z**.
The preceding word is tagged **z** and the following word is tagged **w**.
The preceding (following) word is tagged **z** and the word two before (after) is tagged **w**.

图 8.9 Brill(1995)的模板。每条规则开始都是 *Change tag* **a** *to tag* **b** *when*:...
（当……时，把标记**a**改变为标记**b**）。变量 **a, b, z, w** 在词类范围内取值

在图8.10的核心部分是两个函数GET_BEST_TRANSFORMATION和GET_BEST_INSTANCE。GET_BEST_TRANSFORMATION 使用一个潜在的模板表来进行调用。对于每个模板，再调用 GET_BEST_INSTANCE。GET_BEST_INSTANCE迭代地检查每个模板中的每个可能的实例，对于标记变量 **a, b, z, w**，填入特定的值。

在实际中，还有一些办法可以提高算法的效率。例如，模板和实例转换可以采用数据驱动的方式来进行。如果一个转换改善了某个单词的标记，就可以把它提出来作为转换的实例。在训练语料库中使用潜在可能的转换预先给单词制作索引，可以明显地提高搜索的效率。Roche and Schabes (1997a)说明了如何把每个规则转成一个有限状态转录机，并把所有的转录机组合起来，从而提高标注系统的速度。

图 8.11 说明了使用 Brill 的原来的标注系统学习到的一些规则。

```
function TBL(corpus) returns transforms-queue
 INTIALIZE-WITH-MOST-LIKELY-TAGS(corpus)
 until end condition is met do
  templates ← GENERATE-POTENTIAL-RELEVANT-TEMPLATES
  best-transform ← GET-BEST-TRANSFORM(corpus, templates).
  APPLY-TRANSFORM(best-transform, corpus)
  ENQUEUE(best-transform-rule, transforms-queue)
 end
 return(transforms-queue)
```

```
function GET-BEST-TRANSFORM(corpus, templates) returns transform
 for each template in templates
   (instance, score) ← GET-BEST-INSTANCE(corpus, template)
   if (score > best-transform.score) then best-transform ← (instance, score)
 return(best-transform)
```

```
function GET-BEST-INSTANCE(corpus, template) returns transform
 for from-tag ← from tag−1 to tag−n do
  for to-tag ← from tag−1 to tag−n do
   for pos ← from 1 to corpus-size do
    if (correct-tag(pos) == to-tag && current-tag(pos) == from-tag)
        num-good-transforms(current-tag(pos−1))++
    elseif (correct-tag(pos)==from-tag && current-tag(pos)==from-tag)
        num-bad-transforms(current-tag(pos−1))++
   end
   best-Z ← ARGMAX_t(num-good-transforms(t) - num-bad-transforms(t))
   if(num-good-transforms(best-Z) - num-bad-transforms(best-Z)
            > best-instance.Z) then
     best-instance ← "Change tag from from-tag to to-tag
                      if previous tag is best-Z"
return(best-instance)
```

```
procedure APPLY-TRANSFORM(transform, corpus)
for pos ← from 1 to corpus-size do
 if (current-tag(pos)==best-rule-from)
      && (current-tag(pos−1)==best-rule-prev))
   current-tag(pos) = best-rule-to
```

图 8.10　标记学习的 TBL 算法。GET_BEST_INSTANCE 用于改变除了 *Change tag from X to Y if previous tag is Z*（如果前一个标记是 Z，则把标记 X 改成标记 Y）之外的转换模板(Brill, 1995)

#	更改标记 从	更改标记 到	条件	例子
1	NN	VB	Previous tag is TO	to/TO race/NN → VB
2	VBP	VB	One of the previous 3 tags is MD	might/MD vanish/VBP → VB
3	NN	VB	One of the previous 2 tags is MD	might/MD not reply/NN → VB
4	VB	NN	One of the previous 2 tags is DT	
5	VBD	VBN	One of the previous 3 tags is VBZ	

图 8.11　Brill 标注系统(Brill, 1995)中前 20 条非词汇化的转换中的部分内容

方法论探讨：错误分析

为了改进计算模型，我们需要分析并了解错误发生的情况。在像词类标注这样的分类模式中，错误分析一般使用含混矩阵（confusion matrix），又称为列联表（contingency table）来进行。含有N种方式的分类任务的含混矩阵是一个$N \times N$的矩阵表，其中的单元(x, y)包含正确分类项目x被模型y分类的次数。例如，下面的表是Franz(1996)的HMM标注实验中的含混矩阵的一部分。这个含混矩阵的行表示正确的标记，列表示标注系统给出的假定的标记，含混矩阵的每个单元表示相应的x和y的标注总错误百分比。例如，4.4%的总错误表示这个错误是由于把VBN错误地标注为VBD引起的。表中常见的错误都用粗体字标出。

	IN	JJ	NN	NNP	RB	VBD	VBN
IN	-	0.2			0.7		
JJ	0.2	-	**3.3**	2.1	1.7	0.2	**2.7**
NN		**8.7**	-				0.2
NNP	0.2	**3.3**	**4.1**	-	0.2		
RB	**2.2**	2.0	0.5		-		
VBD		0.3	0.5			-	**4.4**
VBN		**2.8**				**2.6**	-

上面的含混矩阵以及Franz(1996)，Kupiec(1992)和Ratnaparkhi(1996)的有关错误分析说明，当前标注系统面临的主要问题是：

1. **NN–NNP–JJ错误** 这是名词前成分中最难区分的错误。正确地区分出名词对于信息检索和机器翻译都是至关重要的。
2. **RP–RB–IN错误** 这些标记都以卫星序列的形式直接出现在动词后面。
3. **VBD–VBN–JJ错误** 在局部分析（例如，通过过去分词发现被动形式）以及名词短语边界的正确标注中，区分这些标记是非常重要的。

8.7 其他问题

8.7.1 多重标记和多项词

在标注中出现了两个问题，一个问题是标记的不确定性（indeterminacy），另一个问题是含有多个部分的词（multi-part word）。当一个单词在多个标记之间是歧义的并且不可能消解歧义或消解歧义非常困难时，就出现了标记的不确定性问题。在这种情况下，某些标注系统允许使用多重标记（multiple tag）。Penn Treebank和英国国家语料库BNC就是这样做的。常见的不确定标记包括“形容词–过去时动词–过去分词”多重标记（JJ/VBD/VBN），“形容词–以名词做前修饰语的名词”多重标记（JJ/NN）。

第二个问题是含有多个部分的词，即多项词（multiple word）。例如，在标记集C5和C7中，允许“*in terms of*”这样的介词，把它当做单个的词来处理，但是把每个标记加上了一个数字：

in/II31 terms/II32 of/II33

最后，在某些标注语料库中把某些词分离开来。例如，在Penn Treebank和英国国家语料库BNC中，就把缩约形式和表示所有格的 's 与它们的词干分开了：

would/MD n't/RB

children/NNS 's/POS

8.7.2　未知词

我们讨论过的所有标注算法都要求有一部词典，词典中的每个单词都要列出它所有可能的词类。但是，正如在第4章中指出的，尽管在大型词典中也不可能包含每个可能出现的单词，但专有名词和首字母缩写词源源不断地被创造出来，甚至新的普通名词和动词也以惊人的速度不断地进入到语言中。因此，为了建立一个完全的标注系统，我们需要某些方法来猜测未知词的标记。

一种可能的最简单的未知词算法是，假定每个未知词在所有可能的标记中都是有歧义的，它们都具有相同的概率。因此，标注系统必须根据上下文相关的 POS 三元语法给未知词一个恰当的标记。稍微复杂一些的算法基于这样的思想：在未知词上的标记的概率分布与在训练集中只出现一次的单词的标记的概率分布非常相似，这样的思想是 Baayen and Sproat(1996)以及 Dermatas and Kokkinakis(1995)提出的。这些只出现一次的单词称为“一次性罕用词”（hapax legomena，它的单数形式是 hapax legomenon）。例如，未知词和一次性罕用词的相似之处在于，两者都最可能是名词，都跟随在动词之后，但它们与限定词和感叹词的分布都很不相同。这样，一个未知词的似然度 $P(w_i \mid t_i)$就可以根据训练集中所有一次性罕用词的平均分布来确定。我们曾经讲过，使用“我们看过一次的事物”来估计“我们从未看过的事物”的这种思想被证明是很有用的，这种思想在第6章中讨论 Witten-Bell 算法和 Good-Turing 算法时以关键概念“看过一次的事物”给出。

最强的未知词算法使用了关于单词拼写的信息。例如，以字母 *-s* 为结尾的单词最可能是复数名词（NNS），而以 *-ed* 为结尾的单词则倾向于是过去分词（VBN）。以大写字母开头的单词最可能是名词。Weischedel et al.(1993)使用了 4 类特殊的正词法特征，包括 3 个屈折词尾（*-ed*, *-s*, *-ing*）、32 个派生后缀（例如 *-ion*, *-al*, *-ive*, *-ly*）、4 个大写值（首字母大写，非首字母大写，等等）以及连字符规则。使用如下的等式来计算未知词的似然度：

$$P(w_i|t_i) = p(\text{unknown-word}|t_i) * p(\text{capital}|t_i) * p(\text{endings}|\text{hyph}/t_i)$$

在这个公式中，unknown-word 表示未知词，capital 表示大写，ending 表示词尾或后缀，hyph 表示连字符。

其他一些研究者不依赖于上述的手工设计的特征，而是使用机器学习的方法来推导出有用的特征。Brill(1995)使用了 TBL 算法，其中可允许的模板是根据正词法来定义的（单词的前 *N* 个字母，单词的后 *N* 个字母，等等）。该算法推导出了英语中所有的屈折特征、连字符规则以及诸如 *-ly* 和 *-al* 这样的派生特征。Franz(1996)使用了对数线性模型（loglinear model），这个模型包含更多的特征，例如单词的长度和各种前缀，并且还进一步包括各种特征之间的交互信息。

8.7.3　基于类的 *N* 元语法

现在我们已经有了一种方法给语料库中的每个单词自动指派一个类（class）。在这种情况下，我们有可能使用这样的信息来提升 *N* 元语法模型。基于类的 *N* 元语法（class-based N-gram）是 *N* 元语法的一个变体，使用POS（或其他）类序列的频度来帮助产生对于单词符号串的概率的估计，使这样的估计带有更多的信息。当基本的基于类的 *N* 元语法定义一个单词 w_n 的条件概率时，要根据它的历史。这个历史表现为两个因素的乘积，其中一个因素是对于前面给定的类的当前类的概率（建立在类的 *N* 元语法的基础上），另一个因素是对于给定的类的特定单词的概率。

$$P(w_n|w_{n-N+1}^{n-1}) = P(w_n|c_n)P(c_n|c_{n-N+1}^{n-1})$$

对于给定的类的特定单词的概率以及前面给定的类的当前类的概率，其最大似然度估计（MLE）可以按下面的公式来进行计算：

$$P(w|c) = \frac{C(w)}{C(c)}$$

$$P(c_i|c_{i-1}) = \frac{C(c_{i-1}c_i)}{\sum_c C(c_{i-1}c)}$$

基于类的N元语法可以依靠Penn标记集这样的标准标记集，也可以依靠具有特殊应用目标的标记集（例如，对于航空信息系统，可以使用类似CITY和AIRLINE的标记）。这样的类也可以通过在语料库中对单词进行聚类（clustering）的方法自动推导出来(Brown et al., 1992)。有些研究者指出，基于类的N元语法在减少语言模型单词错误率的困惑度方面是很有用的，特别是当把它们与正规的基于单词的N元语法混合起来使用时更有用(Jelinek, 1990; Kneser and Ney, 1993; Heeman, 1999; Samuelsson and Reichl, 1999)。

方法论探讨：通过κ计算一致性

使用“正确百分比”（percent correct）的指标来评价标注系统的一个问题是，这个指标不能控制标注任务的难易程度。例如，如果99%的标记都是NN，那么，并不能认为得到99%的正确率是很好的结果；我们只要简单地猜测NN就可以得到99%的正确率了。这种情况说明，当标注系统在不同的测试集上运行或者它们的任务各不相同时，要对它们进行比较，事实上是不可能的。在前面的方法论探讨中指出，有一个因素能够帮助我们把正确百分比的不同的值归一化，这个因素就是通过给定任务的一元语法底线来测定该任务的困难程度。

实际上，有一种称为kappa(κ)的评价统计方法已经考虑到了这个底线，这种方法能够从内部控制任务的复杂性(Siegel and Castellan, 1988; Carletta, 1996)。当把一个标注系统与黄金标准相比较时，或者特别是当对人的标注者彼此之间进行比较时，如果没有一个正确的回答，Kappa可以用来替代正确百分比。Kappa是两个分类系统一致的次数比例（偶然一致已经改正）与这两个分类系统可能一致的次数的最大比例（偶然一致已经改正）的比值：

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

P(A)是假设与标准一致的次数比例，也就是正确百分比；P(E)是假设与所期望的标准偶然地一致的次数比例，可以从某些其他知识来计算P(E)，或者从用于计算标记的实际含混矩阵来计算P(E)。κ的界限与正确百分比的界限一样，当不存在一致时（与偶然期望的目标不同），$\kappa = 0$；当完全一致时，$\kappa = 1$。

当没有“黄金标准”时，经常使用κ统计方法。例如，当对于一个主观性强的困难任务中的人工标注结果彼此进行比较时，就可以使用κ统计方法。在这种情况下，κ是一个非常有用的统计指标，它是“改正偶然一致的平均成对一致”。Krippendorf(1980)建议，当$\kappa > 0.8$时，就可以认为具有很好的相关性。

8.8 小结

本章介绍了词类和词类标注的基本思想。主要思想如下：

- 一般来说，各种语言都有一个相对小的词的封闭类，封闭类中的词通常都是高频度的，一般都是虚词。在词类标注中，它们可能有很大歧义。开放类的词一般包括各种类型的名词、动词和形容词。现在有一定数量的词类编码方案，所依据的标记集在40个标记到200个标记之间。

- 词类标注是给单词序列中的每个单词指派一个词类标记的过程。词类标注算法包括基于规则的标注算法和随机标注算法。基于规则的标注算法使用手写规则来区分歧义。随机标注算法或者是基于HMM的，或者是基于提示的（cue-based）。基于HMM的标注算法选择单词似然度与标记序列概率的乘积为最大的标记序列作为标注结果；基于提示的随机标注算法使用决策树（decision tree）或最大熵模型（maximum entropy model），并把它们与概率特征结合起来。
- 标注系统评价的通常办法是把系统对测试集的输出与人对于该测试集的标注进行比较。错误分析可以帮助我们准确地确定标注系统在哪些地方还不完善，从而进一步改善系统的性能。

8.9　文献和历史说明

最早的词类指派算法是 Zellig Harris 的剖析程序的一个部分，这个剖析程序是为 Zellig Harris 的“转换和话语分析课题”（Transformation and Discourse Analysis Project，简称 TDAP）设计的，于 1958 年 6 月到 1959 年 7 月在宾夕法尼亚大学实现(Harris, 1962)。过去的一些自然语言处理系统也使用过带有单词的词类信息的词典，但是没有描述如何进行词类歧义的消解。作为剖析程序的一部分，TDAP 使用了 14 条规则进行词类歧义消解，所用的词类标记序列成为了后来所有算法的雏形，该系统的运行考虑到了单词标记的相对频度的顺序。这个剖析 - 标注系统最近由 Joshi and Hopely(1999)以及 Karttunen(1999)再次实现，文献指出这个剖析程序实质上是作为一个层叠式的有限状态转录机来实现的。

在 TDAP 剖析程序之后，Klein and Simmons(1963)设计了“计算语法编码器”（Computational Grammar Coder，简称 CGC）。CGC 由 3 部分组成：一部词典、一个形态分析器和一个上下文歧义消解器。在 1500 个单词的小词典中，包括那些不能在简单的形态分析器中处理的个别的例外单词，还包括虚词以及不规则的名词、动词和形容词。形态分析器根据屈折和派生的后缀来给单词指派词类标记。在运行时，一个单词通过词典和形态分析器之后，产生出候选的词类集合。然后，使用包括500条上下文规则的规则集来对这个候选集合进行歧义消解，歧义消解的依据是环绕在歧义单词周围的无歧义单词组成的岛屿。例如，有一个规则说，在 ARTICLE（冠词）和 VERB（动词）之间只允许的词类序列是 ADJ-NOUN（形容词 - 名词），NOUN-ADVERB（名词 - 副词）或 NOUN-NOUN（名词 - 名词）。CGC 算法报道，对于 Scientific American（科学美国人）和一本儿童百科全书中的文章，应用了 30 个标记，标注正确率为 90%。

TAGGIT 系统(Greene and Rubin, 1971)是在 Klein and Simmons(1963)的基础上建立的。该系统使用了同样的体系结构，此外还扩大了词典的规模并增加了标记集的容量（增加到87个标记）。例如，下面是一个规则的样本。这个规则说明，在第三人称单数动词（VBZ）之前，单词 x 不能是复数名词（NNS）：

$$x \text{ VBZ} \rightarrow \textit{not} \text{ NNS}$$

TAGGIT 被用来标注 Brown 语料库，根据 Francis and Kučera(1982, p.9)，“语料库标注结果的正确率为 77%”（Brown 语料库的其他部分是用手工标注的）。

在 20 世纪 70 年代，Lancaster-Oslo/Bergen（LOB）语料库编制成功，这个语料库是与 Brown 语料库对应的英国英语的语料库。标注工作使用CLAWS标注算法(Marshall, 1983, 1987; Garside, 1987)来进行。CLAWS 算法是一个概率算法，可以看成是近似于 HMM 标注方法的一种算法。该算法使用标记的二元语法概率，但它不存储每个标记的单词似然度，而是给“tag|word”（标记|单词）标

上 *rare*（罕用），*infrequent*（低频度）或 *normally frequent*（正常频度）这样的等级符号，例如 rare (P(tag|word) < 0.01)，*infrequent* (P(tag|word) < 0.10)，或 *normally frequent*(P(tag|word) > 0.10)。

Church(1988)的概率标注算法 PARTS 非常接近于完全 HMM 标注算法。这个算法扩充了 CLAWS 的思想，对于每个“word|tag”偶对，全都指派相应的词汇概率，并使用 Viterbi 解码来发现标记序列。但是，与 CLAWS 算法一样，这个系统存储的是对于给定单词的某个标记的概率：

$$P(\text{tag}|\text{word}) * P(\text{tag}|\text{previous } n \text{ tags}) \quad (8.11)$$

而不使用对于给定标记的某个单词的概率：

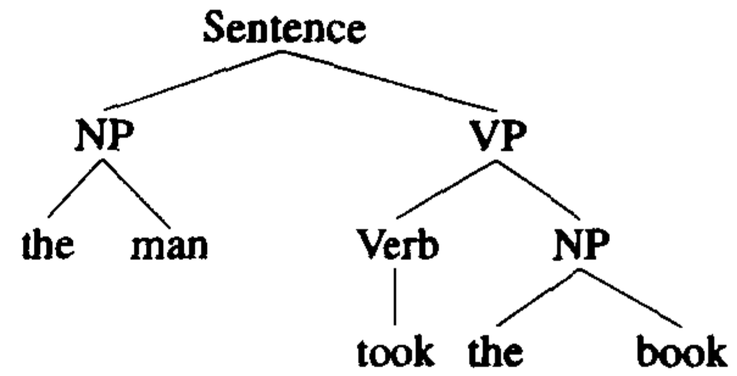
$$P(\text{word}|\text{tag}) * P(\text{tag}|\text{previous } n \text{ tags}) \quad (8.12)$$

后来的一些标注系统明确地使用隐马尔可夫模型，并且常常还同时结合使用 EM 训练算法(Kupiec, 1992; Merialdo, 1994; Weischedel et al., 1993)，包括使用变长马尔可夫模型(Schütze and Singer, 1994)。

新近的一些随机算法使用各种统计和机器学习的工具来估计标记或标记序列的概率，计算时要考虑大量的相关特征，例如相邻的单词、相邻的词类、各式各样的正词法特征和形态特征。然后把这些特征结合起来，或者使用决策树(Jelinek et al., 1994; Magerman, 1995)，或者使用最大熵算法(Ratnaparkhi, 1996)，或者使用对数线性模型(Franz, 1996)，或者使用线性分离子网络（SNOW）(Roth and Zelenko, 1998)来估计标记的概率。Brill(1997)提出了 TBL 算法的一种无指导学习算法的版本。

对于词类的历史有兴趣的读者可以参阅 Robins(1967)或者 Koerner and Asher(1995)有关语言学史的内容，特别是可以参阅 Householder(1995)最近的文章。

第9章　英语的上下文无关语法



The first context-free grammar parse tree (Chomsky, 1956)

If on a winter's night a traveler by Italo Calvino
Nuclear and Radiochemistry by Gerhart Friedlander et al.
The Fire Next Time by James Baldwin
A Tad Overweight, but Violet Eyes to Die For by G. B. Trudeau
Sometimes a Great Notion by Ken Kesey
Dancer from the Dance by Andrew Holleran

Six books in English whose titles are not constituents, from Pullum (1991, p. 195) ①

M. F. K. Fisher(1968)在散文 *Anatomy of Recipe*（处方的解剖学）中，曾经挖苦地评论说，引用“解剖学”这个词来说明对于某些事情或者某些问题的分析，是一种“时髦风尚”。*grammar*（语法）这个词也有相似的用法，在19世纪，用 *grammar* 这个词来描述某个知识领域的结构也是很时髦的，例如Busby(1818)的*A Grammar of Music*（音乐的语法）和Field(1888)的*A Grammar of Colouring*（色彩的语法）。近年来，*grammar* 这个词又重新出现了类似的用法，但现在一般是用 *the grammar*，而不像过去那样用 *a grammar* 了，例如 *The Grammar of Graphics*（制图的语法）和 *The Grammar of Conducting*（指挥的语法）。大概学者们在做学问时不像平常那样时髦，或者大概是由于*Grammar* 这个词本身的意思有了一点儿变化。现在，*Grammar* 的意思从“原则或结构的清单”变成了“作为有关领域的原则或结构”。根据第二个意思，本章中讨论“语法的语法”（*The Grammar of Grammar*）或者“句法的语法”（*The Grammar of Syntax*）。

“句法”（syntax）这个单词来自希腊语的 *sýntaxis*，含义是“放在一起或者进行安排”，指把单词安排在一起的方法。在前面几章中我们已经看到了各种有关句法的概念。第8章把词类范畴作为单词的等价类来讨论，第6章谈到了给词序建立模型的重要性，本章以及后面几章将介绍关于句法和语法的一些更复杂的概念。这里要介绍3个主要的新思想：组成性（constituency）、语法关系（grammatical relation）以及次范畴化和依存关系（subcategorization and dependency）。

① 这是六本英文书的书名，它们的英文原文在语法上成分不全，对应译文如下，供读者参考：

《如果在冬天的夜里，一个旅行者》（Italo Calvino）
《核与反射化学》（Gerhart Friedlander et al.）
《下一次的火焰》（James Baldwin）
《小男孩超重了，可是紫罗兰的眼光却消逝了》（G.B. Trudeau）
《有时，一个伟大的观念》（Ken Kesey）
《从舞会来的舞者》（Andrew Holleran）

组成性的基本思想在于，单词的组合可以具有像一个单独的单位或短语那样的功能，这样的单词组合称为成分（constituent）。例如，我们可以看到，称为名词短语的单词组合通常作为一个单位来使用；名词短语包括单独的单词，如 *she* 或 *Michael*，还包括短语，如 *the house*，*Russian Hill* 以及 *a well-weathered three-story structure*（耐风雨的三层结构）。本章将介绍上下文无关语法（context-free grammar）的使用，这种形式化的语法可以帮助我们为这种组成性的事实建立模型。

语法关系是传统语法关于主语（SUBJECTS）和宾语（OBJECTS）的思想的形式化。在下面的句子中

She ate a mammoth breakfast. (9.1)

（她吃了一顿丰盛的早餐）

名词短语*she*是主语，*a mammoth breakfast*是宾语。在本章中，当我们讨论一致关系（agreement）时将介绍语法关系，并且将在第 11 章中进一步展开语法关系。

次范畴化和依存涉及到单词和短语之间的某些类别的关系。例如，动词 *want* 后面可以跟不定式（*I want to fly to Detroit*）或名词短语（*I want a flight to Detroit*），而动词 *find* 后面不能跟不定式（**I found to fly to Dallas*）。这样的事实称为动词的次范畴，我们将在 9.7 节中讨论这个问题，并在第 11 章中再次讨论它。

上述的所有句法知识都可以通过基于上下文无关语法的各种语法来建立模型。因此，上下文无关语法是自然语言（以及计算机语言）的很多句法模型的支柱（backbone）。在大多数自然语言理解模型、语法检查模型以及最近的语音理解模型中，上下文无关语法都是不可缺少的重要组成部分。上下文无关语法对于句子中单词之间的复杂关系有足够强大的表达能力，采用上下文无关语法剖析句子的各种有效算法也是计算机可循的（我们将在第 10 章中看到这一点）。然后，在第 12 章中，我们将介绍上下文无关语法的一个概率版本，即概率上下文无关语法。概率上下文无关语法可以为人类句子处理的很多方面建立模型，并为语音识别提供一个精致的语言模型。

除了介绍语法的形式化方法之外，本章还将概述英语语法。我们将采用航空旅行信息系统（Air Traffic Information System，简称 ATIS）中的句子作为例子来建模。ATIS 系统是一个能够帮助订飞机票的口语对话系统。用户可以通过与系统对话的方式预订飞机票，对话的语句有一定的限制，例如用户可以说这样的语句：*I'd like to fly from Atlanta to Denver*。在 20 世纪 90 年代初建设 ATIS 系统时，美国政府资助了其中一些困难的研究项目，因此收集了大量的数据，并根据这些数据进行了很多有意义的研究工作。在本章中用来建立模型的句子都是这个系统中用户对话时使用的句子。

9.1 组成性

英语中的单词是怎样组合在一起的呢？我们怎样才能知道这些单词已经真正地组合在一起了呢？让我们来考虑一种标准的组合。这种组合通常称为名词短语（noun phrase），有时称为名词词组（noun group）。这是包围着名词的单词序列，这个单词序列至少由一个名词组成。这里是一些名词短语的例子（由 Damon Runyon 提供）[①]。

three parties from Brooklyn
a high-class spot such as Mindy's
the Broadway coppers
they

① 对于下面的章节中的外语例子，我们一般不再给出汉语译文，只给出一些比较复杂的句子的汉语译文——译者注。

Harry the Horse

the reason he comes into the Hot Box

我们怎样知道这些单词是组合在一起(或形成一个成分)的呢？一个明显的依据是它们都可以出现在相同的句法环境中，例如，它们都可以出现在动词之前。

three parties from Brooklyn *arrive*...

a high-class spot such as Mindy's *attracts*...

the Broadway coppers *love*...

they *sit*

但是，整个名词短语出现在动词之前，并不意味着构成名词短语的每个单独的单词都具有这种性质。下面都是不合英语语法的句子（前面提到过，我们使用星号“*”来表示那些不合英语语法的单词片断）：

*form *arrive*...

*as *attracts*...

*the *is*

*spot *is*...

所以，为了正确地描述关于英语中这些单词的顺序的事实，我们必须这样表述：“名词短语能够出现在动词之前”。

另一个关于组成性的依据来自称为前置（preposed）或后置（postposed）的结构。例如，前置短语*on September seventeenth*在下面的例子中可以放在不同的位置，包括在句子的开头前置和在句子的结尾后置：

On September seventeenth, I'd like to fly from Atlanta to Denver

I'd like to fly On September seventeenth from Atlanta to Denver

I'd like to fly from Atlanta to Denver On September seventeenth

但是，尽管整个短语可以放在不同的位置，组成这个短语的个别的一个或多个单词并不能放在不同的位置：

*On September, I'd like to fly seventeenth from Atlanta to Denver

*On I'd like to fly September seventeenth from Atlanta to Denver

*I'd like to fly on September from Atlanta to Denver seventeenth

在9.11节中，我们将根据上下文无关语法模拟递归结构的能力，说明上下文无关语法的其他问题。

还有其他方面的依据能够证明单词组合的功能与该组合中的一个单独组成成分的功能相同。参阅Radford(1988)，这是一个很好的综述。

9.2　上下文无关规则和树

模拟英语和其他自然语言成分结构的最常用数学系统是上下文无关语法（Context-Free Grammar，简称CFG）。上下文无关语法又称为短语结构语法（Phrase-Structure Grammar），而它的

形式化方法等价于Backus-Naur范式（Backus-Naur Form，简称BNF）。把一种语法建立在成分结构基础上的这种思想，可以追溯到心理学家Wilhelm Wundt(1900)，但是直到Chomsky(1956)才把这种思想形式化，Backus(1959)也独立地进行了相同的工作。

一个上下文无关语法由一套规则（rule）或产生式（production）以及单词和符号的一个词表（lexicon）组成，每个规则表示语言中的符号的组成和排序方式。下面的产生式表示一个NP（或者名词短语）可以由一个专有名词（*ProperNoun*）组成，或者由一个限定词（*Det*）后面跟着一个名词性成分（*Nominal*）组成；一个名词性成分可以是一个或多个名词。

$$NP \rightarrow Det\ Nominal \tag{9.2}$$

$$NP \rightarrow ProperNoun \tag{9.3}$$

$$Nominal \rightarrow Noun \mid Noun\ Nominal \tag{9.4}$$

上下文无关规则可以按层级嵌套，所以前面的规则可以与下面的表示词汇事实的规则结合起来:

$$Det \rightarrow a \tag{9.5}$$

$$Det \rightarrow the \tag{9.6}$$

$$Noun \rightarrow flight \tag{9.7}$$

在CFG中所用的符号分为两类。与语言中的单词相对应的符号（如*the*和*nightclub*）称为终极符号（terminal symbol）；词表是引入这些终极符号的规则的集合。表示这些终极符号的聚类或概括性的符号称为非终极符号（non-terminal）。在每个上下文无关规则中，箭头（→）右边的项是一个或者多个终极符号和非终极符号构成的有序表，而箭头的左边是一个单独的非终极符号，表示某种聚类或概括性。注意，在词表中，与每个单词相关联的非终极符号是它们的词类范畴，或者是在第8章中定义的词类。

通常可以按两种方式来考虑CFG，把它想像成生成句子的装置，或者把它想像成对于给定的句子指派结构的装置。作为句子的生成装置，我们可以把“→”读为“用右边的符号串来重写左边的符号”。这样，如果开始时的符号是

NP,

我们可以使用规则（9.2），把*NP*重写为:

Det Nominal,

然后使用规则（9.4），继续重写为:

Det Noun,

最后，使用规则（9.5）和规则（9.7），重写为:

a flight,

这时，我们说，符号串*a flight*可以从非终极符号*NP*推导（derived）出来。因此，CFG可以用来随机地生成一系列的符号串。这种规则展开的序列称为单词符号串的一个推导（derivation）。通常我们用一个剖析树（parse tree）来表示一个推导（一般是倒过来把树的根置于上方）。图9.1是表示上述推导的一个剖析树。

CFG定义的形式语言是从指定的初始符号（start symbol）开始推导出来的符号串的集合。每个语法必须有一个指定的初始符号，这个初始符号通常称为*S*，由于上下文无关语法通常用来定义

句子，所以 *S* 通常可以解释为“句子”（sentence）。在某个简化的英语语法中，由 *S* 推导出来的符号串的集合就是句子的集合。

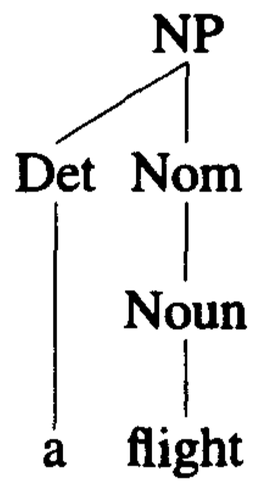


图 9.1　a flight 的剖析树

现在，我们给语法增加几个展开*S*的级别较高的规则以及几个其他规则。其中一个规则用来表示一个句子可以由一个名词短语和一个动词短语构成的事实：

$$S \rightarrow NP\ VP \quad \text{I prefer a morning flight}$$

英语中的一个动词短语可以由一个动词后面跟着有关的其他成分组成，有一种动词短语可以由一个动词后面跟着一个名词短语组成：

$$VP \rightarrow Verb\ NP \quad \text{prefer a morning flight}$$

或者由动词后面跟着一个名词短语和一个介词短语组成：

$$VP \rightarrow Verb\ NP\ PP \quad \text{leave Boston in the morning}$$

或者由动词后面只跟着一个介词短语组成：

$$VP \rightarrow Verb\ PP \quad \text{leaving on Thursday}$$

介词短语一般由一个介词后面跟着一个名词短语组成。例如，在 ATIS 语料库中最常见的介词短语类型用于表示位置和方向：

$$PP \rightarrow Preposition\ NP \quad \text{from Los Angeles}$$

在 *PP* 内，*NP* 不一定总是表示方位的。*PP* 也经常用来表示时间和日期。*PP* 中还可以使用其他非常复杂的名词。下面给出了 ATIS 语料库中的 10 个例子：

to Seattle	on these flights
in Minneapolis	about the ground transportation in Chicago
on Wednesday	of the round trip flight on United Airlines
in the evening	of the AP fifty seven flight
on the ninth of July	with a stopover in Nashville

图9.2给出了词表的一个样本，图9.3总结了我们前面研究过的规则。我们把这个语法称为$\mathcal{L}_0$。注意，我们可以使用表示或者的符号“|”来表示非终极符号的不同展开方式。

我们可以使用这个语法来生成 ATIS 中的句子。从 *S* 开始，把它展开为 *NP VP*，然后随机地展开 *NP*（例如，把 *NP* 展开为 *I*），随机地展开 *VP*（例如，把 *VP* 展开为 *Verb NP*），一直到生成符号串 *I prefer a morning flight* 为止。图 9.4 是一个剖析树，表示句子 *I prefer a morning flight* 的整个推导过程。

Noun → *flights* | *breeze* | *trip* | *morning* | ...
Verb → *is* | *prefer* | *like* | *need* | *want* | *fly*
Adjective → *cheapest* | *non – stop* | *first* | *latest*
| *other* | *direct* | ...
Pronoun → *me* | *I* | *you* | *it* | ...
Proper-Noun → *Alaska* | *Baltimore* | *Los Angeles*
| *Chicago* | *United* | *American* | ...
Determiner → *the* | *a* | *an* | *this* | *these* | *that* | ...
Preposition → *from* | *to* | *on* | *near* | ...
Conjunction → *and* | *or* | *but* | ...

图 9.2 $\mathcal{L}_0$的词表

S	→	*NP VP*	I + want a morning flight
NP	→	*Pronoun*	I
	|	*Proper-Noun*	Los Angeles
	|	*Det Nominal*	a + flight
Nominal	→	*Noun Nominal*	morning + flight
	|	*Noun*	flights
VP	→	*Verb*	do
	|	*Verb NP*	want + a flight
	|	*Verb NP PP*	leave + Boston + in the morning
	|	*Verb PP*	leaving + on Thursday
PP	→	*Preposition NP*	from + Los Angeles

图 9.3 $\mathcal{L}_0$的语法，每个规则有短语示例

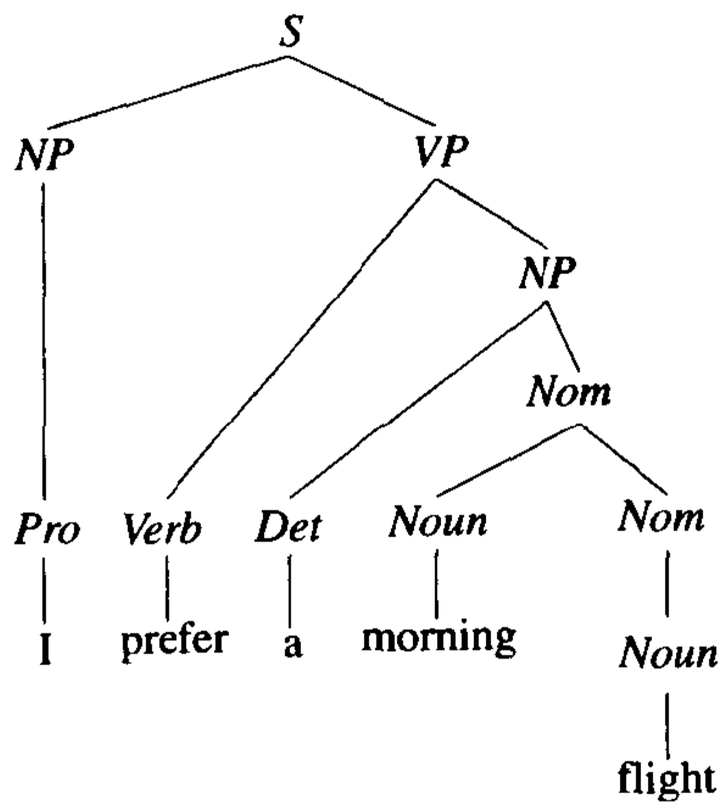


图 9.4 根据语法$\mathcal{L}_0$形成的句子 I prefer a morning flight 的剖析树

有时用更简洁的形式来表示剖析树会更方便，这种简洁形式称为“括号表示”（bracketed notation），这种表示法实质上是树的 LISP 表示法。图 9.4 中的剖析树的括号表示如下：

[*S* [*NP* [*Pro* I]] [*VP* [*V* prefer] [*NP* [*Det* a] [*Nom* [*N* morning [*Nom* [*N* flight]]]]]]

一个诸如$\mathcal{L}_0$的CFG定义了一个形式语言。我们在第2章中讲过，形式语言是符号串的集合。如果由一个语法推导出的句子处于由该语法定义的形式语言中，这个句子就是合语法的（grammatical）。如果不能被某一给定的形式语法推导出的句子不处于由该语法定义的形式语言中，这个句子就是不合语法的（ungrammatical）。对于所有的形式语言的句子来说，要描述它们是处于形式语言“之内”或是“之外”，其界限是很难划分清楚的，但形式语法还是描述自然语言实际工作情况的一种最简单的模型。在语言学中，使用形式语言来模拟自然语言的语法称为“生成语法”（generative grammar），因为语言是通过由语法“生成”的一切可能的句子的集合来确定的。

在本节结束时，我们采用一种简练的方式来描述上下文无关语法以及它生成的语言。一个上下文无关语法有四个参数［在技术上称为“四元组”（4-tuple）］：

1. 非终极符号的集合（或者“变量”）N
2. 终极符号的集合Σ（与N不相交）
3. 生成式的集合P，每个生成式的形式为$A \rightarrow \alpha$，其中A是非终极符号，α是由符号串的无限集$(\Sigma \cup N)^*$中的符号构成的符号串
4. 一个指定的初始符号S

语言通过推导的概念来定义。如果通过一系列的规则应用，一个符号串可以被重写为另一个符号串，我们就说这个符号串推导出另一个符号串。根据Hopcroft and Ullman(1979)，可以更加形式地说，如果$A \rightarrow \beta$是P中的一个产生式，α和γ是$(\Sigma \cup N)^*$中任意的符号串，我们就说$\alpha A \gamma$直接推导出$\alpha\beta\gamma$，或者$\alpha A \gamma \Rightarrow \alpha\beta\gamma$。推导可以由直接推导概括出来。设$\alpha_1, \alpha_2, \cdots, \alpha_m$是$(\Sigma \cup N)^*$中的符号串，$m \geqslant 1$，使得

$$\alpha_1 \Rightarrow \alpha_2, \alpha_2 \Rightarrow \alpha_3, \cdots, \alpha_{m-1} \Rightarrow \alpha_m \tag{9.8}$$

则α_1推导出α_m，或者$\alpha_1 \overset{*}{\Rightarrow} \alpha_m$。

这样，我们可以把由语法G生成的语言$\mathcal{L}_G$形式地定义为由指定的初始符号S推导出的终极符号构成的符号串的集合。

$$\mathcal{L}_G = W | w \text{ is in } \Sigma* \text{ and } S \overset{*}{\Rightarrow} w \tag{9.9}$$

把单词的符号串映射到剖析树的问题称为“剖析”（parse），我们将在第10章和第12章中研究剖析算法。

9.3　句子级的结构

本章的以下部分将介绍英语短语结构的一些更复杂的问题。为了前后一致，我们继续研究ATIS领域中的句子。由于篇幅所限，我们的讨论必须严格地限于把原理讲清楚，不展开讨论。我们强烈地建议读者参阅Quirk et al.(1985)，这是到本书落笔为止最好的英语语法参考书。

在前面的很小的语法$\mathcal{L}_0$中，我们只给出陈述句中的一个句子级的结构，如*I prefer a morning flight*。在英语中，还存在着大量的可能的句子结构，但最常见和最重要的有4种：陈述式结构（declarative structure）、命令式结构（imperative structure）、yes-no疑问式结构（yes-no question structure）、wh疑问式结构（wh-question structure）。

陈述式结构的句子有一个主语名词短语，后面跟着一个动词短语，例如*I prefer a morning flight*。陈述式结构的句子有很多不同的用法，第19章将具体讨论这些用法，这里只举出ATIS语料库中的一些例子：

The flight should be eleven a.m. tomorrow
I need a flight to Seattle leaving from Baltimore making a stop in Minneapolis
The return flight should leave at around seven p.m.
I would like to find out the flight number for the United flight that arrives in San Jose around ten p.m.
I'd like to fly the coach discount class
I want a flight from Ontario to Chicago
I plan to leave on July first around six thirty in the evening

命令式结构的句子通常以一个动词短语开头，并且没有主语。之所以称其为命令式句子，是因为这样的句子总是用于表示命令和建议。在 ATIS 领域中，它们是对系统的命令。

Show the lowest fare
Show me the cheapest fare that has lunch
Give me Sunday's flights arriving in Las Vegas from Memphis and New York City
List all flights between five and seven p.m.
List all flights from Burbank to Denver
Show me all flights that depart before ten a.m. and have first class fares
Show me all the flights leaving Baltimore
Show me flights arriving within thirty minutes of each other
Please list the flights from Charlotte to Long Beach arriving after lunch time
Show me the last flight to leave

为了模拟这一类的句子结构，我们可以加另外一个规则来展开 *S*：

$$S \rightarrow VP \quad \text{Show the lowest fare}$$

Yes-no 疑问式结构句子经常（尽管不是在一切情况下）用于提问（疑问式由此而得名），并且它以一个助动词开头，后面紧跟着一个主语 *NP*，再后面跟着一个 *VP*。这里是一些例子（注意，第三个例子实际上不是一个提问，而是一个命令或建议），第 19 章将讨论这种形式的句子的语用学（pragmatic）用法。

Do any of these flights have stops?
Does American's flight eighteen twenty five serve dinner?
Can you give me the same information for United?

规则如下：

$$S \rightarrow Aux\ NP\ VP$$

各种 *wh* 疑问式结构句子是在句子一级的结构中最复杂的，我们现在来研究它们。之所以称为 *wh* 疑问式，是由于其中的一个成分是 *wh* 短语，即其中包括一个 *wh* 疑问词（*who*, *where*, *what*, *which*, *how*, *why*）。*wh* 主语疑问式结构与陈述式结构相同，但其中的名词短语包括某个 *wh* 词。

What airlines fly from Burbank to Denver?
Which flights depart Burbank after noon and arrive in Denver by six p.m.?
Which flights serve breakfast?
Which of these flights have the longest layover in Nashville?

规则如下：

$$S \rightarrow Wh\text{-}NP\ VP$$

在wh非主语疑问式结构句子中，wh短语不是句子的主语，句子中还包含另一个主语。在这种类型的句子中，与yes-no疑问式结构一样，助动词出现在主语的前面。下面是一个例子：

What flights do you have from Burbank to Tacoma Washington?

下面是一个样本规则：

$$S \rightarrow Wh\text{-}NP\ Aux\ NP\ VP$$

还有其他的句子级的结构在这里没有涉及，例如前置结构（fronting）。在这种结构中，由于话语方面的原因（例如，通常是为了表示主题化或者焦点），某些短语被放到句子的开头：

On Tuesday, I'd like to fly from Detroit to Saint Petersburg

9.4　名词短语

我们可以把名词短语看成是环绕在名词中心语（head）周围形成的短语。在名词短语中，这个中心语就是中心名词（central noun）。在英语句法中，既允许名词前修饰语（prehead modifier），也允许名词后修饰语（post-head modifier）。

9.4.1　在中心名词前的成分

我们已经讨论过名词短语中的某些部分（例如限定词）以及用于表示两个名词构成的短语，即名词性成分（Nominal）的用法。我们知道，名词短语可以用限定词开头，例如：

a stop	those flights
the flights	any flights
that fare	some flights
this flight	

在某些场合下，英语中的限定词是随选的。例如，如果所修饰的名词是复数，就可以省略限定词：

Show me *flights* from San Francisco to Denver on weekdays

前面讲过，物质名词（mass noun）不要求限定词。我们知道，物质名词经常（不是在一切情况下）包括某些作为实体处理的词（例如*water*和*snow*），这些词前面不能有不定冠词*a*，不能有复数。很多抽象名词（如*music*和*homework*）也具有物质名词的这种性质。在ATIS中的物质名词包括*breakfast*，*lunch*和*dinner*：

Does this flight serve dinner?

读者可以自己用CFG语法来表示这样的事实。

在NP中，出现在限定词前面的词类称为“前限定词”（predeterminer）。前限定词通常用于表示数或量的大小。最常见的前限定词是*all*：

all the flights
all flights

在NP中，处于限定词和做中心语的名词（后限定语之间）之间的词类有多种，包括基数词（cardinal number）、序数词（ordinal number）和数量修饰语（quantifier）。

基数词的例子如下：

two friends

one stop

序数词包括*first*，*second*和*third*等，还包括*next*，*last*，*past*，*other*和*another*这样的单词：

the first one

the next day

the second leg

the last flight

the other American flight

any other fares

有些数量修饰语（*many*, (*a*) *few*, *several*）只与复数名词一起出现：

many fares

数量修饰语*much*和*a little*只与不可数名词一起出现。

形容词出现在数量修饰语之后，中心语名词之前：

a *first-class* fare

a *nonstop* flight

the *longest* layover

the *earliest* lunch flight

形容词也可以构成短语，称为形容词短语（adjective phrase）或AP。在AP中，形容词前面可以出现副词（参见第8章中关于形容词和副词的定义）：

the *least expensive* fare

我们可以把所有随选的名词前修饰语结合起来，用一个规则表示如下：

$$NP \rightarrow (Det)\ (Card)\ (Ord)\ (Quant)\ (AP)\ Nominal \tag{9.10}$$

这个简化的名词短语规则的结构扁平，因此比大多数现代语法理论中的结构表示简单得多。我们之所以可以用这样的简化方式来表示这个规则，是因为名词短语中没有那些普遍存在的内部组成关系的一致性等问题。

注意，这里我们用括号“( )”来表示随选成分（optional constituent）。一个带有一套括号的规则事实上表示了两个规则，一个规则是带括号的，另一个规则是没有括号的。

9.4.2 名词后的成分

一个中心语名词的后面可以跟随后修饰语。在英语中，常见的后修饰语有三种：

介词短语	all flights *from Cleveland*
非限定从句	ant flights *arriving after eleven a.m.*
关系从句	a flight *that serves breakfast*

在ATIS语料库中，介词短语后修饰语特别普遍，因为它们可以用来表示飞机的出发地和目的地。下面是一些例子，在其中插入括号是为了表示每个PP的边界。注意，可以把一个以上的PP捆绑在一起：

any stopovers *[for Delta seven fifty one]*
all flights *[from Cleveland] [to Newark]*
arrival *[in San Jose] [before seven p.m.]*
a reservation *[on flight six oh six] [from Tampa] [to Montreal]*

下面是一个新的*NP*规则，名词后有一至三个*PP*后修饰语：

$$Nominal \rightarrow Nominal\ PP\ (PP)\ (PP)$$

最常见的非限定后修饰语有三种：动名词（*-ing*）、*-ed*动词和不定式动词。

之所以称为动名词后修饰语，是因为这种后修饰语由一个以动名词形式（-ing）开头的动词短语组成。在这个动词短语中的动词后面，全部是介词短语，一般来说，动词后面的介词短语可以是任何东西（所谓任何东西是指它们在语义和句法上应该与动名词相容）：

any of those *[leaving on Thursday]*
any flights *[arriving after eleven a.m.]*
flights *[arriving within thirty minutes of each other]*

我们使用一个新的非终极符号*GerundVP*来定义这个规则：

$$Nominal \rightarrow Nominal\ GerundVP$$

我们可以使用前面的*VP*产生式规则，把其中的*V*替换为*GerundV*，造出成分*GerundVP*的规则：

$$\begin{aligned} GerundVP &\rightarrow GerundV\ NP \\ &\mid GerundV\ PP \\ &\mid GerundV \\ &\mid GerundV\ NP\ PP \end{aligned}$$

而*GerundV*可以定义为：

$$GerundV \rightarrow being \mid \text{preferring} \mid arriving \mid leaving \mid \ldots$$

下面的例子中的斜线部分表示另外两种最常见的非限定从句后修饰语：动词不定式和*-ed*形式。

the last flight *to arrive in Boston*
I need to have dinner *served*
Which is the aircraft *used by this flight*?

名词后关系从句（更正确地说应该是限制性关系从句）是那些通常以关系代词（以*that*和*who*最常见）开头的从句。在下面的例子中，关系代词的功能是作为嵌入动词的主语：

a flight *that serves breakfast*
flights *that leave in the morning*
the United flight *that arrives in San Jose around ten p.m.*
the one *that leaves at ten thirty five*

我们可以加入下面的规则来处理这些例子：

$$Nominal \rightarrow Nominal\ RelClause$$

$$RelClause \rightarrow (who \mid that)\ VP$$

在下面的例子中，关系代词也可以作为嵌入动词的宾语。作为练习，读者自己可以写出语法规则来处理诸如此类的更加复杂的从句。

the earliest American Airlines flight that I can get

下面的例子表明，可以把若干个名词后修饰语结合起来：

a flight *[from Phoenix to Detroit] [leaving Monday evening]*
I need a flight *[to Seattle] [leaving from Baltimore] [making a stop in Minneapolis]*
evening flights *[from Nashville to Houston] [that serve dinner]*
a friend *[living in Denver] [that would like to visit me here in Washington DC]*

9.5 并列关系

使用诸如 *and*，*or* 或 *but* 这样的连接词，可以把名词短语与其他成分单元结合起来。例如，一个并列的名词短语可以包括两个其他的名词短语，之间用连接词分开：

Please repeat [*NP* [*NP* the flight] *and* [*NP* the costs]]
I need to know [*NP* [*NP* the aircraft] *and* [*NP* flight number]]
I would like to fly from Denver stopping in [*NP* [*NP* Pittsburgh] *and* [*NP* Atlanta]]

下面是关于并列的新规则：

$$NP \rightarrow NP\ and\ NP \tag{9.13}$$

除了 *NP* 之外，大多数其他类型的短语（包括句子、*VP* 和 *PP*）也可以连接起来：

What flights do you have [*VP* [*VP* leaving Denver] *and* [*VP* arriving in San Francisco]]
[*S* [*S* I'm interested in a flight from Dallas to Washington] *and* [*S* I'm also interested in going to Baltimore]]

对于 *VP* 和 *S* 的这种并列连接，我们也可以做出相似的连接规则：

$$VP \rightarrow VP\ and\ VP \tag{9.14}$$

$$S \rightarrow S\ and\ S \tag{9.15}$$

9.6 一致关系

我们在第3章中曾经讨论过英语的屈折形态学。我们知道，英语中的现在时态有两种形式：用于第三人称单数主语的形式（例如 *the flight does*）和用于其他类型主语的形式（例如 *all the flights do*, *I do*）。第三人称单数（*3sg*）形式一般以 -s 结尾，而非第三人称单数（non-3sg）形式没有 -s 结尾。我们仍然以 *do* 为例，它的各种形式的主语的例句如下：

Do [*NP* any flights] stop in Chicago?
Do [*NP* all of these flights] often first class service?
Do [*NP* I] get dinner on this flight?

Do [*NP* you] have a flight from Boston to Forth Worth?
Does [*NP* this flight] stop in Dallas?
Does [*NP* that flight] serve dinner?
Does [*NP* Delta] fly from Atlanta to San Francisco?

下面是动词 *leave* 的一些例子：

What flights *leave* in the morning?
What flight *leaves* from Pittsburgh?

每当动词有一个名词作为它的主语时，就会发生这样的一致关系（agreement）的现象。注意，凡是主语和它的动词不一致的句子都是不合语法的句子。

*[What flight] *leave* in the morning?
*Does [*NP* you] have a flight from Boston to Forth Worth?
*Do [*NP* this flight] stop in Dallas?

怎样修改原来的语法，使它能够处理这种一致关系的现象呢？一个办法是使用多个规则的集合来扩充原来的语法。一个规则的集合表示 *3sg* 主语，另一个规则的集合表示 *Non3sg* 主语。例如，原来处理 yes-no 疑问式的规则是：

$$S \rightarrow Aux\ NP\ VP$$

我们可以用如下形式的两个规则来替代这个规则：

$$S \rightarrow 3sgAux\ 3sgNP\ VP$$
$$S \rightarrow Non3sgAux\ Non3sgNP\ VP$$

然后在词表中增加如下的规则：

$$3sgAux \rightarrow does \mid has \mid can \mid \ldots$$
$$Non3sgAux \rightarrow do \mid have \mid can \mid \ldots$$

但是，我们还需要把每个 *NP* 规则都改写为两个规则，增加规则 *3sgNP* 和 *Non3sgNP* 的规则。代词有第一人称、第二人称和第三人称之分，而全是词汇的名词短语只有第三人称，所以对于全是词汇的名词短语，只需要区分单数和复数就可以了：

$$3SgNP \rightarrow (Det)\ (Card)\ (Ord)\ (Quant)\ (AP)\ SgNominal$$
$$Non3SgNP \rightarrow (Det)\ (Card)\ (Ord)\ (Quant)\ (AP)\ PlNominal$$
$$SgNominal \rightarrow SgNoun \mid SgNoun\ SgNoun$$
$$PlNominal \rightarrow PlNoun \mid SgNoun\ PlNoun$$
$$SgNoun \rightarrow flight \mid fare \mid dollar \mid reservation \mid \ldots$$
$$PlNoun \rightarrow flights \mid fares \mid dollars \mid reservations \mid \ldots$$

如何处理第一人称和第二人称代词呢？我们将其作为练习留给读者自己去解决。

用这种办法处理“数”的一致关系所产生的问题是会增加语法的规模。每个涉及到名词或动词的规则都要有“单数”版本和“复数”版本。在处理名词或代词的格（case）时也会发生这种规则的增殖现象。例如，英语代词有主格（nominative），例如 *I*, *she*, *he*, *they*，也有宾格（accusative），如 *me*, *her*, *him*, *them*。对于这些格，我们需要给每个 *NP* 和 *N* 都增加新的规则来处理。

在德语和法语中，还会发生一些更有意思的问题，因为在德语和法语中不仅有英语那样的名词-动词一致关系问题，而且还有性的一致关系（gender agreement）问题。名词的性必须和修饰它的形容词和限定词的性一致。这将进一步增加语言的规则集合。

在第11章中，我们将介绍一种办法来处理这些一致关系问题，而不至于使语法的规模过分增大而发生爆炸。这种办法就是使用特征结构（feature structure）来把每个非终极符号参数化（parameterizing）。

9.7 动词短语和次范畴化

动词短语中包括动词和其他一些成分。在前面建立的简单规则中，这些其他成分包括NP和PP以及两者的组合。

$$VP \rightarrow Verb \quad \text{disappear}$$
$$VP \rightarrow Verb\ NP \quad \text{prefer a morning flight}$$
$$VP \rightarrow Verb\ NP\ PP \quad \text{leave Boston in the morning}$$
$$VP \rightarrow Verb\ PP \quad \text{leaving on Thursday}$$

动词短语可以比这种情况复杂得多。很多其他类型的成分，例如整个的嵌入句子，都可以跟随在动词之后，这样的成分称为句子补语（sentential complement）。

You [*VP* [*V* said [*S* there were two flights that were the cheapest]]]
You [*VP* [*V* said [*S* you had a two hundred sixty six dollar fare]]]
[*VP* [V Tell] [*NP* me] [*S* how to get from the airport in Philadelphia to downtown]]
I [*VP* [*V* think [*S* I would like to take the nine thirty flight]]

下面的一个规则可以处理这些句子：

$$VP \rightarrow Verb\ S$$

VP 中的另一个潜在的成分是另一个 *VP*。当动词是 *want*, *would like*, *try*, *intend*, *need* 等时就会发生这种情况：

I want [*VP* to fly from Milwaukee to Orlando]
Hi, I want [*VP* to arrange three flights]
Hello, I'm trying [*VP* to find a flight that goes from Pittsburgh to Denver after two p.m.]

我们在第8章中讲过，动词后面也可以跟随一个小品词，这样的小品词有些类似于介词，但它与动词组合在一起构成一个短语动词（例如，*take off*）。这些小品词一般可以看成是动词的不可分割的组成部分，而其他动词后成分不是动词的不可分割的组成成分，这时我们把短语动词处理成由两个单词组成的一个单个的动词。

动词短语可以有很多种可能的成分，但并不是每个动词和动词短语都是相容的。例如，动词 *want* 可以带 NP 补语（*I want a flight ...*），也可以带不定式 VP 补语（*I want to fly to ...*）。与此相反，动词 *find* 不能带这种类型的 *VP* 补语（**I found to fly to Dallas*）。

动词与不同类型的补语相容的这种思想很久以前就被提出来了；传统语法区分及物动词（transitive）和不及物动词（intransitive）。及物动词，如 *find*，可以带直接宾语 NP（*I found a flight*）；不及物动词，如 *disappear*，不能带直接宾语（**I disappeared a flight*）。

传统语法把动词次范畴化（subcategorize，也就是“再分类”）为两个范畴：及物动词和不及物动词，而现代语法已经把动词区分为100个次范畴。事实上，已经存在的很多标记集都有这样的次范畴化框架；对于COMLEX标记集，可参阅Macleod et al.(1998)；对于ACQUILEX标记集，可参阅Sanfilippo(1993)，我们将在第11章中进一步讨论这个问题。我们说，一个诸如*find*之类的动词按*NP*次范畴化，一个诸如*want*之类的动词或者按*NP*次范畴化，或者按不定式动词次范畴化。我们也可以把这些成分称为动词的补语（complement）（因此在上面使用了“句子补语”这个术语）。这样，*want*可以有一个*VP*补语。动词的这些可能的补语的集合称为该动词的次范畴化框架（subcategorization frame）。讨论动词和这些其他成分之间的关系的另一种办法是把动词想像成一个谓词（predicate），把这些成分想像成这个谓词的论元（argument）。所以，我们可以把这样的谓词-论元关系看成FIND (I, A FLIGHT)或WANT (I, TO FLY)。在第14章中，当讨论动词语义的谓词演算表示时，将更详细地讨论关于动词和论元的这种思想。

这里是一些次范畴化框架和动词的例子：

框架	动词	例子
$\emptyset$	eat, sleep	I want to eat
NP	prefer, find, leave,	Find [$_{NP}$ the flight from Pittsburgh to Boston]
NP NP	show, give	Show [$_{NP}$ me] [$_{NP}$ airlines with flights from Pittsburgh]
PP_{from} PP_{to}	fly, travel	I would like to fly [$_{PP}$ from Boston] [$_{PP}$ to Philadelphia]
NP PP_{with}	help, load,	Can you help [$_{NP}$ me] [$_{PP}$ with a flight]
VPto	prefer, want, need	I would prefer [$_{VPto}$ to go by United airlines]
VPbrst	can, would, might	I can [$_{VPbrst}$ go from Boston]
S	mean	Does this mean [$_{S}$ AA has a hub in Boston]?

注意，一个动词可以被次范畴化为一个特殊类型的动词短语，例如动词是不定式的动词短语（*VPto*），或者动词是光杆动词（*bare stem*）的动词短语（*VPbrst*）。还要注意，一个动词可以采用不同的次范畴化框架，例如动词find可以采用*NP NP*框架（*find me a flight*），也可以采用NP框架。

我们怎样用上下文无关语法来表示动词和它的补语之间的这些关系呢？我们能做的就是应用上下文无关语法来表示一致关系特征，这时要区分动词的各个次类（例如*Verb-with-NP-complement*，*Verb-with-Inf-VPcomplement*，*Verb-with-S-complement*和*Verb-with-NP-plus-PP-complement*等）：

$$\begin{aligned}
\textit{Verb-with-NP-complement} &\rightarrow \textit{find} \mid \textit{leave} \mid \textit{repeat} \mid \ldots \\
\textit{Verb-with-S-complement} &\rightarrow \textit{think} \mid \textit{believe} \mid \textit{say} \mid \ldots \\
\textit{Verb-with-Inf-VP-complement} &\rightarrow \textit{want} \mid \textit{try} \mid \textit{need} \mid \ldots
\end{aligned}$$

这样，我们的*VP*规则就可以修改为要求某个动词次类：

$$\begin{aligned}
\textit{VP} &\rightarrow \textit{Verb-with-no-complement} \quad \text{disappear} \\
\textit{VP} &\rightarrow \textit{Verb-with-NP-comp NP} \quad \text{prefer a morning flight} \\
\textit{VP} &\rightarrow \textit{Verb-with-S-comp S} \quad \text{said there were two flights}
\end{aligned}$$

使用这样的方法，也会产生与解决一致关系特征问题时一样的问题，会引起规则数目的爆炸。解决这两个问题的标准办法是使用特征结构（feature structure）。我们将在第11章中介绍特征结构。在第11章中，我们还将像本章中讨论动词这样，讨论名词、形容词和介词的次范畴化补语问题。

9.8 助动词

动词有一个次类称为助动词（auxiliary）或者称为辅助动词（helping verb）。助动词具有特殊的句法约束，这种句法约束也可以看成是一种次范畴化。助动词包括情态动词（*can*, *could*, *may*, *might*, *must*, *will*, *would*, *shall*, *should*）、完成式助动词（*have*）、进行式助动词（*be*）和被动式助动词（be）。每个助动词都给它后面的动词形式一个约束，而且它们之间要按照一定的顺序进行结合。

情态动词给*VP*次范畴化时，*VP*的中心动词是光杆动词，例如*can go in the morning*, *will try to find a flight*。完成式助动词*have*给*VP*次范畴化时，*VP*的中心动词要用过去分词形式，例如*have booked 3 flights*。进行式助动词*be*给*VP*次范畴化时，*VP*的中心动词要用现在分词形式（动名词形式），例如*am going from Atlanta*。被动式助动词*be*给*VP*次范畴化时，*VP*的中心动词要用过去分词形式，例如*was delayed by inclement weather*。

一个句子可以有多个助动词，但这些助动词出现时要按特定的顺序：

情态助动词 < 完成式助动词 < 进行式助动词 < 被动式助动词

下面是多个助动词的例子：

情态 + 完成	could have been a contender
情态 + 被动	will be married
完成 + 进行	have been feasting
情态 + 完成 + 被动	might have been prevented

助动词常常被当成诸如*want*，*seem*或*intend*这样的动词来处理，它们在次范畴化时要求特定的*VP*补语。这样，*can*在词表中可以当成*verb-with-bare-stem-VP-complement*这一类。关于助动词之间的顺序约束，通常使用Halliday(1985)的系统语法（systemic grammar）来处理。系统语法引入了一种称为动词组合（verb group）的特殊成分，它的次成分包括了所有的助动词以及主要动词。某些顺序约束也可以通过其他方法来处理。例如，由于情态助动词不具备进行式或分词形式，它们从来不允许跟随在进行式或被动式*be*以及完成式*have*后面。建议读者自己为助动词写一套语法规则。

被动式结构具有一些特性，使得它处理起来比其他的助动词更困难。主要的困难是语义方面的。非被动句（主动句）的主语通常是事件在语义上施事（*I prevented a catastrophe*），而被动句的主语通常是事件的受事（*a catastrophe was prevented*）。我们将在第15章中进一步讨论有关内容。

9.9 口语的句法

英语书面语的语法和会话口语的语法有很多共同特征，但在很多方面也不尽相同。本节将简捷地介绍英语口语句法的一些特征。

我们一般使用话段（utterance）而不使用句子（sentence）作为口语的单位。图9.5是ATIS的口语话段的一个样本，它展示了口语语法的一些特点。

这里使用的是一种在语音识别中用于转写口语语料库的标准风格的转写方式。逗号（,）表示短暂的停顿，点号（.）表示长停顿，方括号（[uh]）表示非话语事件（换气声、咂嘴声、“啊”声*uh*和“嗯”声*um*）。

这些话段在很多方面与书面的英语句子不同。其中一个不同点在词汇统计方面，例如英语口语中的代词比书面语中的代词频度高得多，口语中的主语几乎全都是代词。另一个不同点是口语中常

出现各种各样的不流畅现象（犹豫、修正、重说等），我们将在下面讨论这个问题。口语句子中还经常出现一些书面语中不常见的片断或短语（*one way* 或 *around four p.m.*）。

the . [exhale] . . . [inhale] . . [uh] does American airlines . offer any . one way flights . [uh] one way fares, for one hundred and sixty one dollars
[mm] i'd like to leave i guess between [um] . [smack] . five o'clock no, five o'clock and [uh], seven o'clock . P M
around, four, P M
all right, [throat_clear] . . i'd like to know the . give me the flight . times . in the morning . for September twentieth . nineteen ninety one
[uh] one way
[uh] seven fifteen, please
on United airlines . . give me, the . . time . . from New York . [smack] . to Boise-, to . I'm sorry . on United airlines . [uh] give me the flight, numbers, the flight times from . [uh] Boston . to Dallas

图 9.5　用户与 ATIS 系统交互时的口语话段的样本

最后，口语中的句子还带有特定的韵律（prosody）。话段的韵律包括它的特定的音高曲拱（声波基频的升降），它的重音模式或节奏（构成一个句子的重读音节和非重读音节）以及其他诸如语流速度（语速）之类的因素。

9.9.1　不流畅现象

区别口语和书面语的最突出的句法特征是口语的不流畅现象（disfluency）。这种不流畅现象包括使用“啊”声（*uh*）和“嗯”声（*um*），单词重复以及“假开始”（false start）。图9.6所示的ATIS语料库中的句子是使用假开始和“嗯”声的一个例子。当说话人开始问 *one-way flights* 时，他停了下来，自己改正这个提问，然后重新开始，并且问 *one-way fares*，这就是“假开始”。

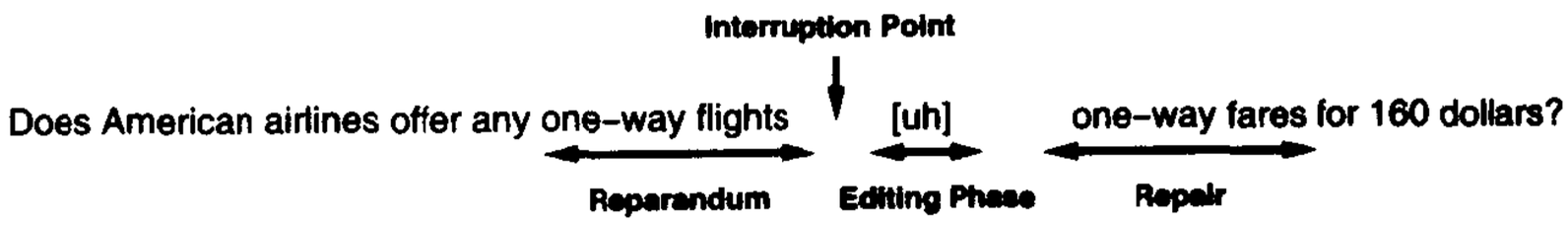


图 9.6　不流畅现象的一个例子(Shriberg, 1994)

在图9.6中，*one-way flights* 这个片断称为“待修正片断”（reparandum），用来替换的序列称为“修正片断”（repair）[这些术语来自 Levelt(1983)]。说话者打断原来的单词序列的地方称为“停顿点”（interruption point），它正好处于单词 *flights* 之后。

单词 *uh* 和 *um* 有时称为有声停顿（filled pause），在词表中可以当成正规的单词那样来处理。在语音识别系统中也经常这样处理它们。在语音识别的HMM发音词典中经常有这些单词的发音模型，在语音识别系统里使用的 *N* 元语法模型中也包括这些单词与其他单词共同出现时的概率。

在语音理解中，我们的目的是弄清楚输入句子的意思，如果能够把这样的“重新开始”探查出来，从而对那些说话者认为可能“正确”的单词进行编辑（例如，在上面的句子中，如果能够探查出重新开始的地方），就能够删除那个“待修改片断”（one-way flights uh），而只对句子的其余部分进行剖析：

Does American airlines offer any one-way flights uh one-way fares for 160 dollars?

这种不流畅现象与句子中的成分结构有什么关系呢？Hindle(1983)指出，要修改的片断通常都与停顿点前面的成分具有同样的结构。因此，在上面的例子中，要修改的片断是一个NP，它就是那个“待修改片断”（reparandum）。这意味着，如果能够自动找出停顿点，通常也就能够自动探查出待修改片断的边界。

9.10 语法等价与范式

形式语言被定义为单词的符号串的集合（这个集合可能是无限的）。这意味着，如果我们要问两个语法是否等价，就可以问这两个语法生成的符号串的集合是否相同。事实上，两个不同的上下文无关语法可能生成同样的语言。

一般来说，我们要区分两种不同的语法等价，一种是强等价（strong equivalence），另一种是弱等价（weak equivalence）。如果两个语法生成相同的符号串集合，而且它们对于每个句子都指派同样的短语结构（只允许改变非终极符号的名字），我们就可以说这两个语法是强等价的。如果两个语法生成相同的符号串集合，但是不给每个句子指派同样的短语结构，我们就可以说这两个语法是弱等价的。

语法都使用一个范式（normal form），在范式中每个产生式都使用一个特定的形式，有时这样非常有用。例如，如果一个上下文无关语法是 ε-自由（ε-free）的，并且如果它的每个产生式的形式为 $A \rightarrow B\ C$ 或 $A \rightarrow a$，也就是说，每个规则的右部或者是两个非终极符号，或者是一个终极符号，那么这个上下文无关语法就是Chomsky范式的（Chomsky normal form，简称CNF）(Chomsky, 1963)。凡是Chomsky范式的语法都具有二叉树（binary tree）形式，这对于某些算法是很有用的。

任何上下文无关语法都可以转变成一个弱等价的Chomsky范式语法。例如，形式为：

$$A \rightarrow B\ C\ D$$

的规则，可以转变成如下两个CNF规则：

$$A \rightarrow B\ X$$

$$X \rightarrow C\ D$$

作为练习，读者可以自己写一个算法，把任意的上下文无关语法转变为Chomsky范式。

9.11 有限状态语法和上下文无关语法

我们在9.1节中讲过，复杂的语法模型必须表示组成性。这是语法的有限状态模型为什么总是不适合的原因之一。现在我们已经研究了名词短语的句法的某些细节，就可以讨论有限状态语法的其他问题了。这个问题就是递归（recursion）。当一个非终极符号的展开式中也包含了这个非终极符号时，就会产生语法的递归问题。例如，在规则 *Nominal* → *Nominal PP* 中，就有递归问题。

为了理解为什么递归是有限状态语法的一个问题，我们首先为前面见过的语法规则建立有限状态模型。例如，我们可以用正则表达式(=FSA)以 *Nominal* 为中心来给名词短语建立如下的模型：

(Det) (Card) (Ord) (Quant) (AP) Nominal

如何来处理后修饰语呢？我们只加一个 *PP*，然后可以把正则表达式扩充如下：

*(Det) (Card) (Ord) (Quant) (AP) Nominal (PP)**

这样，为了完成这个正则表达式，只需要按顺序展开 PP，结果如下：

*(Det) (Card) (Ord) (Quant) (AP) Nominal (P NP)**

但是，当定义 *NP* 时，先要假定存在 *NP*！这样一来，势必要这样来扩充规则：

*(Det) (Card) (Ord) (Quant) (AP) Nominal (P (Det) (Card) (Ord) (Quant) (AP) Nominal (P NP))**

显而易见，这时 *NP* 又要卷土重来了！问题在于 *NP* 是一个递归规则（recursive rule）。在有限状态自动机中，事实上就要按这种鬼鬼祟祟的方式翻来覆去地使用这个特殊的右递归规则（right-recursive rule）。一般来说，有限自动机是不能处理递归的，而递归在整个 *NP* 模型中是非常普遍的现象（例如，在展开 *RelClause* 和 *GerundVP* 时都会遇到 *NP* 的问题）。

*(Det) (Card) (Ord) (Quant) (AP) Nominal (RelClause|GerundVP|PP)**

Chomsky(1959a)特别地证明了一个上下文无关语言L能够被有限自动机生成，当且仅当存在一个生成语言L的、没有任何中心－自嵌入（center-embedded）递归的上下文无关语法（形式为 $A \rightarrow \alpha A \beta$）。

至此我们至少可以看出，不能用有限状态语法来模拟英语的全部句法，但是仍然可以建立FSA来近似地模拟英语（例如，只展开一定数量的 *NP*）。实际上，有的算法能够自动地生成近似于上下文无关语法的有限状态语法(Pereira and Wright, 1997)。

在第10章中，我们将讨论有限状态自动机的一个扩充版本，称为递归转移网络（recursive transition network，简称RTN），它给FSA增加了很强的递归能力。由RTN形成的自动机恰好与上下文无关语法同构（isomorphic）。在一定的场合下，它可以作为研究 CFG 的一个有用的比喻。

9.12　语法和人的语言处理

人们在心理上处理语言的时候是不是也使用上下文无关语法呢？已经证明，要找到对于这个问题的确凿证据是非常困难的。例如，某些早期的实验要求受试人判断在句子中哪些单词更紧密地集成在一起(Levelt, 1970)，结果发现，受试人在直觉上划分出的单词的组合与句法成分是对应的。其他的一些实验检查了句法成分在听觉理解中的作用，让受试人听句子的同时也听在不同时刻发出的短促的“喀嚓”声。Fodor and Bever(1965)发现，当“喀嚓”声出现在句子成分的边界时，受试人常常会听错。他们主张，由此可以认为句法成分是一个“感知单位”（perceptual unit），这个感知单位具有“抵抗干扰”的能力。遗憾的是，这种采用“喀嚓”声的实验在方法论上存在着严重的问题，参阅 Clark H. and Clark E.(1977)。

这些早期的研究的一个普遍问题是，没有认识到这些成分是句法单位的同时也是语义单位。因此，正如我们将在第 15 章中讨论的，*a single odd block* 是一个成分（*NP*），但它也是一个语义单位，表示“具有某种特征的 BLOCK 类型的客体”。人们发现，成分边界的实验可以简单测定的是语义的事实而不是语法的事实。

因此，需要找到证据来说明那些不是一个语义单位的成分。另外，由于存在着很多不是基于成分而是基于词汇依存关系的语法理论，因此重要的是找到不能作为一个词汇事实来解释的证据，也就是要证明，组成性（constituency）不是建立在特定单词的基础上的。

Kathryn Bock 和她的同事们的一系列有启发性的实验支持了组成性的观点。例如，Bock and Loebell(1990)避开了所有这些早期的实验的缺点。早期的实验只研究受试人是否使用某一特定的句法成分（例如，像 *V NP PP* 这样的特定成分），而 Bock 则研究受试人是否更倾向于使用下面一个句子的成分。换言之，她们向受试人提的问题是，成分的使用是否优先于（prime）该成分对于下

一个句子的使用。在前面的章节中可以看到，优先性是检验心理结构存在的一个常见方法。Bock和Loebell依靠的是对英语双及物关系的不同解释（ditransitive alternation）。双及物动词是像*give*这样的可以具有两个论元的动词：

The wealthy widow gave [*NP* the church] [*NP* her Mercedes]. （9.16）

（有钱的寡妇把她的 Mercedes 汽车给了教堂）

*give*这个动词还有另外一个可能的次范畴化框架，称为介词给格（prepositional dative），其中的间接宾语用介词表示：

The wealthy widow gave [*NP* her Mercedes] [*PP* to the church]. （9.17）

我们在讨论动词的次范畴化时曾经指出，除了*give*之外还有很多动词也有这样的不同解释，如*send*和*sell*等，参阅Levin(1993)对各种不同解释的模式进行的总结。Bock和Loebell根据这些不同解释来进行试验，她们给受试人一张图画，然后要求受试人用一个句子描述出图画的内容。设计图画时，要考虑到如何向受试人显示一个事件（例如，一个男孩把一个苹果给老师），使受试人能够抽取出*give*或*sell*之类的动词。因为这些动词都是具有不同解释的双及物动词，受试人可能选择*The boy gave the apple to the teacher*，也可能选择*The boy gave the teacher an apple*。

在描述图画之前，要求受试人大声地朗读一个无关的“优先”句子；这个优先句子的结构可以是*V NP NP*或者*VP NP PP*。更为关键的是，这些优先句子尽管具有相同的成分结构（例如，对于给格可以具有不同解释的句子），但是它们并不具有相同的语义。例如，优先的句子可能是介词方位格，而不是介词给格：

IBM moved [*NP* a bigger computer] [*PP* to the Sears store]. （9.18）

Bock和Loebell发现，如果受试人刚刚读过*V NP PP*的句子，就会更倾向于使用*V NP PP*结构的句子来描述图画。这意味着，对于特定成分的使用会使后来对于该成分的使用变得优先，因此我们可以说，成分必定要在心理上表示出来，以便确定它是优先的还是非优先的。

在最近的研究中，Bock和她的同事们试图继续发现对于这种组成性结构的证据。

关于人们是否使用上下文无关语法的问题，存在着各种不同的观点。很多研究者认为，自然语言不同于形式语言，特别是在自然语言中存在着一些不能用单纯的句法上下文无关语法的产生式来描述的句子。他们主张，完整的句法模型将被证明是不可能的，除非这个句法模型把其他领域的知识也包括进来（例如，语义、语调、语用和社会/交互等领域的知识）。但是，另外一些研究者主张，自然语言的句法可以用形式语言来表示。主张后面这种观点的人被称为模块主义者（modularist）；坚持这种立场的研究者主张，人的句法知识是人的心理上的一个特别的模块。主张前面的观点的人被称为反模块主义者（anti-modularist），他们认为，语法知识中可以引入语义、语用和其他约束。在第12章中，我们再来研究这样的争论。

9.13 小结

本章通过上下文无关语法，介绍了句法的一些基本概念，具体内容如下：

- 在很多语言中，前后相续的单词的组合功能就像一个组或成分，我们可以用上下文无关语法（又称为短语结构语法）来为它们建立模型。

- 一个上下文无关语法由一套规则或产生式组成，这些规则在非终极符号的集合和终极符号的集合上表示。从形式上讲，一个特定的上下文无关语言是从一个特定的上下文无关语法推导出来的符号串的集合。
- 生成语法是形式语言的语言学中的一个传统名称，用于给自然语言的语法建立模型。
- 在英语中，句子一级有很多语法结构，其中陈述式结构、命令式结构、yes-no 疑问式结构、*wh* 疑问式结构是四种最常见的类型，它们可以用上下文无关规则来建模。
- 在英语名词中，中心语名词前面的修饰成分有限定词、数词、数量修饰语和形容词短语，中心语名词后面可以跟随后修饰成分，经常出现的后修饰成分可能有：动名词 VP、不定式 VP 和过去分词 VP 等。
- 英语中的主语与主要动词在人称和数方面保持一致关系。
- 动词可以根据它所期望的补语的类型来进行次范畴化。简单的次范畴是及物动词和不及物动词，大多数语法还包括除此之外的更多范畴。
- 在口语中，句子的连接一般称为话段。话段可能是不流畅的，包括如 *uh* 和 *um* 这样的有声停顿、再开始、修正等。
- 任何上下文无关语法都可以转变为 Chomsky 范式。在 Chomsky 范式中，每个规则的右手边或者是两个非终极符号，或者是一个终极符号。
- 上下文无关语法比有限状态语法更强，但是有限状态语法可以使用 FSA 来近似地表示上下文无关语法。
- 有一些证据表明，在人的语言处理中，组成性在起作用。

9.14　文献和历史说明

> "den sprachlichen Ausdruck für die willkürliche Gliederung einer Gesammtvorstellung in ihre in logische Beziehung zueinander gesetzten Bestandteile" ①
>
> "the linguistic expression for the arbitrary division of a total idea into its constituent parts placed in logical relations to one another"
>
> Wundt's (1900:240) definition of the sentence; the origin of the idea of phrasal consitituency, cited in Percival (1976).

Percival(1976)的最新研究阐明，把句子分割为成分层次的思想最早出现于实验心理学的奠基人 Wundt 的《大众心理学》(Völkerpsychologie, 1900)一书中。与此相反，从古典时期开始的传统的欧洲语法研究如何确定单词与单词之间的关系，而不是研究如何确定成分与成分之间的关系。Wundt 关于组成性的思想被 Leonard Bloomfield 在他的早期著作 An Introduction to the Study of Language（语言研究导论）(Bloomfield, 1914)中引入了语言学。后来，当他的著作 Language（语言论）(Bloomfield, 1933)发表时，直接成分分析法（immediate-constituent analysis）成为了美国语言学研究中的相当完善的方法。与此相反，欧洲的句法学家们仍然强调以词为基础的语法或者依存语法（dependency grammar）。我们在第 12 章中介绍依存语法的时候，还将讨论这方面的问题。

美国结构主义提出了关于直接成分的一些定义，把他们的研究说成是“发现程序”（discovery procedure）；这是描写语言句法的一种有方法论色彩的算法。总的来说，这些研究都试图印证“直

① 原文是 Wundt 著作中的德文，我们按德文照录，并给出了中文译文：
句子是把完整的思想任意分为它的组成成分并把它们置于逻辑关系中的语言表示。
Wundt 给句子下的定义(1900:240)，这是关于短语组成性的思想源头，引自 Percival(1976)

接成分的首要标准就是一个组合作为简单的单位起作用的程度”这样的直觉(Bazell, 1952, p.284)。其中最有名的定义是 Harris 关于使用“可替换性”（substitutability）试验来检验单独的单位分布相似性（distributional similarity）的思想。从实质上看，这种方法是把一个结构分解为若干个成分，把它替换为可能成分的简单结构；如果可以用一个简单形式（例如，*man*）来替换一个比较复杂的结构（例如，*intense young man*），这个比较复杂的结构 *intense young man* 就可能是一个成分。Harris 的试验成为了把成分看成是一种等价类的这种直觉的开端。

这种层次成分思想的最早的形式化描述是Chomsky(1956)定义的短语结构语法(phrase-structure grammar），后来在 Chomsky(1957)和 Chomsky(1975)中又做了进一步的扩充（并提出反对的理由来论证）。从此以后，大多数的生成语法理论都建立在上下文无关语法的基础上，至少也是部分地建立在上下文无关语法的基础上。例如，中心语驱动的短语结构语法（Head-Driven Phrase Structure Grammar）(Pollard and Sag, 1994)，词汇功能语法（Lexical-Functional Grammar）(Bresnan, 1982)，管辖与约束理论（Government and Binding）(Chomsky, 1981)和构造语法（Construction Grammar）(Kay and Fillmore, 1999)等，其中很多理论使用了称为“X 阶标图式”（X-bar schemata）的图式上下文无关模板。

就在Chomsky的开创性研究不久之后，上下文无关语法再次被Backus(1959)和Naur et al.(1960)在描述ALGOL程序语言的工作中独立地发现。Backus(1996)受到了Emil Post的产生式思想的影响，而 Naur 的工作和 Backus 的工作是彼此独立的（回忆一下前面对于科学中多重发现的讨论）。在这些早期的研究工作之后，因为在这个时期研制了很多上下文无关语法的高效剖析算法，所以自然语言处理的大多数计算模型都是建立在上下文无关语法的基础上的（见第 10 章）。

我们已经说过，建立在上下文无关规则基础上的语法并不是无所不在的。这种形式化方法的一个扩充是树邻接语法（Tree Adjoining Grammar，简称 TAG）(Joshi, 1985)。树邻接语法的基本数据结构是树，而不是规则。有两种树，一种是初始树（initial tree），一种是附加树（auxiliary tree）。初始树用于表示简单的句子结构，附加树用于在树中添加递归。树通过两种运算结合起来，一种运算称为替换（substitution），一种运算称为邻接（adjunction）。详细情况可参阅 Joshi(1985)。树邻接语法的一种扩充称为词汇化的树邻接语法（Lexicalized Tree Adjoining Grammar），我们将在第 12 章中讨论这种语法。

另一种语法理论不是建立在上下文无关语法的基础上的。这种语法理论以单词之间的关系为基础，而不以成分之间的关系为基础。这些语法理论中最著名的是各种依存语法（dependency grammar），有代表性的依存语法有 Mel'čuk(1979)的依存语法、Hudson(1984) 的词语法（Word Grammar）和Karlsson(1995)的约束语法（Constraint Grammar）。基于依存的语法最近又在现代统计剖析中流行起来，因为在这个领域中开始认识到单词与单词之间的关系的关键性作用。我们在第12章中将进一步讨论这方面的问题。

对英语语法的通用文献感兴趣的读者选择阅读Quirk et al.(1985)应该不会浪费时间，而会感到获益匪浅。另一个有用的文献是 McCawley(1998)。

关于句法的优秀入门教材已经有许多了。Sag and Wasow(1999)是对形式句法（formal syntax）的介绍，重点讨论中心语驱动的短语结构语法中对短语结构、合一以及类型层级的使用。Van Valin (1999)从一个轻视形式而更注重功能的角度，重点介绍了句法结构的跨语言信息和功能动机。

第10章　基于上下文无关语法的剖析

There are and can exist but two ways of investigating and discovering truth. The one hurries on rapidly from the senses and particulars to the most general axioms, and from them... derives and discovers the intermediate axioms. The other constructs its axioms from the senses and particulars, by ascending continually and gradually, till it finally arrives at the most general axioms.

Francis Bacon, *Novum Organum* Book I.19 (1620)[①]

17世纪西方的哲学传统开始区分关于人类使用和掌握知识的两种重要看法。在英国以Bacon和Locke为代表的经验主义传统（empiricist tradition）主张，知识产生的途径就是根据外界的数据和经验来进行归纳和推理的过程。而在欧洲大陆以Descartes为代表的理性主义传统可以追溯到Plato's Meno，他们主张学习和推理的途径是由先验的知识和与生俱来的思想所指导的。

经验主义和理性主义的论战一直持续到今天，并且对于剖析（parsing）算法的描述起着重要的作用。在第3章中，我们把剖析定义为识别一个输入符号串并给它指派一个结构的过程。因此，句法剖析就是识别一个句子并给句子指派一个结构的过程。本章着重介绍由第9章的上下文无关语法指派的结构。因为上下文无关语法是一种陈述式的形式化方法，这种语法不具体地说明如何从一个句子计算出它的剖析树，所以本章将具体说明如何给一个输入句子自动指派一个上下文无关树（短语结构树）的各种可能的算法。

剖析树可以直接应用于词语处理系统中。如果一个句子不能被剖析，它就很可能有语法错误（或者至少阅读起来有困难）。此外，剖析是语义分析表示的一个重要的中间阶段（我们将在第15章中讨论这个问题），因此剖析在机器翻译（machine translation）、问答系统（question answering）和信息抽取（information extraction）的应用中都起着重要的作用。例如，要回答下面的问题：

What books were written by British women authors before 1800?

如果我们知道这个句子的主语是*what books*，*by*-附接语是*British women authors*，就有助于我们了解到用户需要的是图书的列表（而不仅仅是作者的列表）。句法剖析器还可以应用于词典编纂中，建立联机的词典。最后，随机的剖析算法近来还开始应用于语音识别系统中，既应用于语言模型(Ney, 1991)，也应用于非有限状态的声学和音位结构学模型(Lari and Young, 1991)。

本章讲述的主要的剖析算法是Earley算法（Eaerley algorithm），这是一种基于动态规划（dynamic programming）的上下文无关剖析算法。我们在前面已经介绍了一些动态规划算法：最小编辑距离算法、Viterbi算法和向前算法。其中，Earley算法是最常用的动态规划算法之一，另外两个动态规划算法是Cocke-Younger-Kasami算法（简称CYK算法，将在第12章中介绍）和Graham-Harrison-Ruzzo算法（简称GHR算法）(Graham et al., 1980)。在介绍Earley算法之前，我们先介绍研制算法需要的一些基本剖析概念。首先回顾一下，在第2章中介绍有限自动机时把剖析和识别看成“搜索”

---

① 调查并发现真理的方法有两种。一种方法是匆忙地从感觉和特殊事物到最普遍的公理，然后从这些公理出发……推导并发现中间的公理。另一种方法是从感觉和特殊事物来构造它们的公理，不断地逐渐地上升，最后达到最普遍的公理。
——培根，《新工具》

的比喻，讲解了自顶向下（top-down）和自底向上（bottom-up）的搜索策略。然后，我们介绍基本的自顶向下的回溯剖析算法，介绍一些简单而有效的剖析。尽管这些算法本身的表述非常清楚，效率也相对比较高，但是它们不能有效地处理歧义（ambiguity）。所谓歧义，就是一个句子或者词组出现一个以上的剖析结果，这是剖析中一个至关重要的问题。本章的最后一节将说明Earley算法怎样在自顶向下的剖析过程中借助于自底向上的过滤方法来有效地处理歧义输入。

10.1 剖析就是搜索

在第2章和第3章中，我们曾经说明，通过有限状态自动机发现正确的路径，或者发现对于输入的正确的转录，可以看成是一个搜索问题。例如，对于FSA来说，剖析就是在自动机中搜索一切可能的路径空间。在句法剖析中，剖析可以看成是对于一个句子搜索一切可能的剖析树空间并发现正确的剖析树。搜索一切可能的路径空间可以用FSA的结构来定义，所以，搜索一切可能的剖析树空间也可以用语法来定义。例如，我们来考虑如下的ATIS中的句子：

Book that flight. (10.1)

可以使用图10.2中的微型语法和词表，其中包括了在第9章中介绍的英语CFG规则。指派给这个句子的正确的剖析树如图10.1所示。

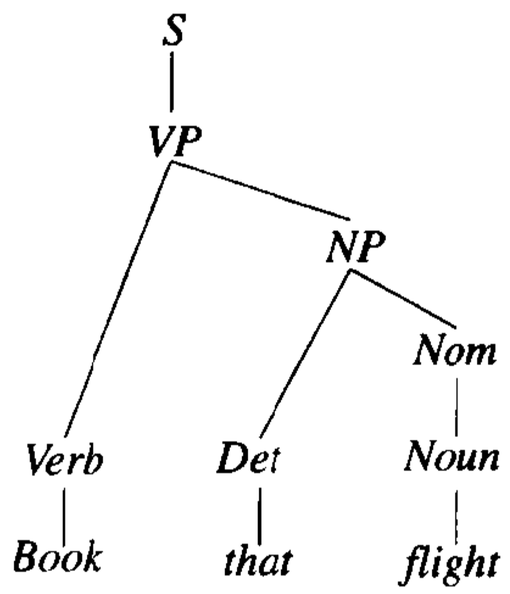


图10.1 根据图10.2中的语法得出的句子 *Book that flight* 的正确的剖析树

怎样使用图10.2中的语法来给例句（10.1）指派图10.1中的剖析树呢？在这种情况下，只存在一个剖析树，但是也可能存在一个以上的剖析树。剖析搜索的目标是发现以初始符号 *S* 为根并且恰好覆盖整个输入符号串的一切剖析树。无论选择什么搜索算法，这里明显地存在着两种约束有助于指导这种搜索。第一种约束来自数据，即输入句子本身。如果最后的剖析树是正确的，它必须有三个叶子，而且这三个叶子应该分别是 *book*，*that* 和 *flight*。第二种约束来自语法。如果最后的剖析树是正确的，它必须有一个根，这个根就是初始符号 *S*。

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	
$Nominal \rightarrow Noun\ Nominal$	$Prep \rightarrow from \mid to \mid on$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	$Nominal \rightarrow Nominal\ PP$

图10.2 英语微型语法和词表

回顾在本章开始时描述的关于经验主义和理性主义的论战，我们可以认识到，这两种约束同时也就产生了大多数剖析算法使用的两种搜索策略：一种是自顶向下或目标制导的搜索（goal-directed search），一种是自底向上或数据制导的搜索（data-directed search）。

10.1.1　自顶向下剖析

自顶向下的剖析在搜索时试图从根结点S到叶子构造剖析树。让我们来考虑自顶向下剖析的搜索空间，现在假定剖析要并行地构造出所有可能的树。算法开始时假定，给初始符号指派S，输入就可以从S开始被推导出来。下一步要搜索所有能够以S为顶点的树，查找在语法的所有规则中左手边为S的规则。在图 10.2 的语法中，有三条规则可以展开S，所以，在图 10.3 的搜索空间第二层中，我们造出了三个局部树。

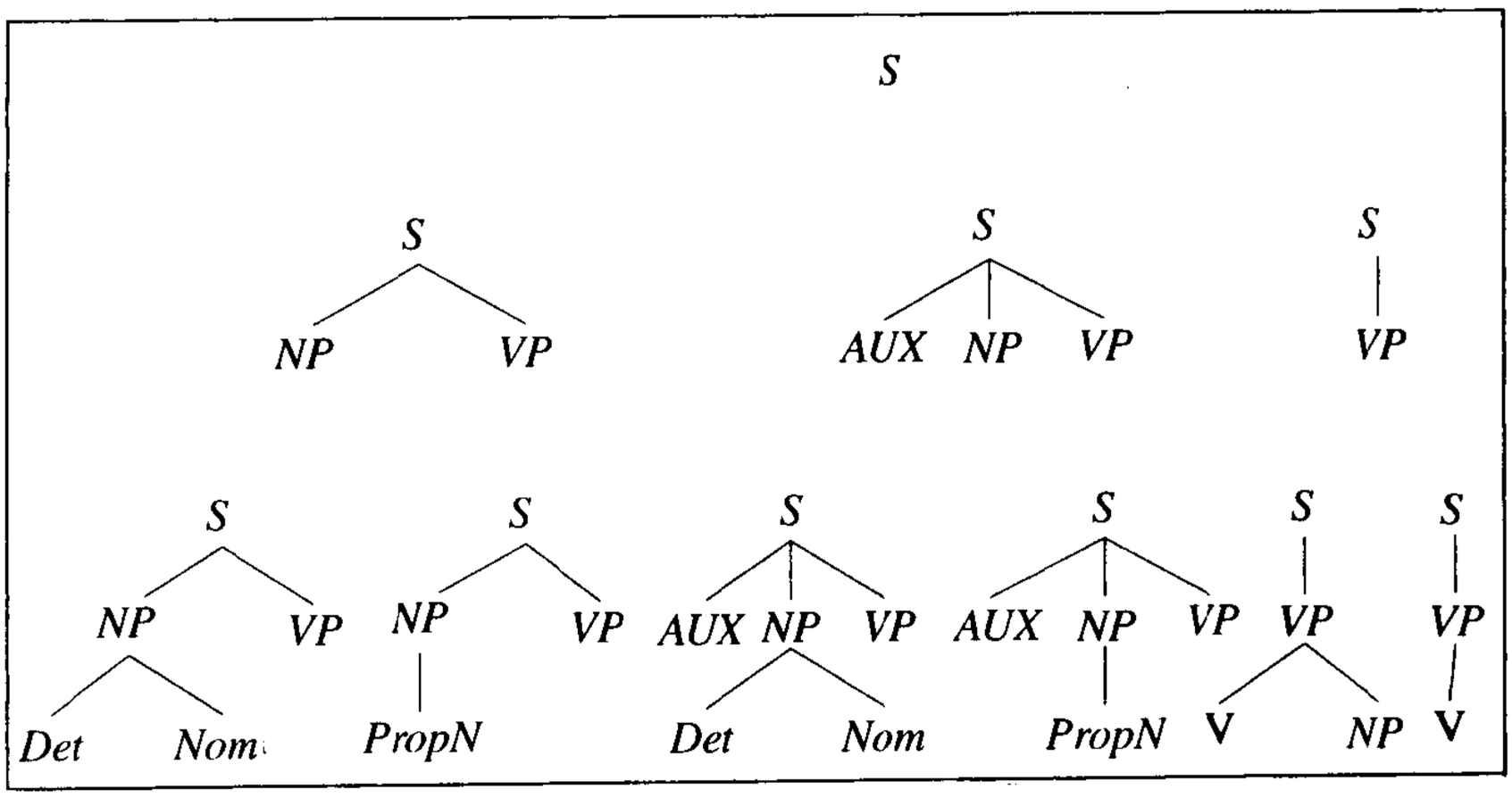


图 10.3　自顶向下搜索空间的展开。每一层的树都是由上一层的树构造而成的，用可能的展开来代替最左的非终极符号，把这些树集中起来构成一个新的层

然后，我们来展开这三个新的树中的成分，展开方法与展开S时所用的一样。第一个树告诉我们，可以把S展开为NP后面跟着一个VP；第二个树告诉我们，可以把S展开为Aux后面跟着一个NP和一个VP；第三个树告诉我们，可以把S展开为一个VP。为了便于在书上表示搜索空间，我们在图 10.3 的第三层中只画出了每个树的可展开的最左叶子的展开结果。在搜索空间的每一层，我们使用规则的右手边来提供剖析所期待的新的集合，使用它们来递归地生成树的其余部分。这样，树不断地向下生长，一直生长到树底部的词类范畴。当生长到这种程度时，叶子不能完全与输入中的单词匹配的那些树就会被拒绝，最后留下的树就代表成功的剖析结果。

在图 10.3 的第三层中，只有第五个剖析树（由规则$VP \rightarrow Verb\ NP$展开的树）最后与输入句子*Book that flight*相匹配。请读者自己根据图 10.1 来检验这个结果。

10.1.2　自底向上剖析

自底向上（bottom-up）剖析是最早提出的剖析算法［由 Yngve(1955)最早提出］，这种算法在计算机语言的移进－归约剖析中得到了普遍使用(Aho and Ullman, 1972)。在自底向上剖析中，剖析从输入的单词开始，每次都使用语法中的规则，试图从底部的单词向上构造剖析树。如果剖析器成功地构造了以初始符号S为根的树，而且这个树覆盖了整个输入，那么剖析就获得了成功。图 10.4 说明了从句子*Book that flight*开始的自底向上剖析的搜索空间。剖析器开始时在词表中查找每个单

词（*book, that, flight*），并且用每个单词的词类构造三个局部树。不过，单词*book*是有歧义的，它可能是名词，也可能是动词。这样，剖析器就必须考虑两个可能的树的集合。图10.4中的前两层说明了搜索空间刚开始时分叉的情况。

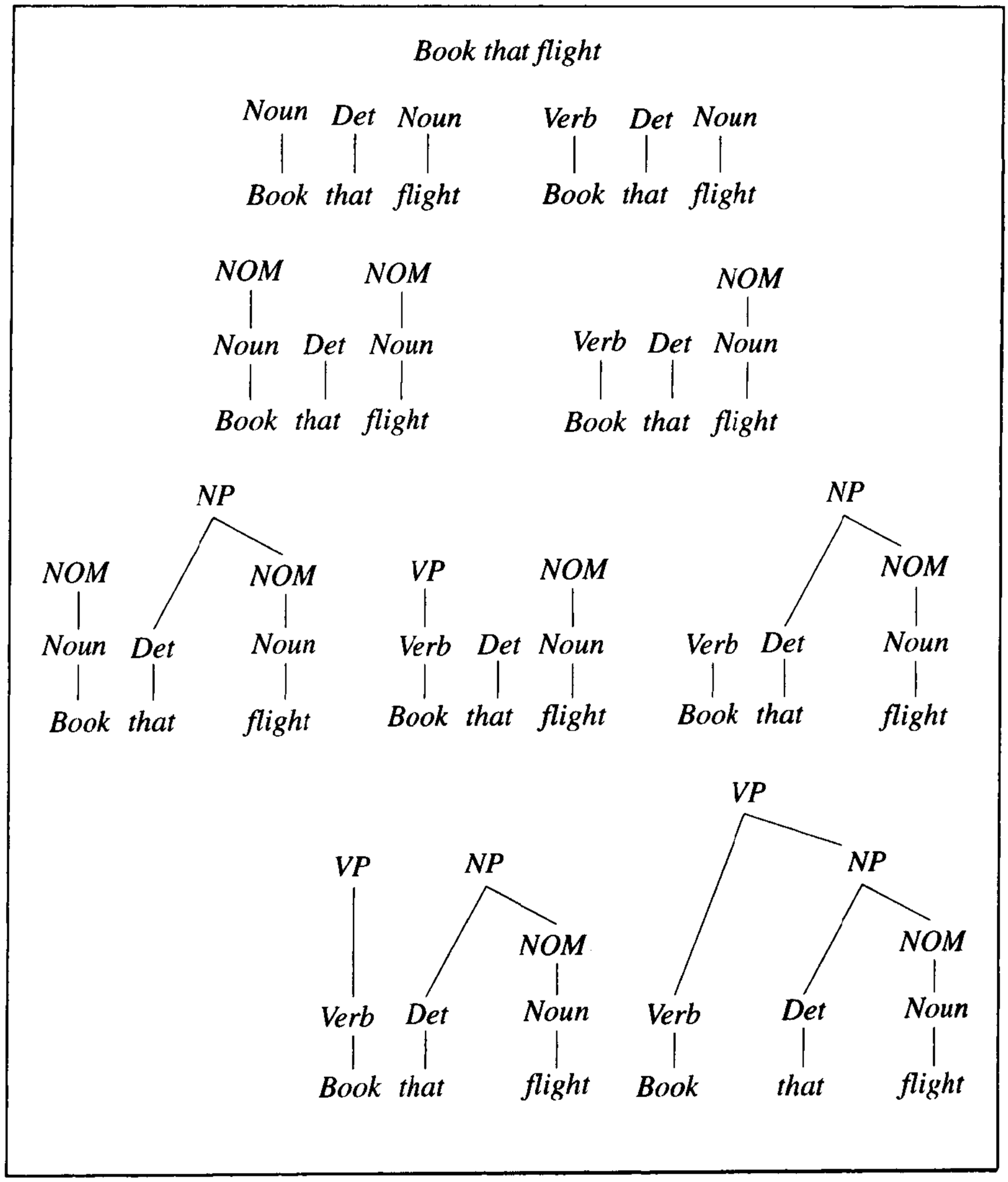


图 10.4 句子*Book that flight*自底向上搜索空间的展开。这个图没有表示出正确的剖析树的搜索的最后一层（见图 10.1）。从这个图中的搜索空间可以理解最后的剖析树是怎样构造出来的

然后展开第二层中的每个树。在剖析左侧的树时（其中一个*book*被错误地考虑为名词），规则*Nominal* → *Noun*被应用于两个名词（*book*和*flight*）。这个规则也被应用于剖析右侧的树的单个名词（*flight*），构造出第三层的树。

一般来说，当剖析器从一层到下一层时，要查找被剖析的成分是否与某个规则的右手边相匹配。这与前面的自顶向下剖析正好相反，它要查找某个规则的左手边是否与某个还没有展开的非终极符号相匹配。因此，在第四层，在第一个和第三个剖析树中，序列*Det Nominal*被识别为规则*NP* → *Det Nominal*的右手边。

在第五层，把*book*解释为名词的树枝在搜索空间中被剪除。这是因为语法中没有以*Nominal NP*为右手边的规则，因而剖析无法进行下去。

搜索空间的最后一层（图 10.4 没有出现）是正确的剖析树（见图 10.1）。请读者确定，在倒数第二层的两个剖析中，哪一个能够导致正确的剖析结果。

10.1.3　自顶向下剖析与自底向上剖析的对比

这两种剖析方法各有自己的优点和缺点。自顶向下的策略绝不会浪费时间去搜索一个不可能以 *S* 为根的树，因为它恰恰是从 *S* 开始来生成树的。这意味着，自顶向下剖析绝不会去搜索那些在以 *S* 为根的树中找不到位置的子树。与此相反，在自底向上的策略中，那些没有希望导致 *S* 的树，或者那些与它们相邻的成分，都毫无控制地大量孳生出来。例如，在图 10.4 中，搜索空间的左半部的剖析完全是白费精力，这是因为这样的剖析在句子开始时就把 *book* 错误地解释为名词，而在给定的语法中，这样的树不可能推导出 *S*。

自顶向下的方法也有它的不足之处。虽然自顶向下方法不会浪费时间去剖析那些不可能推导出 *S* 的树，但这种方法却花费大量的努力去孳生那些与输入不一致的根为 *S* 的树。注意，在图 10.3 的第三层的六个树中，前四个树的左分支都不能与单词 *book* 匹配，因此这些树都不可能被用来剖析这个句子。自顶向下剖析的这个弱点是由于这种剖析方法在还没有检查输入符号之前就开始生成树了。反之，自底向上剖析绝不会去搜索那些不是以实际的输入为基础的树。

这两种剖析方法都不能有效地利用语法和输入单词中的约束条件。在下一节中，我们将介绍一种把自顶向下剖析和自底向上剖析结合起来的基本的剖析算法。这种剖析的效果虽然不如后面将要介绍的 Earley 算法和 CYK 算法那样好，但它对于说明剖析的基本操作还是很有用的。

10.2　基本的自顶向下剖析

把自顶向下剖析和自底向上剖析的优点结合起来的办法有许多种。有一个非常直接的办法是使用一种技术作为基本的控制策略来进行树的生成，然后使用另一种技术作为约束条件把搜索空间的各层次中那些不恰当的剖析加以过滤。本节中要研究的剖析器使用自顶向下的控制策略，同时又使用自底向上的过滤机制来增强它。我们的第一步是研究如何具体实现上一节中描述的自顶向下策略。然后，用一个基于自底向上剖析的输入约束的过滤器来过滤那些不恰当的剖析，把这个过滤器移植到自顶向下剖析器中。

在讨论自顶向下剖析和自底向上剖析的时候，假定我们要并行地搜索所有可能的剖析树。因此，在图 10.3 和图 10.4 中的每一层，都要说明对前一层的剖析树的所有可能的展开情况。虽然一定能够实现这样的方法，但是还需要使用一个非现实的存储器来存储在剖析过程中构造出来的树空间。由于实际的语法要比图 10.2 中的微型语法存在更多的歧义，这样做是特别重要的。

一种更合理的方法是使用深度优先策略（depth-first-strategy），这种策略在第 2 章和第 3 章中介绍各种有限状态机时曾经使用过。深度优先方法递增地展开搜索空间，系统地在一个时刻搜索一个状态。选择展开的状态也就是最新的生成状态。如果深度优先策略达到的树与输入符号串不一致，就返回到最新生成的、但是还没有被搜索过的树继续进行搜索。这种策略的纯效应在于：剖析只是一心一意地紧追着树，在返回到搜索过程的前一步已经生成的树的处理工作之前，剖析一直在试探对于树的处理是成功了或者是失败了。图 10.5 使用图 10.2 中的语法描述了一个自顶向下、深度优先的推导过程。

注意，这个推导并不是完全由自顶向下、深度优先策略的规定来决定的，还存在着两种没有被规定的选择，会导致不同的推导结果。一种是对于树中要展开的叶子结点的选择，一种是对于语法规则应用顺序的选择。在这个推导中，当前树中最左的还未展开的叶子结点是首先要展开的结点，而语法中可用的规则是根据它们在语法中的文本顺序来应用的。确定首先展开最左的未展开结点是

非常重要的，因为它决定了输入单词中用来构造树的单词的顺序。具体地说，它以一种相对自然的方式影响了输入单词加入到树中的前后顺序。按文本顺序来应用语法中的规则，对于这种选择，我们将在以后讨论。

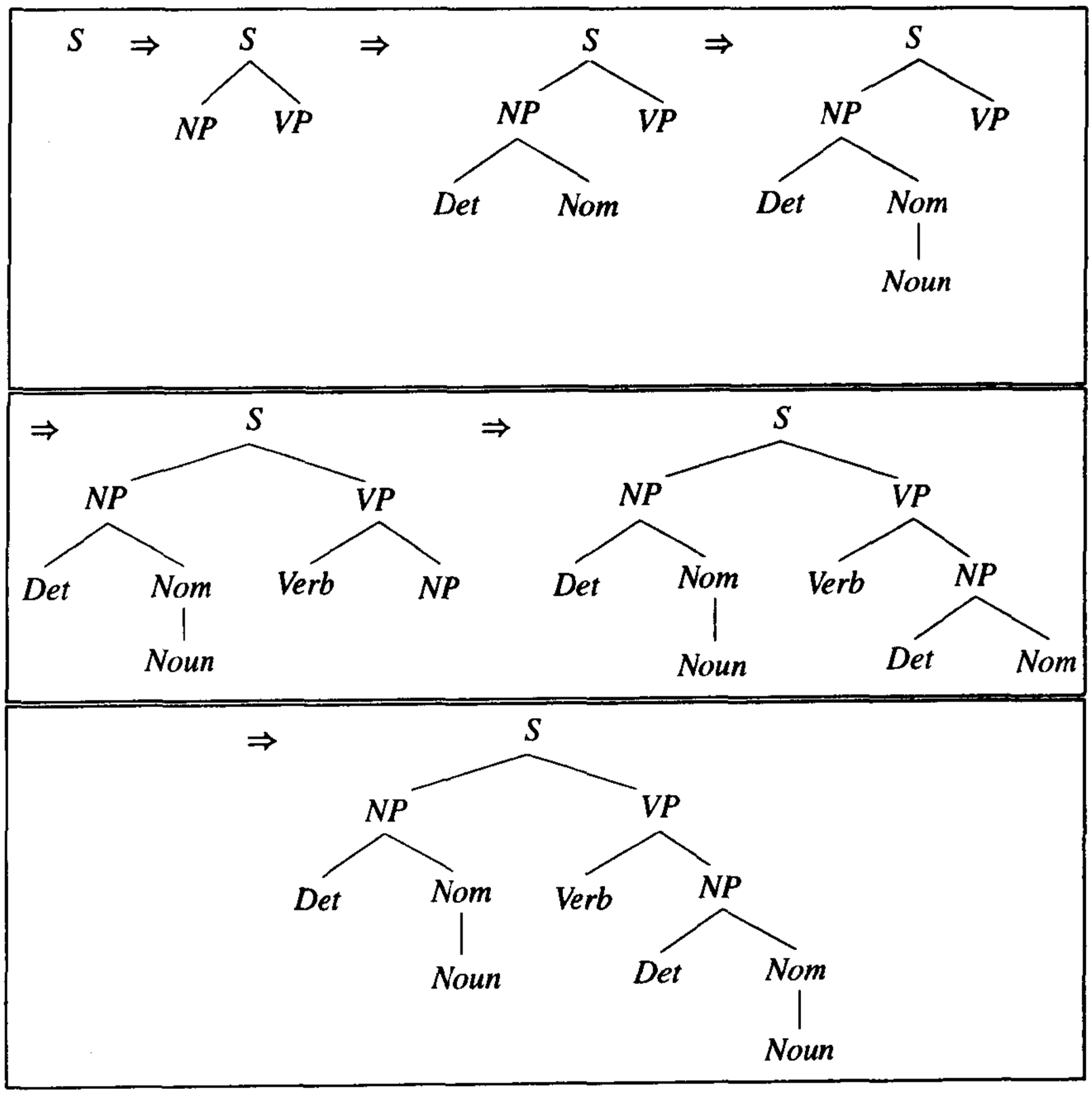


图 10.5　使用图 10.2 中的语法进行自顶向下、深度优先的推导

图 10.6 是一个剖析算法，具体说明了自顶向下、深度优先、从左向右的策略。这个算法维护一个搜索状态的项目表（agenda）。每个搜索状态由局部树和指向句子中下一个单词的指针组成。

剖析中的主要环路（loop）从项目表的前面取一个状态，使用所有的可应用语法规则，从与该状态相关的树中最左的未展开结点，产生一个新的状态集。然后，根据用来生成它们的语法规则的文本顺序，把这个新的状态集加入到项目表的前面。这个过程一直进行到发现一个成功的剖析树，或者项目表变空，说明输入不能剖析。

图 10.7 说明在剖析下面的句子时，这个算法检查的状态序列：

Does this flight include a meal?　　(10.2)

在这个图中，当前被展开的结点用方框表示，当前输入单词用方括号表示。被方括号括起来的单词的左边是已经被包含在树中的词。

开始时，剖析使用规则 $S \rightarrow NP\ VP$ 进行毫无成果的搜索，最后还是失败了，因为单词 *Does* 不可能从任何以 *NP* 开头的词类中推导出来。因此，剖析把规则 $S \rightarrow NP\ VP$ 删除。项目表中的下一个搜索状态相应于规则 $S \rightarrow Aux\ NP\ VP$。当找到了这个状态时，就按深度优先、从左向右的方式，对推导的其余部分继续进行搜索。

```
function TOP-DOWN-PARSE(input, grammar) returns a parse tree

 agenda ← (Initial S tree, Beginning of input)
 current-search-state ← POP(agenda)
 loop
  if SUCCESSFUL-PARSE?(current-search-state) then
    return TREE(current-search-state)
  else
    if CAT(NODE-TO-EXPAND(current-search-state)) is a POS then
      if CAT(node-to-expand)
          ⊂
          POS(CURRENT-INPUT(current-search-state)) then
        PUSH(APPLY-LEXICAL-RULE(current-search-state), agenda)
      else
        return reject
    else
      PUSH(APPLY-RULES(current-search-state, grammar), agenda)
  if agenda is empty then
    return reject
  else
    current-search-state ← NEXT(agenda)
end
```

图 10.6　自顶向下、深度优先、从左向右的剖析算法

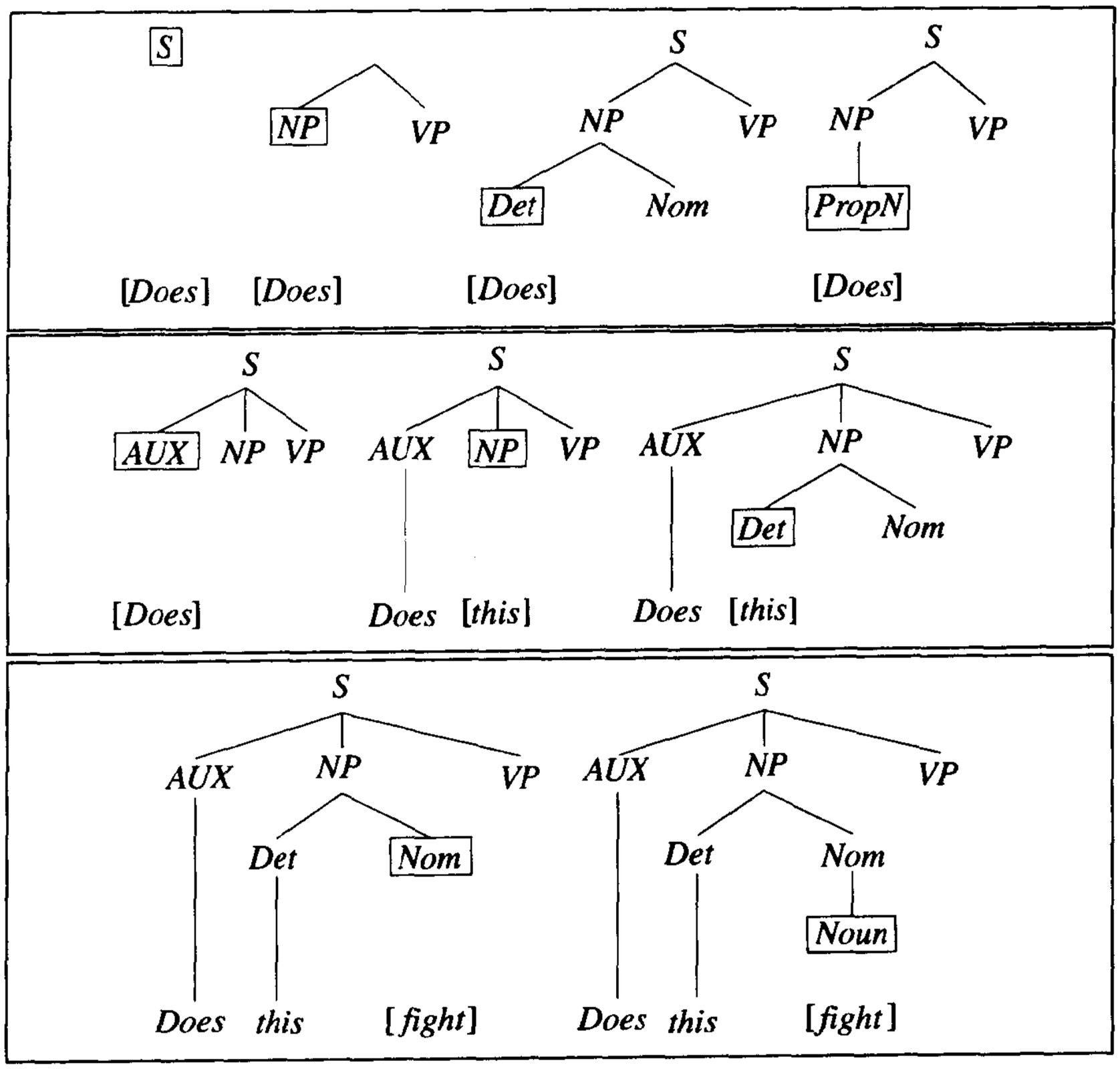


图 10.7　自顶向下、深度优先、从左向右的推导

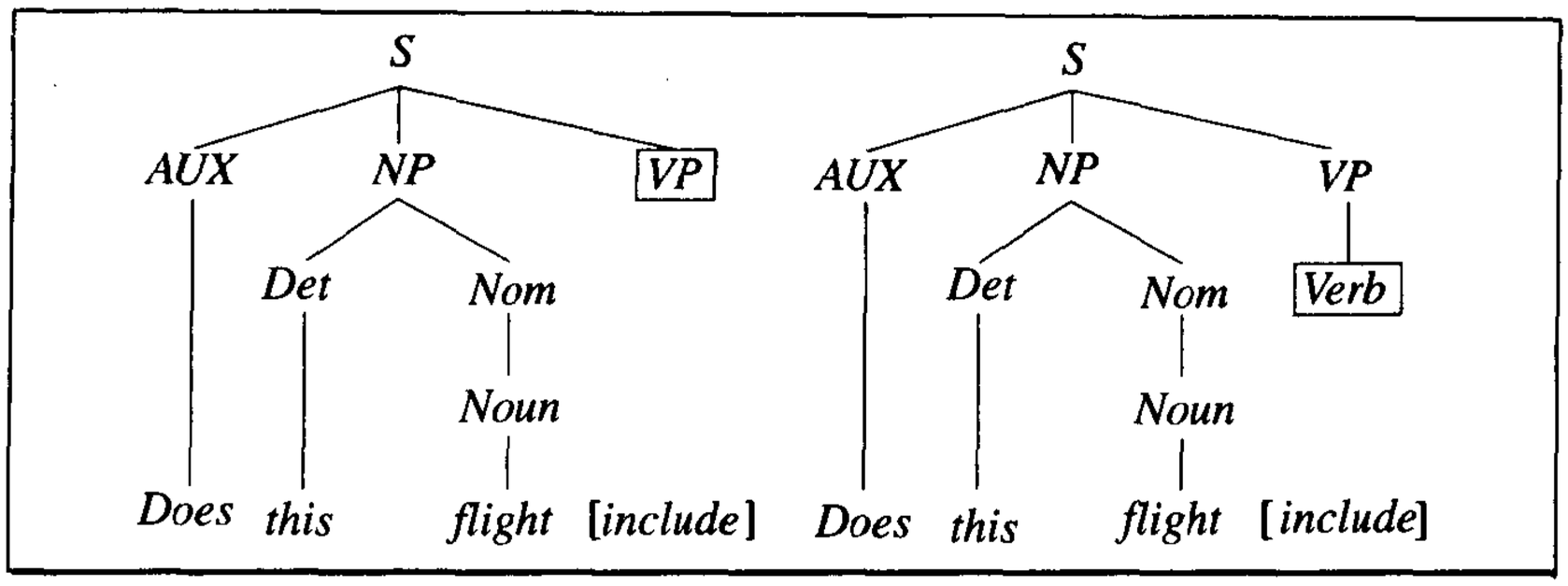
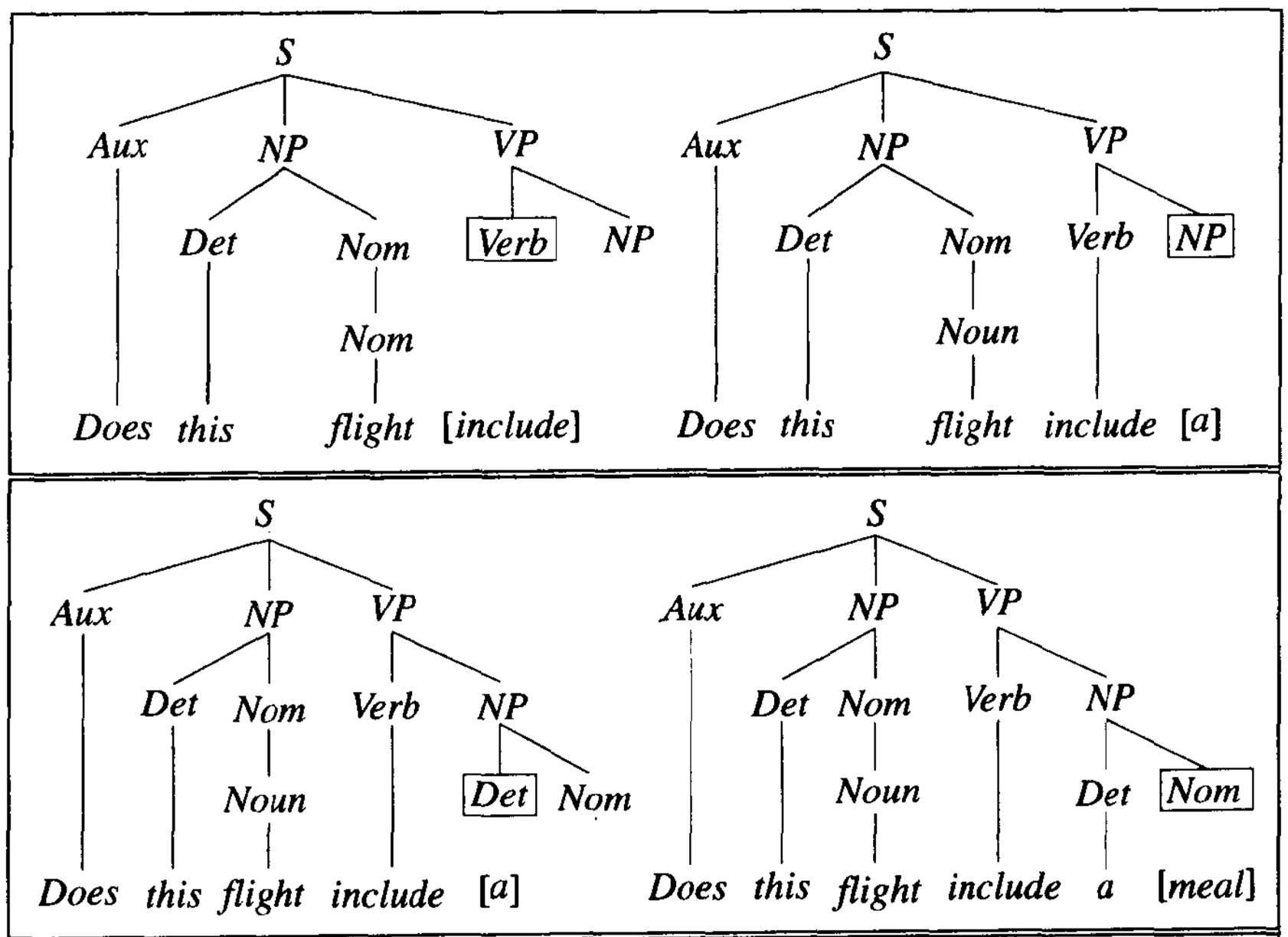


图 10.7　自顶向下、深度优先、从左向右的推导（续）



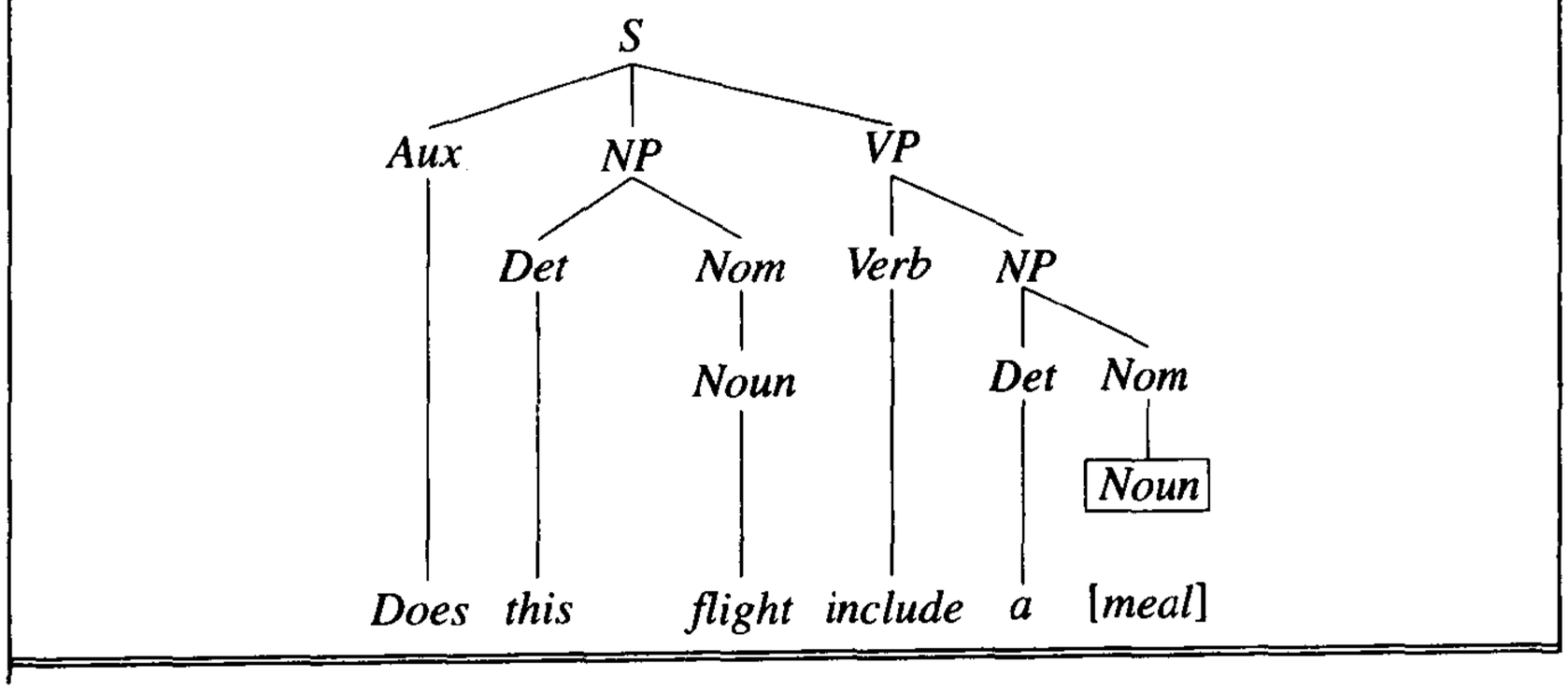


图 10.8　自顶向下、深度优先、从左向右的推导（续）

10.2.1　增加自底向上过滤

图 10.7 说明了自顶向下剖析的重要性质。剖析器从树的根开始，沿着这个树左侧的边，展开非终极符号，一直到这个树底部的最左的边。每当一个单词被加入到树中，输入的指针就向前移动一个位置，而剖析器就把下一个最左的非终极符号展开为新的最左的单词。

这样，在任何成功的剖析中，当前输入单词必定是剖析器当前处理的未展开结点的推导中的第一个单词。因此，在剖析器中增加自底向上过滤将会起到重要的作用。如果当前的输入不能作为语法规则中某个推导的最左的边上的第一个单词，剖析器就不考虑这个语法规则。沿着某个推导的最左的边上的第一个单词称为树的左角（left-corner）。

我们来研究图 10.9 中的 *VP* 的剖析树。如果把这个 *VP* 看成一个三角形。这个三角形的底部是单词。单词 *prefer* 位于树的左下角。形式地说，对于非终极符号 A 和 B，如果下面的关系成立：

$$A \stackrel{*}{\Rightarrow} B\alpha$$

那么，B 就是 A 的左角。

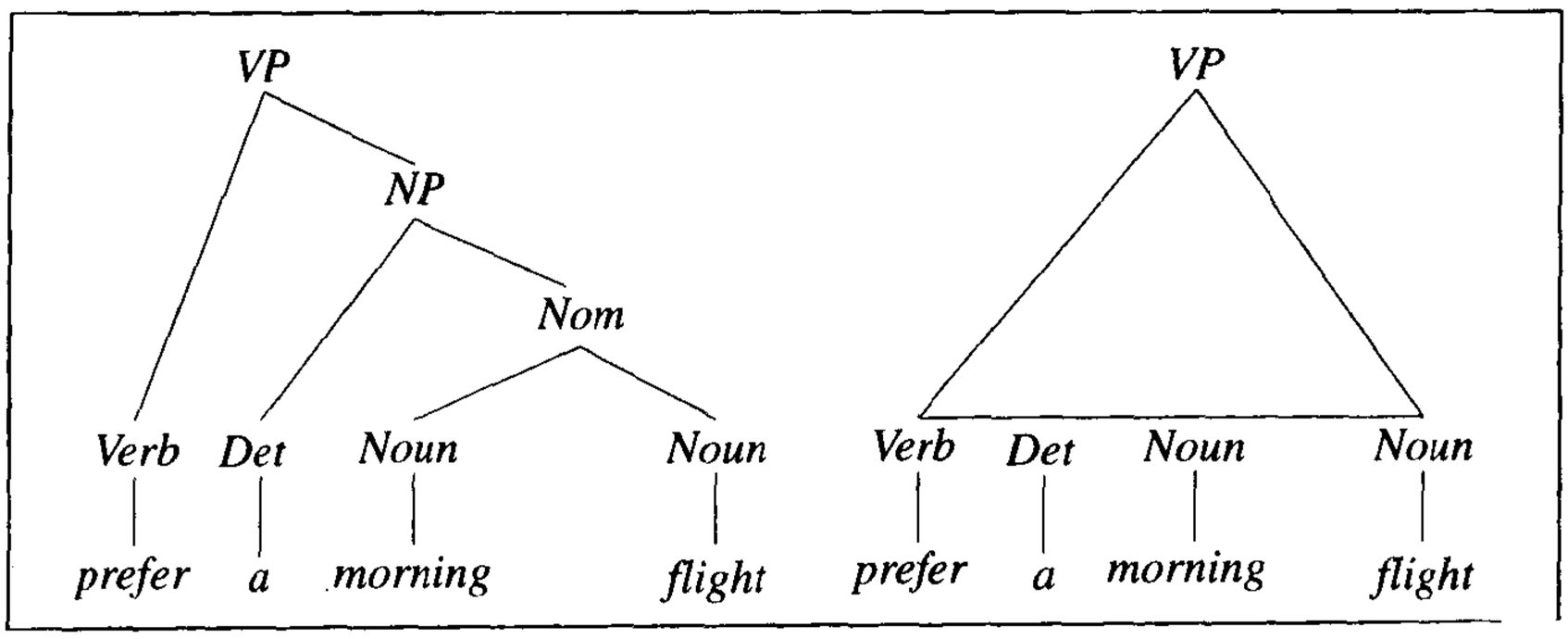


图 10.9　左角概念的图示。结点 *Verb* 和结点 *prefer* 都是 *VP* 的左角

换言之，如果存在以 B 为开始的 A 的一个推导，那么 B 就是 A 的左角。

我们再回到前面举例的句子 *Does this flight include a meal*? 图 10.2 中的语法给我们提供了 3 条规则，可以用来展开范畴 S：

$$S \rightarrow NP\ VP$$

$$S \rightarrow Aux\ NP\ VP$$

$$S \rightarrow VP$$

使用左角的概念很容易看出，只有规则 $S \rightarrow Aux\ NP\ VP$ 是可行的候选，因为 *Does* 不能作为另外两个 S 规则所要求的 *NP* 或 *VP* 的左角。知道这个道理，剖析器就可以重点考虑规则 $S \rightarrow Aux\ NP\ VP$，而用不着像前面在图 10.7 中所做的那样，首先去构造其他东西或者失败之后再进行回溯了。

为了有效地实现这样的过滤，可以把所需的信息编制成一个表，表中列出语法中每个非终极符号的一切可行的左角的范畴。当考虑一个规则的时候，就在表的条目中查找该规则右手边开始的那个范畴。如果其中不包含与当前输入有关的任何词类，就从我们考虑的对象中删除这条规则。

下面是为图 10.2 的语法建立的左角表：

范畴	左角
S	Det, Proper-Noun, Aux, Verb
NP	Det, Proper-Noun
Nominal	Noun
VP	Verb

读者可以用这个左角表作为过滤器改进图 10.6 中的剖析算法。

10.3 基本的自顶向下剖析的问题

尽管我们采用自底向上过滤来增强自顶向下剖析，但图10.7中的自顶向下剖析器还有三个问题，这三个问题使得通用的剖析问题难以得到充分的解决。这三个问题是：左递归问题、歧义问题、子树的无效重复剖析问题。研究了这些问题的实质之后，我们将介绍Earley算法，这种算法完全可以避免这些问题。

10.3.1 左递归

在一个无限的搜索空间中进行搜索时，深度优先搜索有一个众所周知的弊病：这种搜索可能会迷失在无限深的路径中，永远不能回来访问那些还没有展开的状态。当使用左递归（left-recursive）语法时，这个问题马上就在自顶向下、深度优先、从左向右的剖析中表现出来了。形式地说，当一个语法至少包含一个非终极符号A，使得对于某个α和β以及$\alpha \stackrel{*}{\Rightarrow} \varepsilon$，有$A \stackrel{*}{\Rightarrow} \alpha A \beta$，这个语法就是左递归的。换言之，一个语法如果包含一个非终极范畴，而该非终极范畴的推导中又在沿着其最左树枝的任何地方包含它自身，这个语法就是左递归的。在第9章的语法中，在类似Atlanta's airport的主有*NP*的规则中，就有这样的左递归的例子：

$$NP \rightarrow Det\ Nominal$$

$$Det \rightarrow NP\ 's$$

这些规则将把左递归带进语法中，因为在*NP*的第一个成分*Det*的推导中又以一个*NP*作为*Det*的第一个成分。

自然语言语法中更明显和常见的左递归是直接左递归规则。直接左递归规则的形式是$A \rightarrow A\beta$，其中规则右手边的第一个成分与规则的左手边等同。下面是英语中常见的直接左递归规则：

$$NP \rightarrow NP\ PP$$

$$VP \rightarrow VP\ PP$$

$$S \rightarrow S\ and\ S$$

一个左递归的非终极结点可能引起自顶向下、深度优先、从左向右的剖析递归地展开同一个非终极符号，以完全相同的方式不断重复，导致剖析树的无限扩展。

图10.10说明了这种无限扩展的情况。如果给我们的微型语法中的第一条*NP*规则增加直接左递归规则$NP \rightarrow NP\ PP$，就会出现这种无限的扩展。

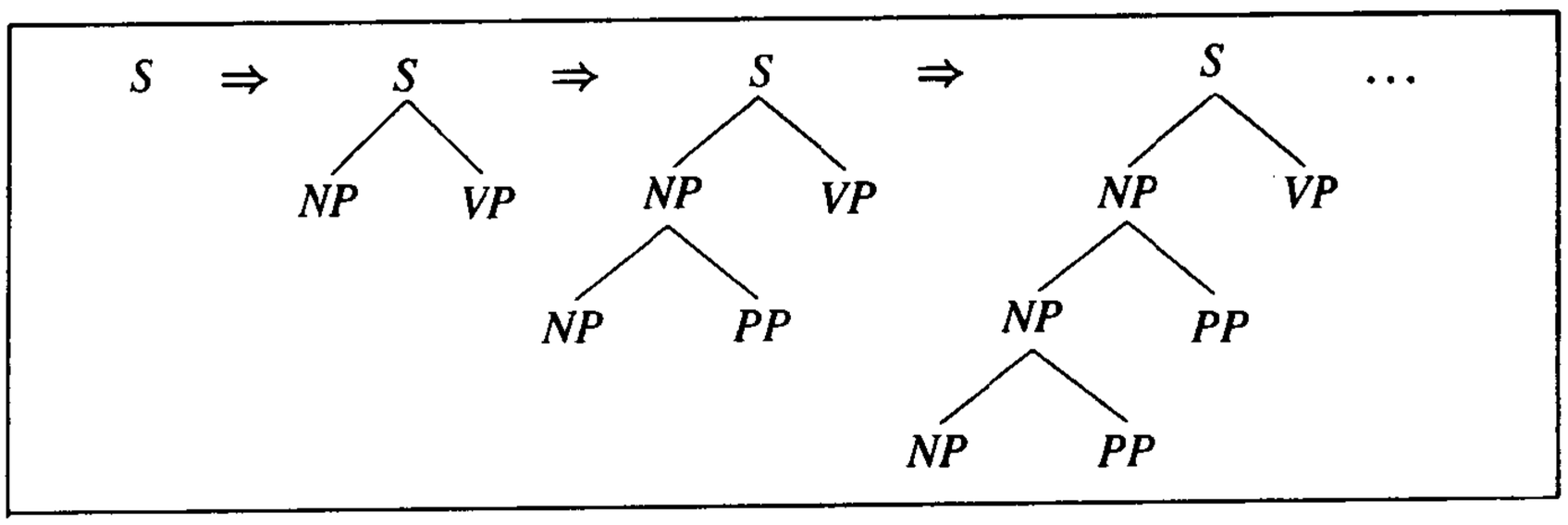


图10.10 由于左递归规则引起无限搜索开始时的情况

在一个有回溯的自顶向下剖析中，有两种合理的办法可以处理左递归：一个办法是对语法进行改写；一个办法是在剖析中明确地控制搜索的深度。在第9章中讲过，可以改写语法的规则，把一个语法变成一个弱等价的新语法，使得这个新语法能够像原来的语法那样接受完全相同的语言。通

过这种把左递归语法改写为弱等价的非左递归语法的办法，可以从某些常见的语法类中消除左递归语法。我们可以根据如下的图示，使用新的符号 A' 来改写每个形式为 $A \rightarrow A\beta$ 的左递归规则：

$$A \rightarrow A\beta \mid \alpha \qquad \Rightarrow \qquad \begin{aligned} A &\rightarrow \alpha A' \\ A' &\rightarrow \beta A' \mid \varepsilon \end{aligned}$$

这样的转换把左递归变成右递归，而这些规则所形成的树由左分支结构变成了右分支结构。遗憾的是，用这种办法改写的语法有一个很大的缺点：改写过的短语结构规则可能不再能以最符合语法特性的方式来表示特定的句法结构。我们在第 15 章中可以进一步看到，这样的改写可能会造成语义解释上的困难。

10.3.2　歧义

One morning I shot an elephant in my pajamas. How he got into my pajamas I don't know.

Groucho Marx, *Animal Crackers*, 1930[①]

图 10.6 中自顶向下剖析的第二个问题是，这种算法不能有效地处理歧义（ambiguity）。第 8 章介绍了范畴歧义（单词具有一个以上的词类）和歧义消解（给一个单词选择正确的词类）。

在这一节中，我们介绍一种新的歧义，这种歧义称为结构歧义（structural ambiguity），它是在剖析句法结构时发生的。当语法可以给一个句子指派一个以上的剖析时，就会发生结构歧义。如图 10.11 所示，我们在本节开头引用的 Groucho Marx 关于 Spaulding 上尉的有名的句子就是有结构歧义的，因为其中的 *in my pajamas* 这个短语可以是以 *elephant* 为中心语的名词短语 *NP* 的一部分，也可以是以 *shot* 为中心语的动词短语 *VP* 的一部分。

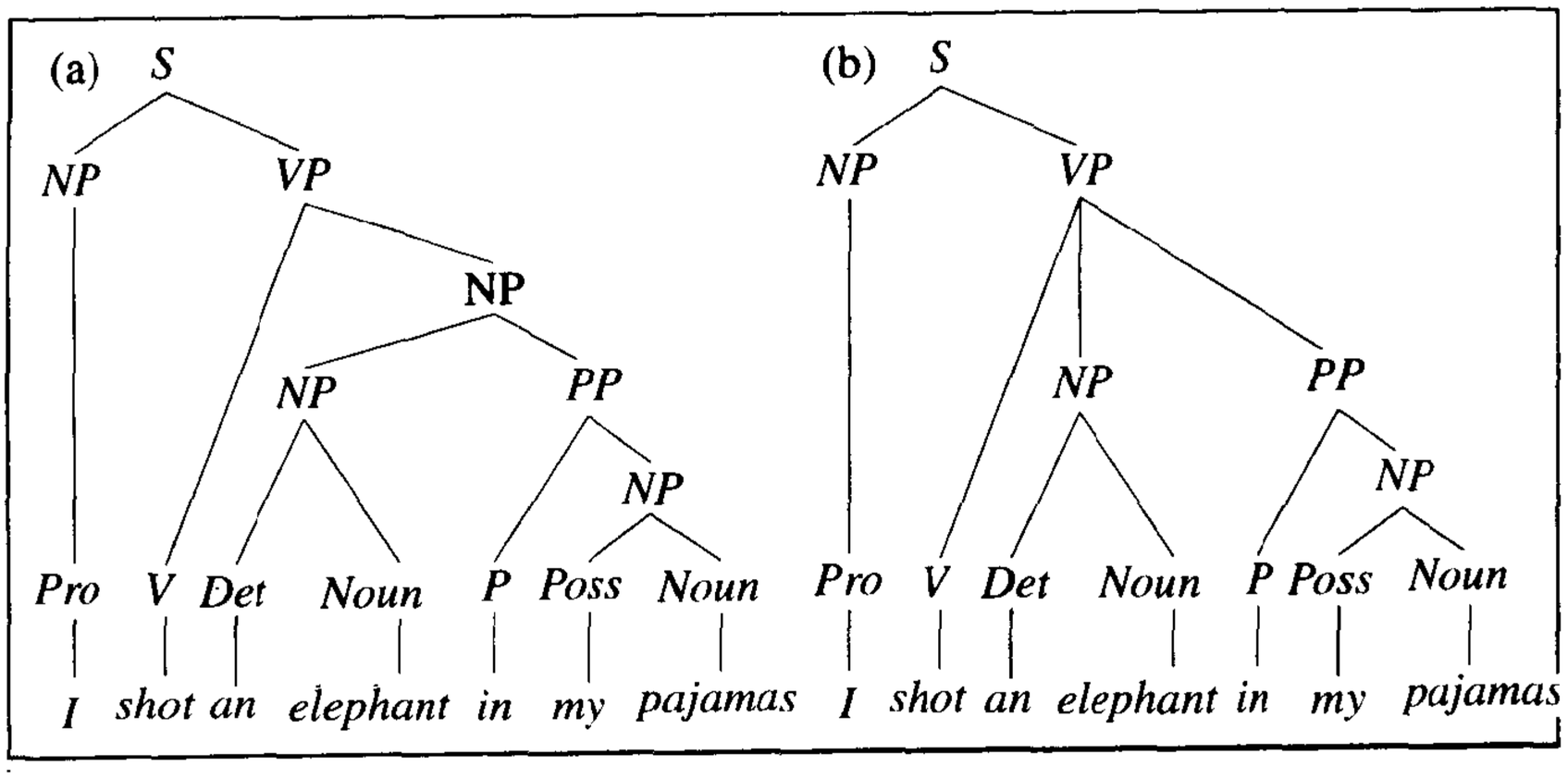


图 10.11　一个歧义句子的两个剖析树。剖析（a）的意思有些异想天开，睡衣上居然有大象；剖析（b）的意思是 Spaulding 上尉穿着他的睡衣射击

结构歧义有很多种形式。英语中最常见的结构歧义有三种：附着歧义（attachment ambiguity）、并列连接歧义（coordination ambiguity）和名词短语层次歧义（noun-phrase bracketing ambiguity）。

如果一个特定的成分可以附着在剖析树的一个以上的位置，句子就可能出现附着歧义。前面的句子就是 PP 附着歧义的一个例子。各种类型的状语短语也会出现附着歧义。例如，在下面的例子中，动名词 VP *"flying to Paris"* 可以作为以 *Eiffel Tower* 为主语的动名词从句的一个部分（意思是

① 这是一个歧义句。一个意思是："一个早晨我指责在我的睡衣上的大象。我不明白，它是怎么到我的睡衣上的。"另一个意思是："一个早晨我穿着睡衣向大象开枪。我不明白，它是怎么冲到我的睡衣上来了。"——译者注

"我们看见 Eiffel 铁塔飞向了 Paris"），也可以作为一个附加语（adjunct）来修饰以 saw 为中心语的 *VP*（意思是"当我们飞向 Paris 时看见了 Eiffel 铁塔"）。

We saw the Eiffel Tower flying to Paris. （10.3）

句子 *Can you book TWA flights* 也有类似的歧义。这个歧义句的意思可以是 *Can you book flights on behalf of TWA*（您能以 TWA 的名义订飞机票吗?）；也可以是 *Can you book flights run by TWA*（您能订TWA公司的飞机票吗?）。当采用前面一个意思时，理解为有两个NP，分别是动词短语 VP 的两个不同的姐妹结点；当采用后面一个意思时，理解为一个 NP 附着到另一个 NP 上，形成一个复杂的 NP（*TWA flights*）。图 10.12 说明了这两种不同的剖析。

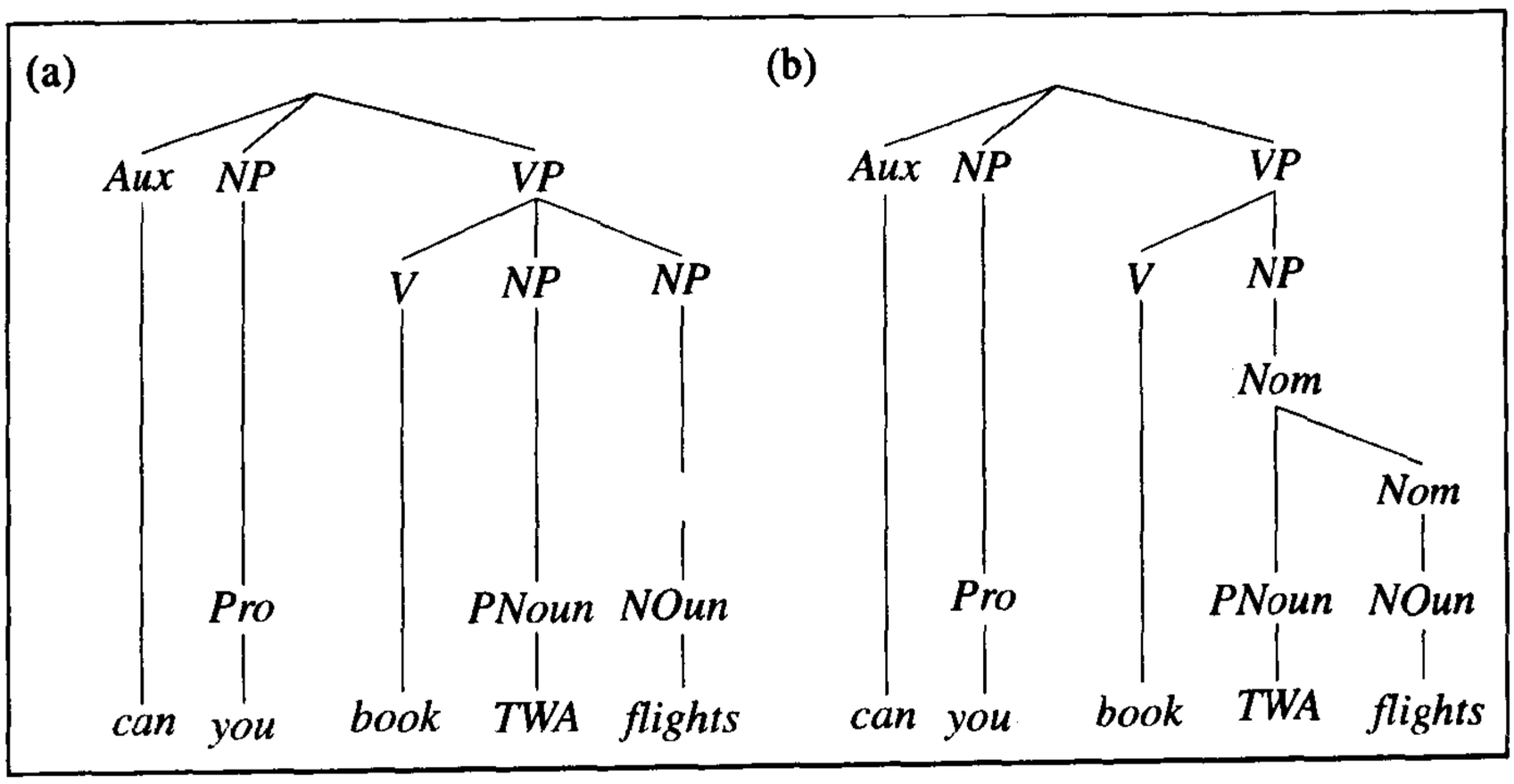


图 10.12 一个歧义句的两个剖析树。剖析（a）的意思是 *Can you book flights on behalf of TWA*（您能以 TWA 的名义订飞机票吗?），剖析（b）的意思是 *Can you book flights which are run by TWA*（您能订 TWA 公司的飞机票吗？）

另外一种结构歧义是并列连接歧义。在这种歧义结构中，存在着不同的短语，这些短语之间用 and 这样的连接词相连。例如，*old men and women* 这个短语的括号式层次可以是[*old* [*men and women*]]，意思是 *old men and old women*（年老的男人和年老的女人）；也可以是[*old men*] *and* [*women*]，意思是 *only the men who are old*（只有男人是年老的，而女人不一定都是年老的）。

这些歧义又都能够以很复杂的方式结合起来。下面是来自 Brown 语料库中关于新闻综述节目的一个例子，需要我们对句子进行剖析：

President Kennedy today pushed aside other White House business
to devote all his time and attention to working on the Berlin crisis address he will
deliver tomorrow night to the American people over nationwide television and radio. （10.4）

这个句子有若干个歧义，其中有的歧义在语义上是不合理的。要细心地阅读这个句子才能发现这些歧义。最后一个名词短语可以剖析为[*nation wide* [*television and radio*]]，或[*nation wide television*] and [*radio*]。*pushed aside* 的直接宾语可以是 *other White House business*，但也可以是[*other White House business to devote all his time and attention to working*]这个意思有点儿古怪的短语（这个结构就像 *Kennedy denied* [*his attention to propose a new budget to address the deficit*]）。*on the Berlin crisis address he will deliver tomorrow night to the American people* 这个短语可以作为动词 *pushed* 的附加修饰语。*over nationwide television and radio* 这个 *PP* 可以附着于前面的任何一个 *VP* 或 *NP*（例如，这个 *PP* 可以修饰 *people* 或 *night*）。

如果一个句子有很多不合理的剖析结果，这是一个非常令人头痛的问题，它会影响到所有的剖析算法。在实际上，剖析一个句子需要进行歧义消解（disambiguation）。所谓歧义消解就是从多个可能的剖析中选择一个正确的剖析。歧义消解算法一般既需要统计知识，又需要语义知识，这些问题将在后面的第 12 章和第 17 章中介绍。

对于没有进行过歧义消解的剖析，可以简单地把所有可能的剖析树都返回给一个给定的输入。由于图10.7中的自顶向下剖析只返回了它所发现的第一个剖析，因此还有必要加以修改，使之能够返回所有可能的剖析结果。算法也需要加以改变，使之能够收集每个发现的剖析并继续查找更多的剖析。当搜索空间穷尽时，就把所有发现的剖析树做成一个列表加以返回。然后，再通过后续的处理或人工的分析，决定这些返回的剖析中哪个是正确的。

不过，在实际应用中，我们几乎可以肯定地说，我们并不希望从一个鲁棒的、高度歧义的、覆盖面广泛的语法中，得到所有可能的剖析结果。其原因在于，对于一定的输入来说，可能会潜在地存在着指数级的剖析结果。我们来研究 ATIS 的例句（10.5）：

Show me the meal on Flight UA 386 from San Francisco to Denver.　　（10.5）

如图 10.13 所示，当把递归规则 $VP \rightarrow VP\ PP$ 和 $NP \rightarrow NP\ PP$ 加进我们那个很小的语法中以后，在这个句子后面的三个介词短语凑合起来一共能够给这个句子造出 14 个剖析树。例如，*from San Francisco* 这个介词短语还可能成为以 *show* 为中心语的 *VP* 的一部分（不过，对这种剖析结果的解释有些怪里怪气的：从 *San Francisco* 告诉我飞机上的用餐情况）。

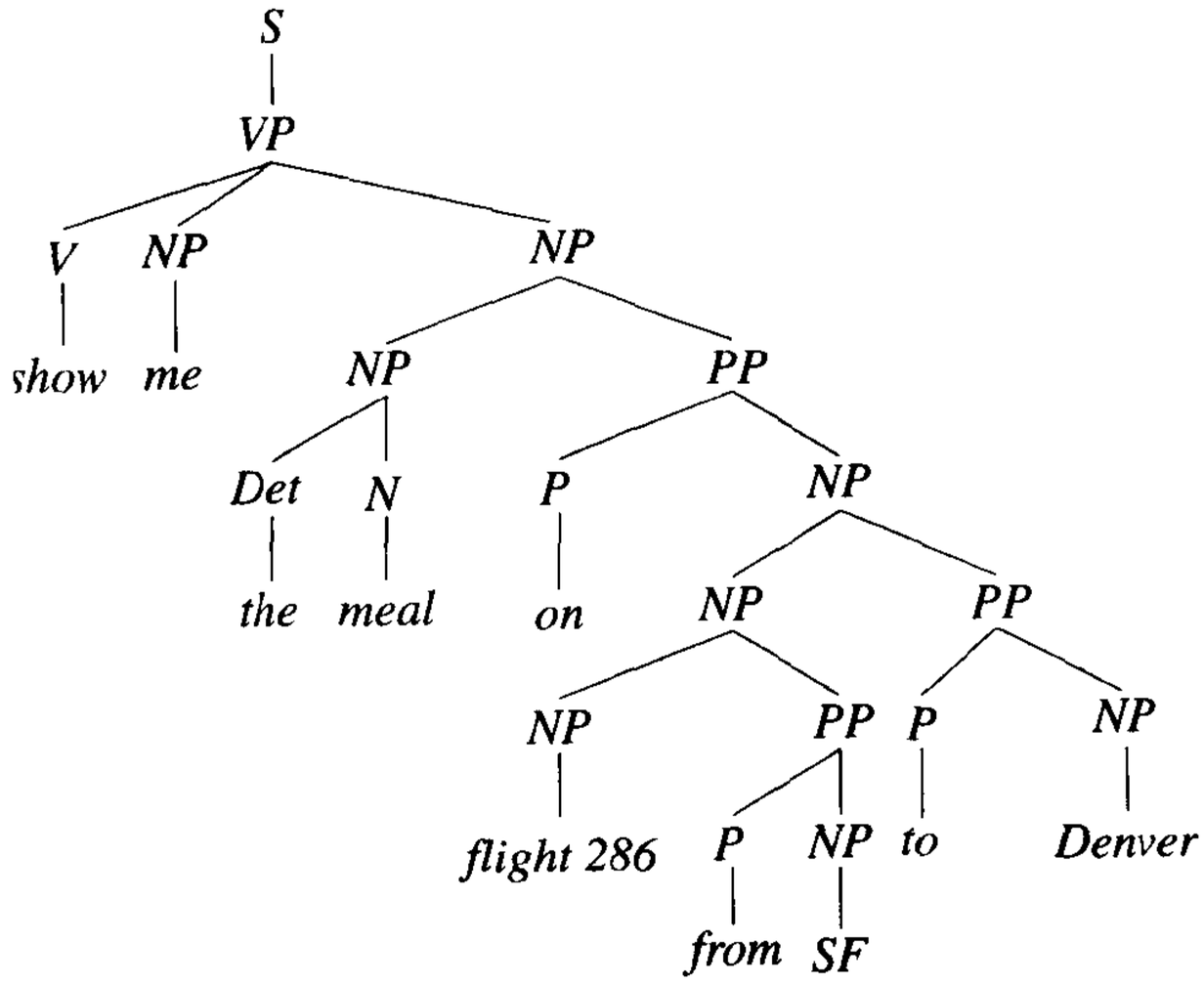


图 10.13　例句（10.5）的一个正确的剖析结果

Church and Patil(1982)说明，这种类型的句子的剖析数目的增长速率与算术表达式插入括号数量的增长速率相同。这个括号插入问题（parenthesization problem）是以所谓的 Catalan 数（Catalan number）按指数增长的：

$$C(n) = \frac{1}{n+1}\begin{pmatrix} 2n \\ n \end{pmatrix}$$

下面的表说明了简单名词短语的剖析的数目是一连串介词短语的数目的函数。可以看出，这种类型的歧义能够很快地使每个可能的剖析结果大量增长。

PP数目	NP 剖析数目
2	2
3	5
4	14
5	132
6	469
7	1430
8	4867

要摆脱这种困境有两个基本方法。一种方法是使用动态规划来研究搜索空间的规律性，使得常见的部分只能推导一次，从而减少与歧义有关的开销；另一种方法是使用试探性的方法来改善剖析器的搜索策略，从而指导剖析器首先进行最恰当的剖析。我们在下一节将研究动态规划方法，第12章将介绍试探性的搜索策略。

尽管一个句子是没有歧义的，但是由于这个句子存在局部歧义（local ambiguity），也会使得剖析效率不高。当句子的某个部分有歧义时，就会发生局部歧义，这时，尽管整个句子没有歧义，也要进行一个以上的剖析。例如，句子 *Book that flight* 是没有歧义的，但是当剖析器看到第一个单词 *Book* 时，它不知道这个单词应该是名词还是动词（直到剖析的最后才可能知道），因此剖析器必须使用回溯或者并行地考虑两种可能的剖析。

10.3.3 子树的重复剖析

歧义问题还与10.2节中自顶向下剖析的另一种引起低效率的因素有关。剖析器经常会对于输入的某个局部构造出正确的树，在回溯时把这个树抛弃，但后来又发现还应该重新构造这些已被抛弃的树。我们来研究发现例句（10.6）中的 *NP* 的一个正确剖析的过程：

a flight from Indianapolis to Houston on TWA （10.6）

使用10.2节中介绍的剖析算法得到的正确剖析结果如图10.14最下面的剖析树所示。这个名词短语存在五个不同的剖析，现在我们只重点讨论在搜索这个正确的剖析结果过程中出现的荒唐的重复工作量问题。

因为我们使用自顶向下、深度优先、从左向右的方法，剖析器首先得到一个小的剖析树，但由于这个小剖析树不能覆盖整个输入，所以剖析失败。一系列连续的失败导致不断地进行回溯，使得剖析递增覆盖越来越多的输入。我们的这个自顶向下的剖析器构造出的一系列树如图10.14所示。

这个图清楚地描绘了由于回溯方法而造成的这种无聊而笨拙的重复工作。除了最顶端的成分之外，最后形成的剖析树的每个部分都要被推导一次以上。下面的表说明了最后形成的树的每个主要成分的推导次数。

a flight	4
from Indianapolis	3
to Houston	2
on TWA	1
a flight from Indianapolis	3
a flight from Indianapolis to Houston	2
a flight from Indianapolis to Houston on TWA	1

显而易见，如果是在动词短语一级或者在句子一级的回溯而产生的工作量，将比这个名词短语的例子所做的工作量还大得多。注意，尽管这个例子是针对自顶向下剖析的，但这种白费力气的工作在自底向上剖析中也是存在的。

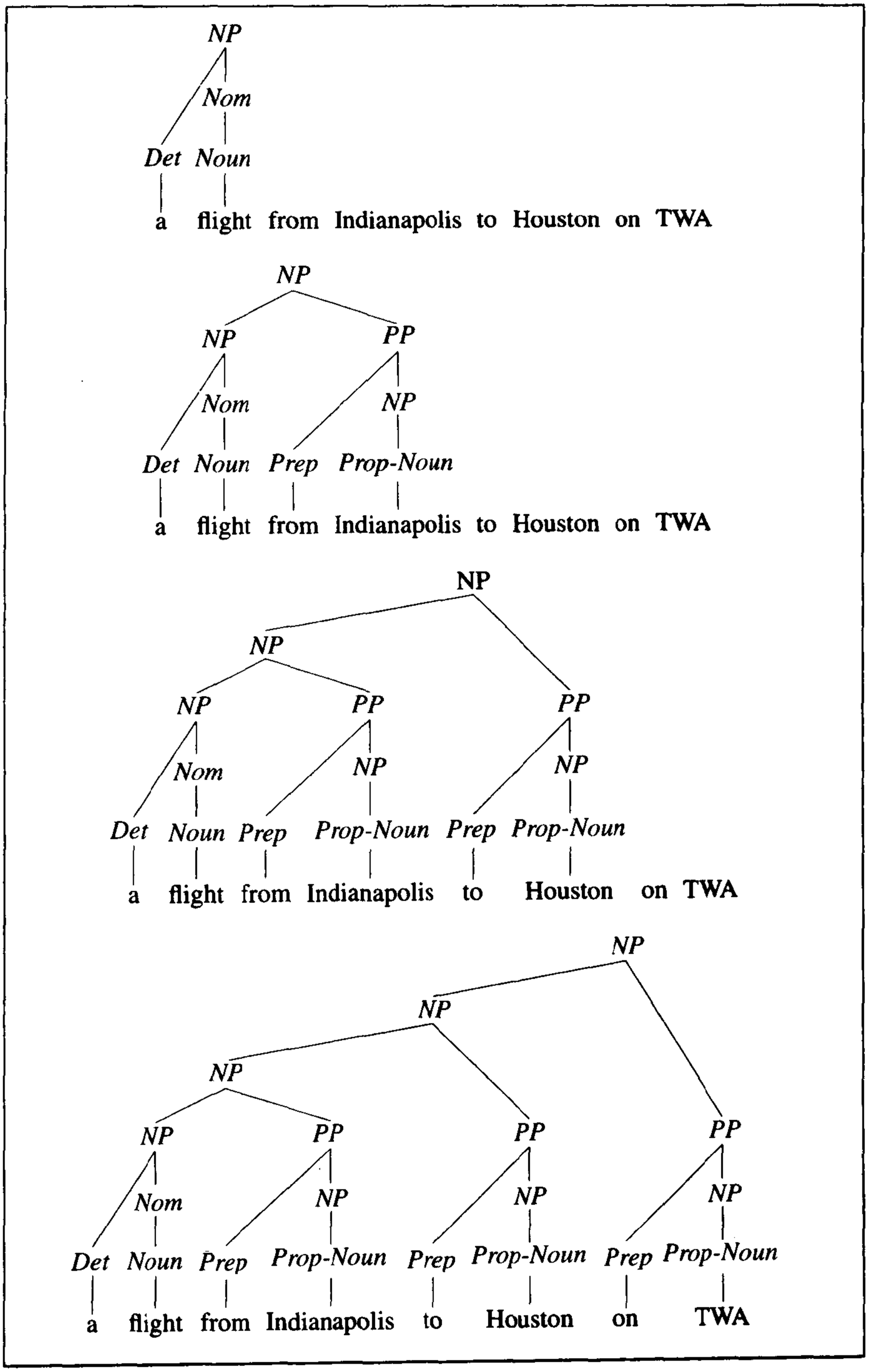


图 10.14　在自顶向下剖析中因回溯而造成的重复工作

10.4 Earley 算法

前面一节指出，尽管这样的标准算法可以使用过滤和其他方法得到改善，但是，仍然有三个问题使这样的标准的自底向上剖析或自顶向下剖析感到束手无策。这三个问题是：左递归规则、歧义和无效的子树重复剖析。幸运的是，有一类简单算法能够解决所有这些问题。动态规划曾经在最小编辑距离、Viterbi 算法和向前算法中提供过帮助。现在，动态规划又为解决上述问题提供了框架。我们知道，动态规划方法系统地把对于子问题的解填到表中。当填完表时，动态规划的表中包含了对所有子问题的解，而这些子问题是解决整个问题所必需的。在剖析的场合，这样的表用来存储输入中各个成分的子树，当这些成分被发现时，就把它们的子树存储到表中。一旦发现这些子树，就马上存储到表中，然后在整个剖析过程中调用这些成分，从而提高剖析的功效。这样就解决了重复剖析的问题（只需要查找子树，而不必重新剖析）和歧义问题（剖析表隐含地存储着所有可能的剖析结果，因为表中存储了所有成分以及它们可能导致的剖析）。另外，动态规划剖析算法也可以解决左递归的问题。我们前面讨论过，有三个著名的动态规划剖析算法：Cocke-Younger-Kasami 算法（CYK 算法），Graham-Harrison-Ruzzo 算法（GHR 算法）(Graham et al., 1980)和 Earley 算法(Earley, 1970)。我们将在第 12 章介绍 CYK 算法，在本章的下面介绍 Earley 算法。

Earley算法(Earley, 1970)使用动态规划方法有效地实现了10.1节中讨论过的那种并行的自顶向下搜索。由于使用了很多动态规划的解，这种算法消除了回溯方法中固有的子问题重复求解的弊病，把外观上的指数时间复杂度问题（exponential-time）降低到多项式时间复杂度（polynomial-time）问题。这样，动态规划方法在最坏的情况下，其行为的复杂度为 $O(N^3)$，其中 N 是输入中的单词数目。

Earley 算法的核心是一个从左到右的传递，它填充一个称为线图（chart）的数组，如果输入有 N 个单词，则线图含有 $N+1$ 个项目（entry）。对于句子中每个单词的位置，线图包含一个状态表来表示已经生成的部分剖析树。在句子的结尾，线图把对于给定输入的所有可能的剖析结果简洁地进行编码。每个可能的子树只表示一次，并且这个子树表示可以被需要它的所有的剖析共享。

包含在每个线图项目（chart entry）中的状态含有三种信息：关于与语法的一个规则相对应的子树的信息，关于完成这个子树已经通过的进程的信息，关于这个子树相对于输入的位置的信息。用图形来表示，我们将在某一状态的语法规则的右手边中，使用一个点（dot）来标志识别这个状态所走过的进程。这样形成的结构称为点规则（dotted rule）。一个状态对于输入的位置用两个数来表示，分别说明该状态开始的位置以及点所在的位置。我们来研究下面三个状态的例子，它们是使用 Earley 算法在剖析例句（10.7）的过程中产生的：

Book that flight. ［与例句（10.1）相同］ （10.7）

$$S \rightarrow \bullet\, VP, [0,0]$$

$$NP \rightarrow Det \bullet Nominal, [1,2]$$

$$VP \rightarrow V\, NP \bullet, [0,3]$$

在第一个状态中，点处于成分的左侧，表示自顶向下地预测这个特定的开始结点 S。第一个 0 表示这个状态所预测的成分开始于输入符号串的开头；第二个0表示点也在开始的位置。第二个状态是在处理这个句子的下一个阶段产生的，表明 NP 开始于位置 1，Det 已经被成功地剖析，期待下一步处理 Nominal。在第三个状态中，点处于规则中两个成分的右侧，表示已经成功地找到了与 VP 相对应的树，而且这个 VP 横跨在整个输入符号串上。这些状态也可以用图来表示，其中剖析的状态是边（edge）或弧（arc），线图是一个有向非成圈图（directed acyclic graph），如图 10.15 所示。

Earley剖析算法的基本操作是以从左向右的方式走过线图中的由$N+1$个状态组成的集合，按顺序处理每个集合中的各个状态。在每一步，根据具体情况将下面描述的三个操作中的一个应用于每

个状态。在每个场合，这样的操作的结果会增加一个新的状态，这个新的状态或者加在当前状态的后面，或者加在该线图的下一个状态的集合中。这样的算法始终通过线图向前移动，每向前移动一步增加一个新的状态，这些状态从来不会被抹掉，当算法向前移动之后，就再也不会回溯到前面已走过的线图项目中。在最后一个线图项目的状态表中的状态$S \rightarrow \alpha \bullet, [0, N]$表示剖析已经成功。图 10.16 给出了一个完全的算法。

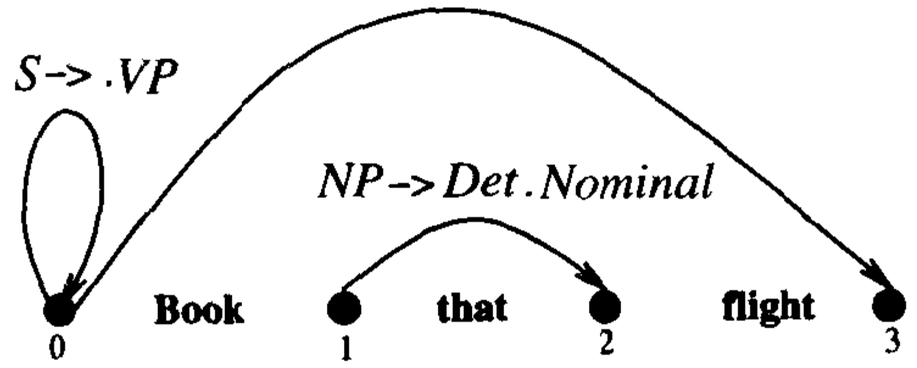


图 10.15　表示上面三个点规则的有向非成圈图

```
function EARLEY-PARSE(words, grammar) returns chart

  ENQUEUE((γ → • S, [0,0]), chart[0])
  for i ← from 0 to LENGTH(words) do
   for each state in chart[i] do
     if INCOMPLETE?(state) and
              NEXT-CAT(state) is not a part of speech then
        PREDICTOR(state)
     elseif INCOMPLETE?(state) and
              NEXT-CAT(state) is a part of speech then
        SCANNER(state)
     else
        COMPLETER(state)
   end
  end
  return(chart)

procedure PREDICTOR((A → α • B β, [i, j]))
   for each (B → γ) in GRAMMAR-RULES-FOR(B, grammar) do
       ENQUEUE((B → • γ, [j, j]), chart[j])
   end

procedure SCANNER((A → α • B β, [i, j]))
   if B ⊂ PARTS-OF-SPEECH(word[j]) then
      ENQUEUE((B → word[j], [j, j + 1]), chart[j+1])

procedure COMPLETER((B → γ •, [j,k]))
   for each (A → α • B β, [i, j]) in chart[j] do
       ENQUEUE((A → α B • β, [i,k]), chart[k])
   end

procedure ENQUEUE(state, chart-entry)
   if state is not already in chart-entry then
      PUSH(state, chart-entry)
   end
```

图 10.16　Earley 算法

下面三节详细地描述在线图中处理状态的三个操作。每个操作取一个单独的状态作为输入，并从这个状态推导出一个新的状态。只要新的状态不再是当前状态，就把它们加到线图中。“预测”（Predictor）和“完成”（Completer）操作把状态加到正在处理的线图项目中，而“扫描”（Scanner）操作把状态加到下一个线图项目中。

10.4.1 预测

预测操作的名称是Predictor。从名称可以知道，Predictor的任务是造出一个新的状态来表示在剖析过程中生成的自顶向下的预测。Predictor应用于点规则中点的右侧为非终极符号但又不是词类范畴的任何状态。使用Predictor的结果，对于语法提供的这个非终极符号的每个不同的展开，都造出一个新的状态。这些新的状态作为生成的状态被放到同样的一个线图项目中。它们从同一个点开始，在同一个点结束，这个点就处于输入中生成状态结束的地方。

例如，在状态 $S \rightarrow \bullet\ VP, [0, 0]$，如果我们使用Predictor，其结果是在第一个线图项目中增加状态 $VP \rightarrow \bullet\ Verb, [0, 0]$和状态 $VP \rightarrow \bullet\ Verb\ NP, [0, 0]$。

10.4.2 扫描

扫描操作的名称是Scanner。当在点规则中点的右侧是词类范畴时，就调用Scanner来检查输入符号串，并把对应于所预测的词类范畴的状态加入到线图中。使用Scanner操作之后，从输入状态中造出一个新的状态，并且点规则中的点跨过所预见的输入范畴，向前移动一个位置。注意，Earley剖析算法使用自顶向下输入来帮助对于词类歧义进行歧义消解；只有那些被某个状态所预见的单词的词类才能在线图中找到它们的路径。

再回到我们的例子，当处理状态 $VP \rightarrow \bullet\ Verb\ NP, [0, 0]$时，因为点后面的范畴是词类，所以Scanner要在输入中查找当前的单词。这时，Scanner注意到，*book*可能是一个*Verb*，它与当前状态中的预测相匹配，其结果是造出一个新的状态 $VP \rightarrow Verb\ \bullet\ NP, [0, 1]$。然后，把这个新的状态加到线图项目中，跟随在当前处理过的状态之后。

10.4.3 完成

完成操作的名称是Completer。当一个状态中的点到达规则的右端时。从直观上说，这样的状态说明剖析算法已经成功地找到了在输入的某个跨度上的一个特定的语法范畴。Completer的目标就是查找输入中在这个位置的语法范畴，发现并且推进前面造出的所有状态。造新的状态时可以复制老的状态，但是要把点规则中的点跨过所预测的范畴向前推进，并把这个新的状态安置在当前的线图项目中。

例如，当处理状态 $NP \rightarrow Det\ Nominal\ \bullet, [1, 3]$时，Completer要查找以1为结尾并预测*NP*的状态，在这个状态，结尾是1而且要预测*NP*。在当前的例子中，它要找到由Scanner造出的状态 $VP \rightarrow Verb\ \bullet\ NP, [0, 1]$。其结果是加入一个新的完成状态 $VP \rightarrow Verb\ NP\ \bullet, [0, 3]$。

10.4.4 示例

图10.17说明了在剖析例句（10.1）或例句（10.7）的整个过程中造出的状态序列。开始时，算法播下一个种子线图自顶向下地预测*S*。这个种子线图的种植是通过在Chart[0]中加入哑状态（dummy state）$\gamma \rightarrow \bullet\ S, [0, 0]$来实现的。当处理这个状态时，算法转入Predictor，造出三个状态来表示对于*S*的每个可能的类型的预测，并逐一造出这些树的所有左角的状态。当处理状态 $VP \rightarrow \bullet\ Verb, [0, 0]$时，调用Scanner并查找第一个单词。这时，代表*book*的动词意义的状态被加入到线图项目Chart[1]中。注意，当处理状态 $VP \rightarrow \bullet\ Verb\ NP, [0, 0]$时，还要再次调用Scanner。但是，

这一次没有必要再加一个新的状态，因为在线图中已经有一个与它等同的状态了。还要注意，由于我们这个语法确实是很不完善的，它不能产生对于 *book* 的名词意义的预测，因此在线图中就不必为此造一个线图项目了。

Chart[0]

$\gamma \rightarrow \bullet S$	[0,0]	Dummy start state
$S \rightarrow \bullet NP\ VP$	[0,0]	Predictor
$NP \rightarrow \bullet Det\ NOMINAL$	[0,0]	Predictor
$NP \rightarrow \bullet Proper\text{-}Noun$	[0,0]	Predictor
$S \rightarrow \bullet Aux\ NP\ VP$	[0,0]	Predictor
$S \rightarrow \bullet VP$	[0,0]	Predictor
$VP \rightarrow \bullet Verb$	[0,0]	Predictor
$VP \rightarrow \bullet Verb\ NP$	[0,0]	Predictor

Chart[1]

$Verb \rightarrow book \bullet$	[0,1]	Scanner
$VP \rightarrow Verb \bullet$	[0,1]	Completer
$S \rightarrow VP \bullet$	[0,1]	Completer
$VP \rightarrow Verb \bullet NP$	[0,1]	Completer
$NP \rightarrow \bullet Det\ NOMINAL$	[1,1]	Predictor
$NP \rightarrow \bullet Proper\text{-}Noun$	[1,1]	Predictor

Chart[2]

$Det \rightarrow that \bullet$	[1,2]	Scanner
$NP \rightarrow Det \bullet NOMINAL$	[1,2]	Completer
$NOMINAL \rightarrow \bullet Noun$	[2,2]	Predictor
$NOMINAL \rightarrow \bullet Noun\ NOMINAL$	[2,2]	Predictor

Chart[3]

$Noun \rightarrow flight \bullet$	[2,3]	Scanner
$NOMINAL \rightarrow Noun \bullet$	[2,3]	Completer
$NOMINAL \rightarrow Noun \bullet NOMINAL$	[2,3]	Completer
$NP \rightarrow Det\ NOMINAL \bullet$	[1,3]	Completer
$VP \rightarrow Verb\ NP \bullet$	[0,3]	Completer
$S \rightarrow VP \bullet$	[0,3]	Completer
$NOMINAL \rightarrow \bullet Noun$	[3,3]	Predictor
$NOMINAL \rightarrow \bullet Noun\ NOMINAL$	[3,3]	Predictor

图 10.17　剖析*Book that flight*时在线图中造出的状态序列。每一项中都要说明状态、它的开始位置和结束位置以及把它放在这个线图中的 Earley 算法的功能

当在 Chart[0]中的所有状态都处理了以后，算法就转移到 Chart[1]，在这里它找到了代表 *book* 的动词意义的状态。由于这个状态中的点处于它的成分右侧，显然这是一个完成的状态，因此调用 Completer。然后，Completer 找到两个前面存在的 *VP* 状态，在输入中的这个位置上预测 *Verb*，复制这些状态并把它们的点向前推进，然后把它们加入到 Chart[1]中。完成的状态对应于一个不及物动词 *VP*，这将导致造出一个表示命令句 *S* 的状态。另外，在及物动词短语中的点后面还有 *NP*，这将导致造出两个状态来预测 *NP*。最后，状态 $NP \rightarrow \bullet\ Det\ Nominal$, [1, 1]引起 Scanner 去查找单词 *that*，并把相应的状态加入到 Chart[2]中。

移动到 Chart[2]时，算法发现代表 *that* 的限定词意义的状态。这个完成状态导致在 Chart[1]预测的 *NP* 状态中把点向前推进一步，并预测各种类型 *Nominal*。其中第一个 *Nominal* 引起最后一次调用 Scanner 去处理单词 *flight*。

移动到Chart[3]时，出现了代表*flight*的状态，这个状态导致一系列快速的Completer操作，分别完成一个*NP*、一个及物的*VP*以及一个*S*。在这个最后的Chart中出现了状态$S \rightarrow VP \bullet, [0, 3]$，这意味着算法已经找到了成功的剖析。

10.4.5 从线图中检索剖析树

刚才描述的Earley算法的这个版本实际上是一个识别器，而不是一个剖析器。在处理完成之后，正确的句子也就离开线图中的状态$S \rightarrow \alpha \bullet, [0, N]$了。遗憾的是，这时还没有办法把句子*S*的结构检索出来。为了把这个算法转变为剖析器，必须能够从线图中把一个一个的剖析都抽取出来。为了做到这一点，每个状态的表示必须再增加一个区域来存储关于生成句子中各个成分所完成的状态的信息。

只要简单修改一下Completer就可以收集到这种信息。我们知道，当状态中的点后面的成分被找到以后，Completer通过推进老的未完成状态的办法，造出了一个新的状态，惟一需要修改的是让Completer给老的状态在新状态的前面状态的表中增加一个指针。当算法从线图检索剖析树时，只要从在最后的线图项目中代表一个完全*S*的那个状态（或一些状态）开始，递归地进行检索，就能把剖析树从线图中检索出来。图10.18说明了使用适当修改过的Completer构造线图的过程。

Chart[0]

S0	$\gamma \rightarrow \bullet S$	[0,0]	[]	Dummy start state
S1	$S \rightarrow \bullet NP\ VP$	[0,0]	[]	Predictor
S2	$NP \rightarrow \bullet Det\ NOMINAL$	[0,0]	[]	Predictor
S3	$NP \rightarrow \bullet Proper\text{-}Noun$	[0,0]	[]	Predictor
S4	$S \rightarrow \bullet Aux\ NP\ VP$	[0,0]	[]	Predictor
S5	$S \rightarrow \bullet VP$	[0,0]	[]	Predictor
S6	$VP \rightarrow \bullet Verb$	[0,0]	[]	Predictor
S7	$VP \rightarrow \bullet Verb\ NP$	[0,0]	[]	Predictor

Chart[1]

S8	$Verb \rightarrow book \bullet$	[0,1]	[]	Scanner
S9	$VP \rightarrow Verb \bullet$	[0,1]	[S8]	Completer
S10	$S \rightarrow VP \bullet$	[0,1]	[S9]	Completer
S11	$VP \rightarrow Verb \bullet NP$	[0,1]	[S8]	Completer
S12	$NP \rightarrow \bullet Det\ NOMINAL$	[1,1]	[]	Predictor
S13	$NP \rightarrow \bullet Proper\text{-}Noun$	[1,1]	[]	Predictor

Chart[2]

S14	$Det \rightarrow that \bullet$	[1,2]	[]	Scanner
S15	$NP \rightarrow Det \bullet NOMINAL$	[1,2]	[S14]	Completer
S16	$NOMINAL \rightarrow \bullet Noun$	[2,2]	[]	Predictor
S17	$NOMINAL \rightarrow \bullet Noun\ NOMINAL$	[2,2]	[]	Predictor

Chart[3]

S18	$Noun \rightarrow flight \bullet$	[2,3]	[]	Scanner
S19	$NOMINAL \rightarrow Noun \bullet$	[2,3]	[S18]	Completer
S20	$NOMINAL \rightarrow Noun \bullet NOMINAL$	[2,3]	[S18]	Completer
S21	$NP \rightarrow Det\ NOMINAL \bullet$	[1,3]	[S14,S19]	Completer
S22	$VP \rightarrow Verb\ NP \bullet$	[0,3]	[S8,S21]	Completer
S23	$S \rightarrow VP \bullet$	[0,3]	[S22]	Completer
S24	$NOMINAL \rightarrow \bullet Noun$	[3,3]	[]	Predictor
S25	$NOMINAL \rightarrow \bullet Noun\ NOMINAL$	[3,3]	[]	Predictor

图10.18 剖析*Book that flight*时在线图中造出的状态序列，状态中包括结构信息

如果对于某个给定的句子，存在着复杂度为指数时间数量的树，Eealey算法不可能像变魔术那样把它们全部转变成复杂度为多项式时间数量的树。最好的办法是让Earley算法构造具有多项式时间复杂度的线图。图 10.19 描绘了这个线图的一部分，这个线图是从图 10.17 使用有向图的概念构成的。注意，由于这种格式的线图太大时会使人眼花缭乱，这个图中只包括那些在最后的剖析中起作用的状态。

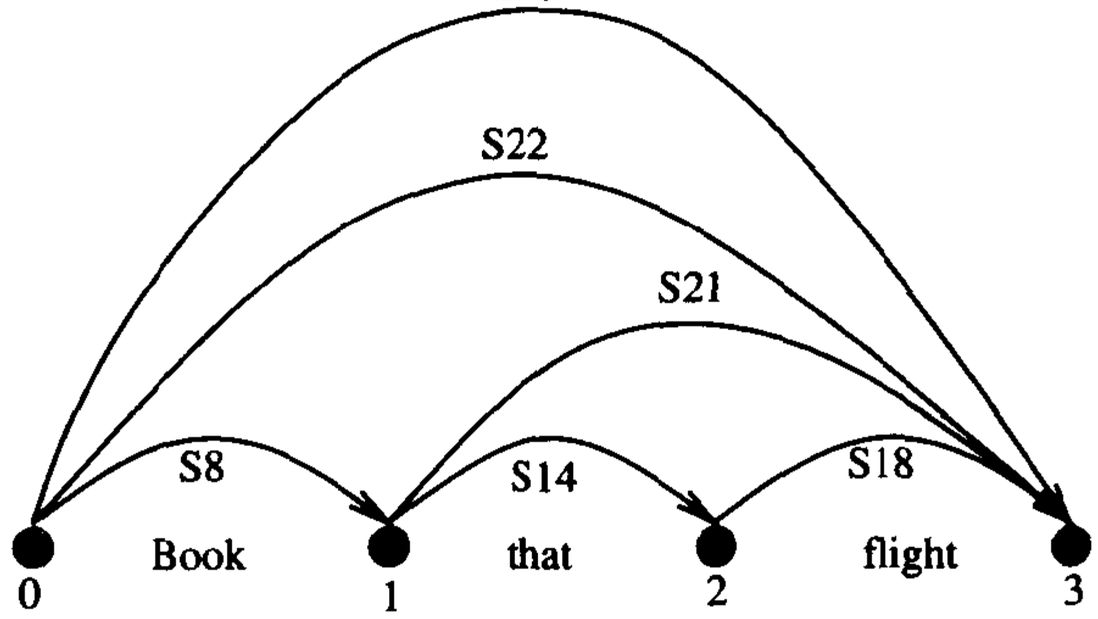


图 10.19　用有向非成圈图表示的图 10.17 所说明的线图的一部分

10.5　有限状态剖析方法

有些自然语言处理的任务并不需要进行完全的剖析。对于这些任务，对输入句子的局部剖析（partial parse）或浅层剖析（shallow parse）已经足够了。例如，信息抽取（information extraction）算法一般不必抽取在文本中所有可能的信息，只要抽取那些可以填充所要求数据的某种模板就足够了。很多局部剖析系统都使用层叠式（cascade）的有限状态自动机来替代上下文无关语法。依靠这样简单的有限状态自动机而不依靠完全的剖析，使得这些系统的效率非常高。因为有限状态系统不能模拟某些递归规则，它们只好以效率来弥补覆盖面的不足。我们将在第 15 章中讨论信息抽取的问题，这里只说明怎样使用有限状态自动机来识别基本短语（basic phrase），如名词短语、动词短语和地点词等。下面是FASTUS基本短语识别器的输出，当然，究竟处理什么样的基本短语依赖于具体的应用要求。

Company Name:	Bridgestone Sports Co.
Verb Group:	said
Noun Group:	Friday
Noun Group:	it
Verb Group:	had set up
Noun Group:	a joint venture
Preposition:	in
Location:	Taiwan
Preposition:	with
Noun Group:	a local concern
Conjunction:	and
Noun Group:	a Japanese trading house
Verb Group:	to produce
Noun Group:	golf clubs
Verb Group:	to be shipped
Preposition:	to
Location:	Japan

这些基本短语是通过编写到转录机中的一些有限状态规则产生出来的。为了具体说明其工作的情况，我们从 Appelt and Israel(1997)的 FASTUS 规则中抽出一个简化的集合，用这些简化的规则来探测名词组（noun group）。名词组是名词短语的核心；一个名词组包含中心名词和左修饰语（限定词、形容词、数量修饰语、数词等）。为了表达上的方便，我们使用带有箭头符号（→）的规则，这样的规则看起来很像上下文无关规则的形式，但在实际情况下，这些规则被编为有限自动机，并没有像上下文无关规则那样来处理。

一个名词组可以只包括一个代词（*she*，*him* 或 *them*），一个时间短语（*yesterday*）或一个日期：

NG → Pronoun | Time-NP | Date-NP

名词组也可以只包含一个单独使用的限定词（*this* 或 *that*），或者包含一个中心名词（*HdNns*），这个中心名词前面可以有随选的限定词短语（*DETP*）和/或随选的形容词（*Adjs*）（*the quick and dirty solution*, *the frustrating mathematics problem*），或者包含一个由动名词短语修饰的中心名词（the rising index）：

NG → (DETP) (Adjs) HdNns | DETP Ving HdNns
| DETP-CP (and HdNns)

上面的括号用来表示随选的成分，同时括号也可以用来表示组合。限定词短语有两个变体：

DETP → DETP-CP | DETP-INCP

完全限定词短语（DETP-CP）能够单独作为 NP 使用，诸如 *only five*, *another three*, *this*, *many*, *hers*, *all* 和 *the most*。*Adv-pre-num* 是那些在限定词短语中可以出现在数词之前的副词（*almost 5*, *precisely 5*）。*Pro-Poss-cp* 是可以单独像一个完全的 *NP* 那样使用的主有代词（*mine* 或 *his*）。数量修饰语（*Quantifiers*，简写为 *Q*）包括 *many*，*few* 和 *much* 等。

DETP-CP → ({ Adv-pre-num | "another" |
{ Det | Pro-Poss } ({Adv-pre-num | "only" ("other")})}) Number
| Q | Q-er | ("the") Q-est | "another" | Det-cp | DetQ | Pro-Poss-cp

非完全限定词短语（DETP-INCP）不能单独作为 *NP* 使用，例如 *the*，*his only*，*every* 和 *a* 等。*Pro-Poss-incomp* 是不能单独作为 *NP* 使用的主有代词（例如 *my* 或 *her*）：

DETP-INCP → { { { Det | Pro-Poss } "only"
| "a" | "an"
| Det-incomp
| Pro-Poss-incomp } ("other")
| (DET-CP) "other"}

形容词序列（Adjs）包含一个或多个形容词或者分词，各个词之间用逗号和/或连接词分开（例如，*big*［大］，*bad*［坏］，*and ugly*［并且丑陋］，*or interesting but outdated*［或者有趣然而过时］）：

Adjs → AdjP ({ "," | (",") Conj } { AdjP | Vparticiple}) *

形容词短语可以由形容词、分词、序数词、名词 - 动词组合（如 *man-eating*）构成，可以被比较级和最高级的数量修饰语（Q-er: *more*, *fewer*; Q-est: *most*, *fewest*）修饰。这一套规则不允许分词处于形容词短语或名词短语的第一个位置，避免不正确地把很多动词 - 宾语组合当成名词组。

AdjP → Ordinal
| ({Q-er | Q-est}){ Adj | Vparticiple } +
| { N[sing,!Time-NP] ("-") { Vparticiple }
| Number ("-") { "month" | "day" | "year"} ("-") "old"}

名词可以连接起来（*cats and dogs*）：

HdNns → HdNn ("and" HdNn)

最后，我们还需要处理名词 - 名词组合以及其他像名词的名词前修饰语，以便覆盖中心名词组，如 *gasoline and oil tanks*，*California wines*，*Clinton* 以及 *quick-reaction strike*：

HdNn → PropN
| { PreNs | PropN PreNs} N[!Time-NP]
| { PropN CommonN[!Time-NP] }

名词前面的的名词修饰语可以连接起来（*gasoline and oil*）或者用短横"-"连接（*quick-reaction*）。*Adj-noun-like* 是指那些能够出现在名词前位置的形容词（例如 *presidential retreat*）：

PreNs → PreN ("and" PreN2) *
preN → (Adj "-") Common-Sing-N
preN2 → PreN | Ordinal | Adj-noun-like

图 10.20 是名词组识别器 *Adjs* 部分的一个 FSA 以及 *AdjP* 部分的一个 FSA。

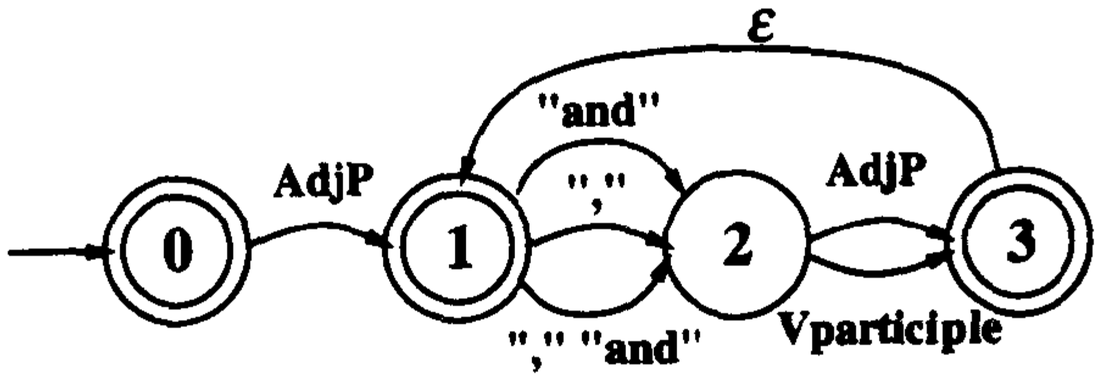


图 10.20　覆盖并列的形容词短语的FSA语法的一部分。在一个实际的自动机中，每个AdjP结点可以复制图10.21所示的AdjP自动机来展开

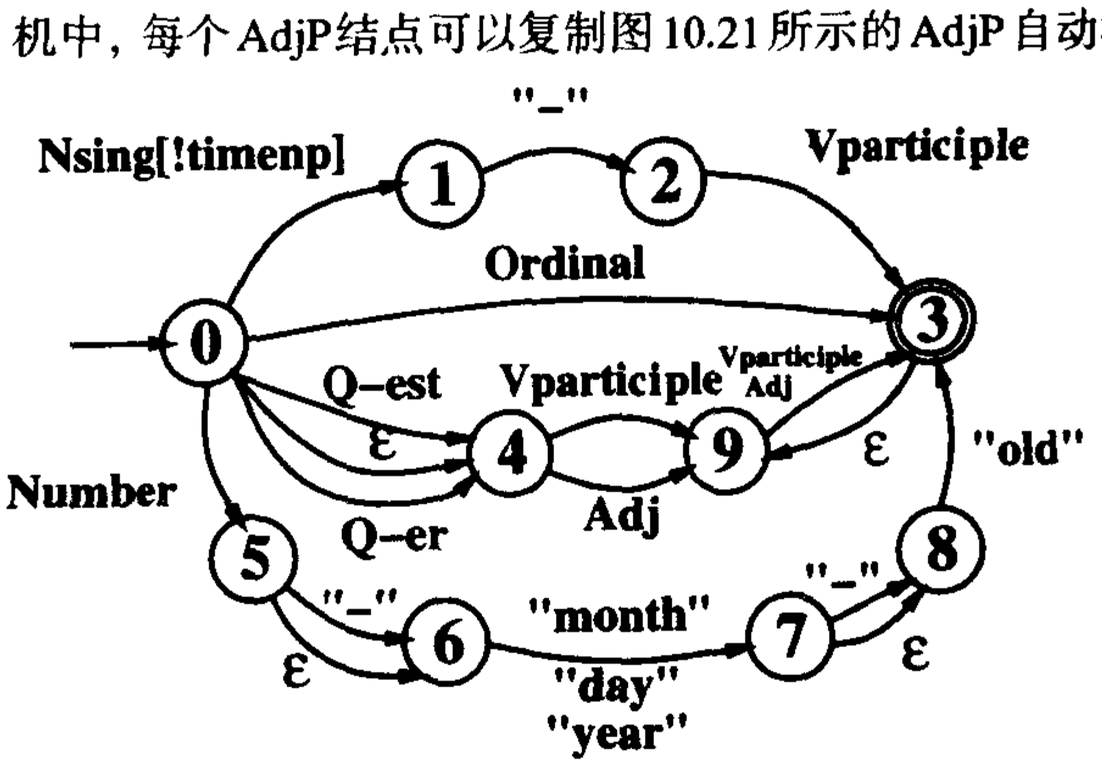


图 10.21　覆盖形容词短语内部组成成分细节的 FSA 语法的 FSA 语法的一部分

图10.20和图10.21中的自动机可以组合成一个单一的更大的名词组识别器。这个识别器以NG自动机开始，迭代地把每个参照展开为其他规则或自动机。因为这些参照中没有一个是递归的，即AdjP的展开不必参照 *AdjP*，所以这样的展开是可能的。

但是，我们在9.11节中曾经说明过，如果建立一个更复杂的英语语法，就会出现这种递归。例如，在关系从句和其他名词后修饰语的规则中，定义一个完整的 *NP* 需要参照其他 *NP*。

处理递归的一个方法是限制递归的数量，只允许有限数量的递归。FASTUS就是这样做的。在FASTUS中使用了层叠式自动机，一层一层地从较小的名词组逐渐构成较大的名词组，这样就限制了递归在每个层次中的数量。FASTUS的第二层发现非递归的名词组；第三层把这些名词组加上度量短语，结合成更大的像 *NP* 那样的单位：

20, 000 iron and "metal wood" clubs a month

然后再附加上介词短语：

production of 20, 000 iron and "metal wood" clubs a month

再处理名词组连接词：

a local concern and a Japanese trading house

在一个单层的系统中，这些现象的每个都分别要求递归规则（例如，*NP* → *NP and NP*）。但是，如果把剖析分解成两个层次，FASTUS就可以把规则左手边的一个*NP*作为不同种类的对象来处理，也就是说，把规则左手边的一个*NP*作为规则右手边的两个*NP*来处理。

处理递归的第二种方法是使用貌似有限状态而实际上并非有限状态的模型。其中的一个模型是递归转移网络（Recursive Transition Network，简称RTN）。一个RTN可以使用如图10.20和图10.21所示的状态图的集合来定义，其中每个弧上有一个终极符号或非终极符号。RTN和FSA的不同之处在于它们处理非终极符号的方法。在RTN中，每当自动机遇到一个带有非终极符号的弧时，它把这个非终极符号作为一个子程序（subroutine）来处理。自动机把它的当前位置放到一个栈中，跳到这个非终极符号。当这个非终极符号剖析结束之后，自动机再跳回原位置。如果一个*NP*规则包含一个自身的参照，RTN再次把当前位置放到栈中，然后再跳回NP的开始处。

由于RTN与上下文无关语法是完全等价的，所以遍历一个RTN就可以想像成是使用一个上下文无关规则来进行简单的自顶向下剖析的过程，而且这个过程可以用图表示出来。在描述或显示一个语法时，通常可以使用RTN作为一个恰当的比喻图示，或者作为实现具有小量递归但又不是有限状态的系统的一种方法。

10.6 小结

本章介绍了很多材料。本章中最重要的两个思想是关于剖析的思想和关于局部剖析的思想。下面是本章中这些思想的要点：

- 剖析可以看成一个搜索问题。
- 两种最常见的搜索是自顶向下搜索（从根开始朝着输入单词向下长出树形图）和自底向上搜索（从单词开始朝着树根*S*向上长出树形图）。
- 一个简单的剖析算法是图10.6中的自顶向下、深度优先、从左向右的剖析器。
- 自顶向下剖析可以使用左角表来优先处理那些与输入匹配的非终极符号，从而提高剖析器的效率。
- 歧义、左递归、子树的重复剖析是这种简单剖析算法的普遍问题。
- 如果语法给一个句子指派一个以上的可能剖析，那么这个句子就是有结构歧义的。
- 常见的结构歧义有PP附着歧义、并列连接歧义、名词短语层次歧义。
- 动态规划剖析算法使用一个局部剖析的表来有效地剖析歧义句子。Earley算法是自顶向下的动态规划算法，而CYK算法则是自底向上的动态规划算法。
- 有些信息抽取问题可以不使用完全剖析来解决，而可以通过层叠式FSA来解决。

10.7 文献和历史说明

Knuth在谈到编译技术的历史时曾经写道："在这个领域，同样的技术被各自独立地工作的人们并行地发现的事情是屡见不鲜的。"

如果多重发现是一个规律，那么这样的事情就不能仅仅说成是“屡见不鲜”的了。不过，尽管有足够的并行发表的算法，为了简明起见，在历史上一般只说明每个算法中最早提及哪一个，这显然是历史的错误。有兴趣的读者请参阅 Aho and Ullman(1972)。

自底向上算法似乎最早是由Yngve(1955)提出的，Yngve提出了一种广度优先、自底向上的算法，并把这种算法作为机器翻译过程的一个部分来描述。用于剖析和翻译的自顶向下方法最早是由Glennie(1960)，Irons(1961)和Kuno and Oetinger(1963)分别提出的（假定他们是各自独立地提出的）。动态规划剖析也有这种多重发现的历史。根据 Martin Kay（私人通信）的意见，以 CYK 算法为核心的动态规划剖析最早是由 John Cocke 于 1960 年实现的。后来，这个算法做了进一步扩充和形式化，并且论证了它的时间复杂性(Kay, 1967; Younger, 1967; Kasami, 1965)。有关良构子串表（well-formed substring table，简称 WFST）的概念，似乎是由 Kuno(1965)独立地提出的，他把良构子串表作为一种数据结构来存储在剖析过程中前面的计算结果。基于对 Cocke 工作的进一步概括，Kay(1967)和 Kay(1973)的论文都独立地描述了类似的数据结构。把动态规划应用于自顶向下的剖析是 Earley 在他的博士论文(Earley, 1968)中以及 Earley(1970)中提出的。Sheil(1976)证明了 WFST 和 Earley 算法的等价性。Norvig(1991)说明，所有这些动态规划算法的效率，在任何语言中都可以使用备忘（memoization）功能（如 LISP 一样）来实现，这时只要给简单的自顶向下剖析增加一个备忘操作即可。

在剖析的早期历史上，曾经普遍使用层叠式的有限状态自动机(Harris, 1962)，后来研究的重点很快就转移到完全CFG剖析方面了。Church(1980)提出，应该回头使用有限状态语法作为自然语言理解的处理模型。早期的其他有限状态剖析模型可参阅 Ejerhed(1988)。Abney(1991)强调，浅层剖析（shallow parsing）在实际应用方面有重要的作用。最近关于浅层剖析的许多工作应用机器学习的方法来研究模式学习问题，有关例子可参阅 Ramshaw and Marcus(1995)，Argamon et al.(1998)和 Munoz et al.(1999)。

关于剖析算法的经典参考资料是 Aho and Ullman(1972)，尽管这本书的重点是计算机语言，但是大多数算法可应用于自然语言。另外，诸如Aho et al.(1986)的优秀的程序语言教材也是很有用的。

第11章　特征与合一

FRIAR FRANCIS: *If either of you know any inward impediment why you should not be conjoined, charge you, on your souls, to utter it.*

William Shakespeare, *Much Ado About Nothing* ①

根据还原主义者（reductionist）的观点，近百年来自然科学发展的历史可以看成是探索如何使用较小基原（primitive）的行为结合起来解释较大结构的行为的历史。在生物学中，遗传的性质用基因的行为来解释，而基因的性质用脱氧核糖核酸（DNA）的行为来解释。在物理学中，物质被还原为原子，而原子又被还原为比原子更小的粒子。在计算语言学中，也逃不出这种还原主义思想的影响。本章我们将介绍诸如*VPto*，*Sthat*，*Non3sgAux*或*3sgNP*这样的语法范畴的思想，还要介绍使用这些范畴构成诸如$S \rightarrow NP\ VP$这样的语法规则的思想，所有这些都可以把客观事物（object）想像成由上述特征关联而成的复杂特征的集合。在这些特征中的信息用约束（constraint）来表示，所以这一类模型通常称为“基于约束的形式化方法”（constraint-based formalism）。

为什么我们需要颗粒度更小的表达并把约束加到语法范畴上去呢？我们在第9章中看到，即使是一致关系和次范畴化这样的语法现象的非常朴素的模型也会导致过度生成（overgeneration）问题。例如，为了避免诸如*this flights*这样的不合语法的名词短语以及诸如*disappeared a flight*这样的不合语法的动词短语，我们不得不使用增殖的方法，创造大量的基本语法范畴，如*Non3sgVPto*，*NPmass*，*3sgNP*和*Non3sgAux*等。反之，这些新的语法范畴又会导致语法规则数目的爆炸性激增，并使这样的语法失去一般性。基于约束的表示策略将使我们能够表示细颗粒的信息，例如，关于数、人称、一致关系、次范畴化的信息，以及诸如“质量/可数”这样的语义范畴信息。

基于约束的形式化方法还有本章没有提到的其他优点。例如，这种形式化方法能够模拟比上下文无关语法更复杂的现象，能够有效地和方便地计算句法表示的语义。

我们现在来简单说明如何使用这种方法来处理语法的数。从第9章中我们知道，*this flight*和*those flights*这样的名词短语可以根据它们是单数还是复数来区分。如果使用称为NUMBER的性质，并使NUMBER具有单数和复数这样的值，把它们与*NP*范畴的适当成员结合起来，就可以实现这样的区分。如果有了这样的能力，就可以说*this flight*是*NP*范畴的一个成员，并且对于NUMBER这个性质它具有单数这样的值。相同的性质也可以按相同的方式应用来区分*VP*范畴的单数成员和复数成员，如*serves lunch*和*serve lunch*。

显而易见，如果只是简单地把这样的性质与各个单词和短语结合起来，并不能解决过度生成的问题。为了使这样的性质变得有用，还需要具有某些运算的能力。例如，检验这些性质是否相等的能力。把这样的检验与我们的核心语法规则配合起来，就能够加上各种各样的约束，以确保语法只能生成一个合语法的符号串。例如，如果需要知道给定的名词短语和动词短语在NUMBER的性质方面是否具有相同的值，就可以进行这样的检验。这样的检验可以用如下形式的规则来表示：

---

① 这是莎士比亚《小题大做》中的一段话：
Friar Francis: 如果你们两人有内心的烦恼，为什么不把它合在一起，加载到你们的心灵上，然后把它吐露出来呢？

$$S \rightarrow NP\ VP$$
仅当 NP 的 NUMBER 与 VP 的 NUMBER 相等

本章将根据特征结构（feature structure）和合一（unification）描述这种基于约束的形式化方法的计算机实现细节。下一节描述特征结构，采用特征结构来表示我们心目中的语法性质的类别。11.2 节介绍合一运算（unification operator），用合一运算来实现对于特征结构的基本操作。11.3 节把这些结构结合起来描写形式化语法。11.4 节介绍合一算法和它所要求的数据结构。11.5 节研究如何把特征结构与合一运算结合起来，应用到剖析中。最后，11.6 节讨论这种基于约束的形式化方法的最有意义的扩充，类型（type）和继承关系（inheritance）的应用以及其他方面的扩充。

11.1　特征结构

把我们心目中的性质的类型进行编码的一种最简单的办法是使用特征结构。特征结构是“特征-值”偶对的简单集合，其中“特征”是从某个有限集合中抽出来的不能再分析的原子符号，“值”或者是原子符号，或者是特征结构。这样的特征结构在传统上用下面的矩阵来表示，这样的矩阵表示称为“特征-值矩阵”（attribute-value matrix，简称 AVM）：

$$\begin{bmatrix} \text{FEATURE}_1 & \text{VALUE}_1 \\ \text{FEATURE}_2 & \text{VALUE}_2 \\ \vdots & \\ \text{FEATURE}_n & \text{VALUE}_n \end{bmatrix}$$

为了具体说明这种 AVM，我们来研究上面讨论过的“数”这个性质。为了描述这个性质，我们使用符号 NUMBER 来表示这个语法属性，用 SG 和 PL 来表示它在英语中能取的值（在第 3 章中介绍过）。由这样的符号串特征组成的特征结构表示如下：

$$\begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix}$$

再增加一个特征-值偶对来表示人称的语法概念，得到如下的特征结构：

$$\begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix}$$

然后，我们还可以使用特征 CAT，给予这个结构相应的成分的语法范畴进行编码。例如，我们可以指出，这些特征是与名词短语 NP 相关联的。特征结构如下：

$$\begin{bmatrix} \text{CAT} & \text{NP} \\ \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix}$$

这个特征结构可以用来表示第9章中介绍的范畴 *3sgNP*，它反映了名词短语的一个受限制的次范畴。与此相对应的复数名词短语的特征结构如下：

$$\begin{bmatrix} \text{CAT} & \text{NP} \\ \text{NUMBER} & \text{PL} \\ \text{PERSON} & 3 \end{bmatrix}$$

注意，在最后的两个特征结构中，特征 CAT 和特征 PERSON 的值是相同的。这说明特征结构可以允许我们保持语法范畴的核心集合，并能区分单个范畴成员之间的不同点。

前面在定义特征结构时说过，特征的值不仅限于原子符号，也可以是其他特征结构。如果我们想把一个特征-值偶对的集合与相似的处理方式捆绑在一起，取特征结构为值是特别有用的。例如，NUMBER 和 PERSON 这两个特征常常捆绑在一起，因为句子中的语法主语总是与谓语在数和人称的性质方面保持一致关系。这种捆绑在一起的情况可以引入 AGREEMENT 这个特征来表示，这个特征取由 NUMBER 和 PERSON 的特征-值偶对组成的特征结构为其值。用这样的特征来表示第三人称单数的名词短语，特征结构如下：

$$\begin{bmatrix} \text{CAT} & \text{NP} \\ \text{AGREEMENT} & \begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \end{bmatrix}$$

做了这样的安排之后，我们就可以通过检验两个成分的 AGREEMENT 特征的相等性来检验两个成分的 NUMBER 和 PERSON 值的相等性。

一个特征可以取特征结构为值的这种能力使我们能够直接导出特征路径（feature path）的概念。特征路径是在特征结构中引导到一个特定值的各个特征所组成的一个表。例如，在最后一个特征结构中，我们可以说，特征路径〈AGREEMENT, NUMBER〉引导到值 SG，而特征路径〈AGREEMENT PERSON〉引导到值3。这个特征路径的概念自然地使我们想到特征结构的图形表示方法，如图 11.1 所示。我们在 11.4 节中将会看到，这样的图形表示方法对于研究它们的具体实现方法很有启发。在这种图形表示中，特征结构被描写为一个有向图，图中，特征标在边上，值标在结点上。

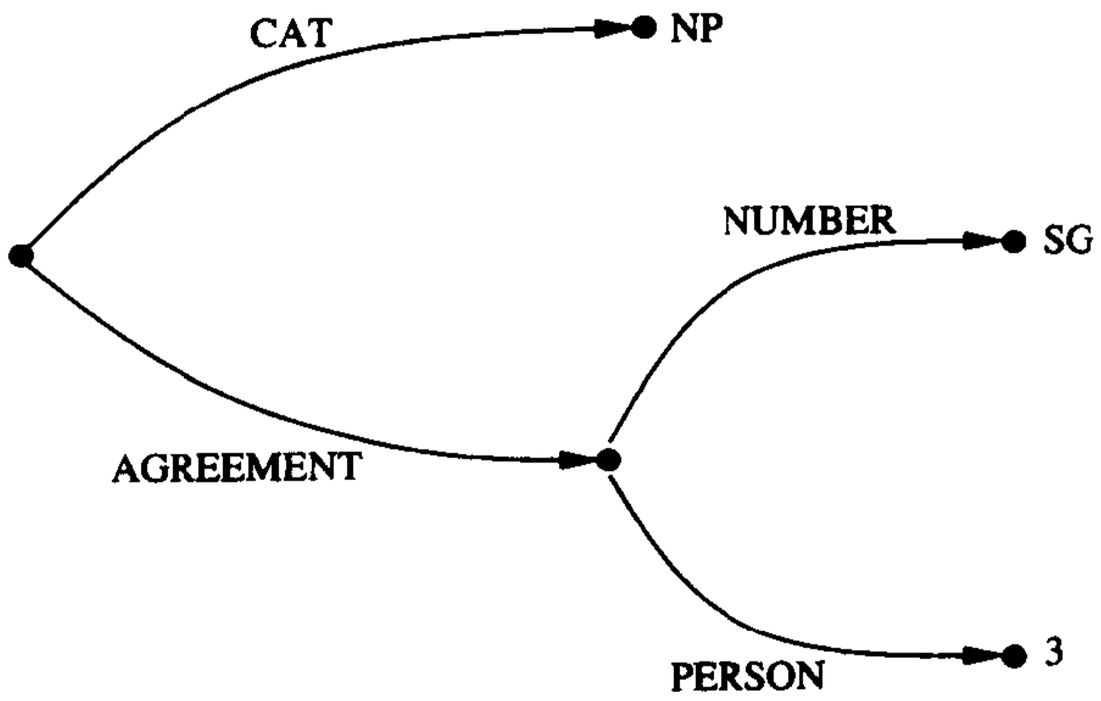


图 11.1 特征结构的有向图表示法

特征路径的概念在很多方面被证明是很有用的，在这里介绍这个概念有助于我们解释一种重要的特征结构，这种特征结构所包含的特征共享某些作为值的特征结构。这样的特征结构称为重入结构（reentrant structure）。现在我们心目中的不是两个特征可能会有相同的值这种简单想法，而是它们正好共享同样的特征结构（或者共享图中的结点）。在简单相等的场合，两条特征路径通向图中的不同结点，而这个图锚定了相同的而不是相异的结构。在重入结构的场合，两条特征路径通向结构上完全相同的结点。

图 11.2 是重入结构的一个简单的例子。在这个结构中，特征路径〈HEAD SUBJECT AGREEMENT〉与特征路径〈HEAD AGREEMENT〉通向同样的位置。在矩阵图示中，共享的结

构加上一个数码索引符号来表示共享的值。图 11.2 中的特征结构的矩阵表示如下，这里采用了 PATR-II 系统中的表示方法(Shieber, 1986)，这种表示方法是根据 Kay(1979)提出的。

$$\begin{bmatrix} \text{CAT} & \text{S} \\ \text{HEAD} & \begin{bmatrix} \text{AGREEMENT} & \boxed{1}\begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \boxed{1} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

我们将会看到，这种简单结构使我们能够以非常精练和漂亮的方式来表示语言中的概括性现象。

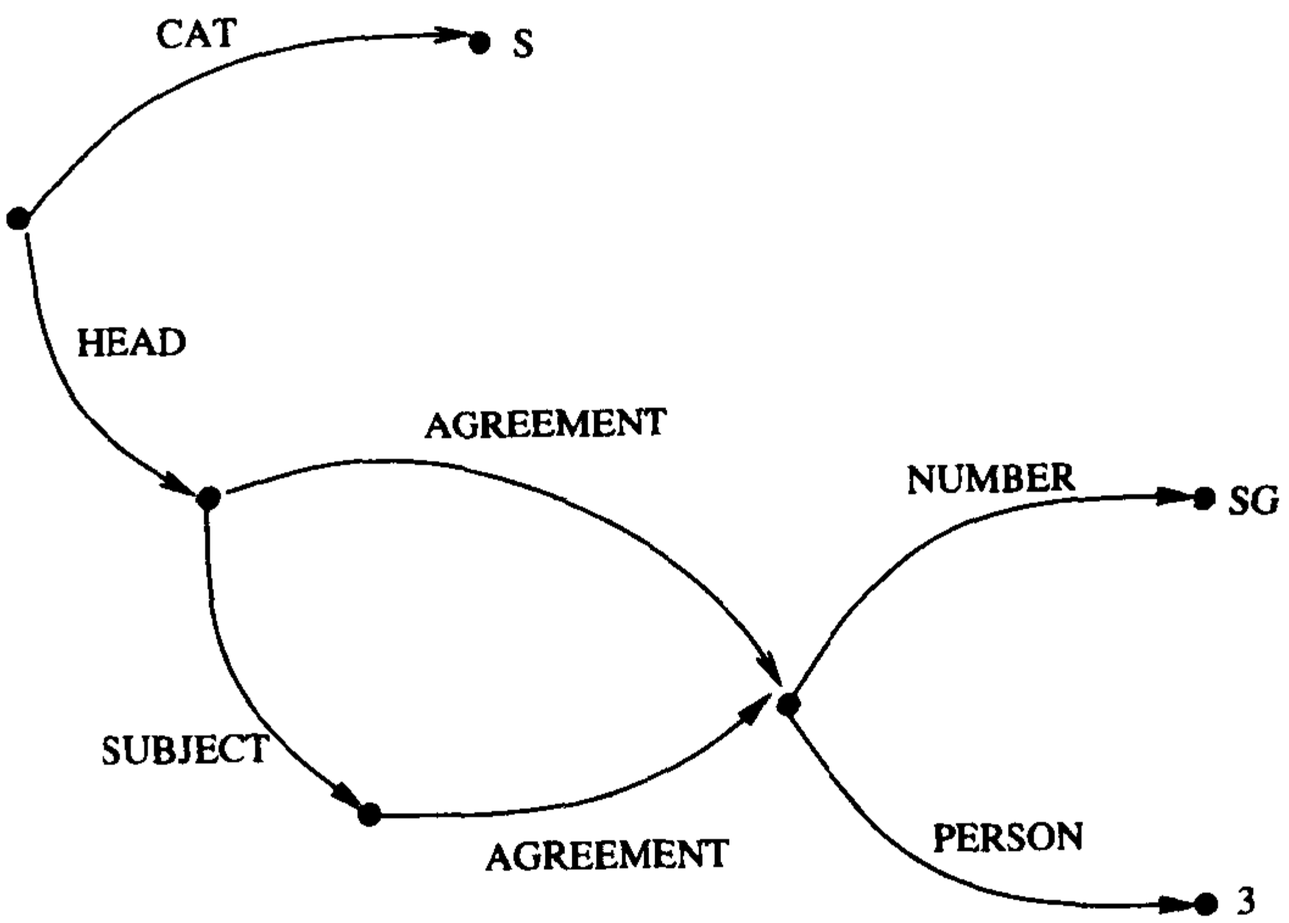


图 11.2　具有共享值的特征结构。顺着路径〈HEAD SUBJECT AGREEMENT〉找到的位置（特征值）与顺着路径〈HEAD AGREEMENT〉找到的位置（特征值）是相同的

11.2　特征结构的合一

我们前面说过，如果只是有特征结构，而不能对它们进行合理有效并且有力的运算，这样的特征结构是没有多少用处的。我们将会看到，为达到这样的目标，需要两种主要操作，一种操作是合并两个结构的信息内容，一种操作是拒绝合并不相容的结构。幸运的是，有一种简单的计算技术可以满足这两个操作的要求，这种运算称为“合一”（unification）。本节的大部分内容将通过一系列实例来说明合一运算是怎样把合并和相容等概念具体化的。对于合一的算法和它的实现方法，将推迟到 11.4 节讨论。

首先，我们从合一算子的简单应用开始。

$$\begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix} \sqcup \begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix} = \begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix}$$

正如这个等式说明的，合一算子是一个二元算子（用⊔表示），它接受两个特征结构作为运算单元，当合一成功时返回一个特征结构作为合一的结果。在这个例子中，合一被用来进行简单的相等关系的检查。因为在每个结构中相应的特征NUMBER以及它们的值是一致的，合一成功。这时，

因为原来的结构是等同的，所以输出与输入相等。下面相似的检查却失败了，因为在两个结构中的特征 NUMBER 具有不相容的值。

$$\begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix} \sqcup \begin{bmatrix}\text{NUMBER} & \text{PL}\end{bmatrix} \textit{ Fails!}$$

下面的合一运算说明了在合一运算中相容概念的一个重要方面。

$$\begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix} \sqcup \begin{bmatrix}\text{NUMBER} & []\end{bmatrix} = \begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix}$$

在这种情况下，特征结构也被看成是相容的，所以就可以进行合并，尽管每个NUMBER特征的给定值不同。在第二个特征结构中的“[ ]”值说明，这个值是没有内容的。带有“[ ]”值的特征可以与另一个结构中的相应特征的任何值匹配。因此，在这个场合中，第一个结构中的值SG可以与第二个结构中的值“[ ]”匹配。正如输出所显示的，这种类型的合一的结果是一个结构，这个结构的值由另一个参与合一的更特殊的非零值来提供。

下面一个例子说明了合一运算中相容概念的另一个方面。

$$\begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix} \sqcup \begin{bmatrix}\text{PERSON} & 3\end{bmatrix} = \begin{bmatrix}\text{NUMBER} & \text{SG} \\ \text{PERSON} & 3\end{bmatrix}$$

这时，合一的结果是把原来的两个结构合并为一个更大的结构。这个更大的结构包含原来每个结构中存储的所有信息。尽管这是一个简单的例子，但是它对于理解为什么这些结构可以判断为相容的这个问题非常重要：这些结构之所以相容，是由于它们当中不包含明显不相容的特征。这些结构中的每个结构所包含的特征-值偶对都是另一个结构所没有的，这样的事实并不会导致合一运算的失败。

现在，我们来研究更复杂的重入结构的合一的各种情况。下面的例子说明了当第一个运算单元中出现重入结构时，相等性检查的复杂情况。

$$\begin{bmatrix}\text{AGREEMENT} & \boxed{1}\begin{bmatrix}\text{NUMBER} & \text{SG} \\ \text{PERSON} & 3\end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix}\text{AGREEMENT} & \boxed{1}\end{bmatrix}\end{bmatrix} \sqcup \begin{bmatrix}\text{SUBJECT} & \begin{bmatrix}\text{AGREEMENT} & \begin{bmatrix}\text{PERSON} & 3 \\ \text{NUMBER} & \text{SG}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

$$= \begin{bmatrix}\text{AGREEMENT} & \boxed{1}\begin{bmatrix}\text{NUMBER} & \text{SG} \\ \text{PERSON} & 3\end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix}\text{AGREEMENT} & \boxed{1}\end{bmatrix}\end{bmatrix}$$

这个例子中的重要成分是在第二个输入结构中的特征SUBJECT。这些特征可以成功地进行合一，因为在第一个运算单元的数码索引 [1] 之后所发现的值与直接出现在第二个运算单元中的值是完全匹配的。注意，在第一个运算单元中的特征 AGREEMENT 的值对于这个合一的成功是没有意义的，因为第二个运算单元的最外一个层级没有 AGREEMENT 特征。它们之间的惟一关系是特征 AGREEMENT 的值与特征 SUBJECT 共享。

下面的例子说明合一运算的复制功能。

$$\begin{bmatrix} \text{AGREEMENT} & \boxed{1} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \boxed{1} \end{bmatrix} \end{bmatrix}$$

$$\sqcup \begin{bmatrix} \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{PERSON} & 3 \\ \text{NUMBER} & \text{SG} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} \text{AGREEMENT} & \boxed{1} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \boxed{1} \begin{bmatrix} \text{PERSON} & 3 \\ \text{NUMBER} & \text{SG} \end{bmatrix} \end{bmatrix} \end{bmatrix} \tag{11.1}$$

这里，特征结构的值是通过第二个运算单元中的〈SUBJECT AGREEMENT〉特征发现的，这个值被复制到第一个运算单元中的相应位置。此外，第一个运算单元中的特征AGREEMENT也接受了一个值，由于数码索引的作用，这个值与特征〈SUBJECT AGREEMENT〉的终点是彼此链接的。

下面的例子说明了实际共享值的特征与只是具有相同值的特征之间的重要区别。

$$\begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix} \end{bmatrix} \end{bmatrix} \tag{11.2}$$

$$\sqcup \begin{bmatrix} \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{PERSON} & 3 \\ \text{NUMBER} & \text{SG} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

在第一个运算单元中，路径〈SUBJECT AGREEMENT〉和路径〈AGREEMENT〉的终点的值是相同的，但并不共享。两个运算单元中的SUBJECT特征进行合一，在结果中增加一个来自第二个运算单元的PERSON信息。但是，由于没有数码索引把AGREEMENT特征和〈SUBJECT AGREEMENT〉特征链接起来，所以这个PERSON信息不会被加到AGREEMENT特征的值中。

最后，我们来研究合一运算失败的例子。

$$\begin{bmatrix} \text{AGREEMENT} & \boxed{1} \begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \boxed{1} \end{bmatrix} \end{bmatrix}$$

$$\sqcup \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{NUMBER} & \text{PL} \\ \text{PERSON} & 3 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Fails!

顺次通过特征进行运算。首先，我们发现在这两个运算单元中的AGREEMENT特征可以成功地匹配。但是，当移动到SUBJECT特征时，我们发现在每个〈SUBJECT AGREEMENT NUMBER〉路径终点的值是不相等的，从而导致合一的失败。

特征结构是用于表示某些语言学实体的局部信息的一种方法，也是用于把信息约束加到可以接受该信息的语言学实体上的一种方法。合一运算可以看成是合并各个特征结构中的信息的一种方法，或者是描述满足两个约束集合的语言学实体的一种方法。从直觉上说，把两个特征结构进行合一，产生一个新的特征结构，这个新的特征结构或者比原来的输入特征结构更特殊（具有更多的信息），或者与原来的输入特征结构等同。我们可以说，不够特殊（更加抽象）的特征结构“蕴涵于”（subsume）等同的或者更特殊的特征结构。“蕴涵于”（subsumption）用算子$\sqsubseteq$表示。特征结构F蕴涵于特征结构G（记为$F \sqsubseteq G$），当且仅当：

1. 对于F中的任何特征x，$F(x) \sqsubseteq G(x)$，这里的$F(x)$表示“特征结构F中的特征x的值”。
2. 对于F中的一切路径p和q，如果有$F(p)=F(q)$，那么对于G中的一切路径p和q，$G(p)=G(q)$。

例如，我们来研究如下的特征结构：

$$\begin{bmatrix} \text{NUMBER} & \text{SG} \end{bmatrix} \tag{11.3}$$

$$\begin{bmatrix} \text{PERSON} & 3 \end{bmatrix} \tag{11.4}$$

$$\begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \tag{11.5}$$

$$\begin{bmatrix} \text{CAT} & \text{VP} \\ \text{AGREEMENT} & \boxed{1} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \boxed{1} \end{bmatrix} \end{bmatrix} \tag{11.6}$$

$$\begin{bmatrix} \text{CAT} & \text{VP} \\ \text{AGREEMENT} & \boxed{1} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \begin{bmatrix} \text{PERSON} & 3 \\ \text{NUMBER} & \text{SG} \end{bmatrix} \end{bmatrix} \end{bmatrix} \tag{11.7}$$

$$\begin{bmatrix} \text{CAT} & \text{VP} \\ \text{AGREEMENT} & \boxed{1} \\ \text{SUBJECT} & \begin{bmatrix} \text{AGREEMENT} & \boxed{1} \begin{bmatrix} \text{PERSON} & 3 \\ \text{NUMBER} & \text{SG} \end{bmatrix} \end{bmatrix} \end{bmatrix} \tag{11.8}$$

下面的“蕴涵于”关系成立：

$$11.3 \sqsubseteq 11.5$$

$$11.4 \sqsubseteq 11.5$$

$$11.6 \sqsubseteq 11.7 \sqsubseteq 11.8$$

“蕴涵于”关系是偏序关系（partial order）；有一些特征结构对彼此之间既不“蕴涵于”，也不“蕴涵”。例如，11.3和11.4就是这样的特征结构对。

$$11.3 \not\sqsubseteq 11.4$$

$$11.4 \not\sqsubseteq 11.3$$

因为每个特征结构都蕴涵空结构[ ]，特征结构之间的关系可以定义为半格（semilattice）。半格通常可以用图形来表示，最普通的特征[ ]处于半格的顶端，特征结构之间的“蕴涵于”关系用它们之间的连线来表示。这样就可以用“蕴涵于”关系的半格来定义合一。给定两个特征结构 F 和 G，合一 $F \sqcup G$ 定义为更特殊的特征结构 H，使得 $F \sqsubseteq H$ 并且 $G \sqsubseteq H$。因为用合一所确定的信息的顺序是半格，所以合一运算就是单调的（monotonic）(Pereira and Shieber, 1984; Rounds and Kasper, 1986; Moshier, 1988)。这意味着，如果一个特征结构的描述是正确的，那么把这个特征结构与其他特征结构合一的结果所形成的新的特征结构仍然满足原来描述的要求。因此，合一运算与顺序无关（order-independent）；给定特征结构的一个集合来进行合一，可以按任意顺序来检查它们，所得到的结果都是相同的。因此，在上面讲述合一运算的最后一个例子中，尽管我们改换不同的方式来首先检查 AGREEMENT 的属性，合一也是注定要失败的。

总之，合一是整合来自不同约束的知识的一种实现方法。给定两个相容的特征结构作为输入，合一能够产生出更特殊的特征结构，并且这个新的特征结构包含了输入中的全部信息。给定两个不相容的特征结构，合一就会失败。

11.3　语法中的特征结构

前面介绍的特征结构和合一运算，为出色地表达那些仅只使用上下文无关语法机制难以表达的语法约束提供了一条有效的途径。因此，下一步将把特征结构和合一运算结合起来，说明如何使用它们来描述语法的途径。为了实现这样的目标，我们要使普通上下文无关语法的规则加以提升，对于规则中的成分都加上特征结构的说明，并且使用适当的合一运算来表达对于这些成分的约束。从语法的角度来说，加上特征结构和合一运算，可以用来实现下面的目标：

- 把复杂特征结构与词典条目和语法范畴的示例联系起来。
- 根据语法成分的组成部分的特征结构，指导如何把这些特征结构组合成更大的语法成分。
- 加强语法结构各部分之间的相容性约束。

我们使用下面的表示方法来说明对于语法的这种提升。提升之后的语法将使我们可能实现上述的目标，这种表示方法参考了在 Shieber(1986)中描述的 PATR-II 系统：

$$\beta_0 \rightarrow \beta_1 \cdots \beta_n$$
（约束的集合）

具体的约束具有如下的形式：

$$\langle \beta_i \ feature\ path \rangle = Atomic\ value$$
$$\langle \beta_i \ feature\ path \rangle = \langle \beta_j \ feature\ path \rangle$$

$\langle \beta_i\ feature\ path \rangle$ 记法表示通过与规则中的上下文无关部分的组成成分 β_i 相关联的特征结构的一条特征路径。约束的第一种形式说明，在给定路径终点发现的值必须与特定的原子值进行合一。约束的第二种形式说明，在两条给定的路径的终点发现的值必须是可以合一的。

为了说明这些约束的用法，再次讨论本章开始时提出的关于数的一致关系问题的非形式解决办法。当时是这样描述的：

$$S \rightarrow NP\ VP$$

仅当NP的NUMBER与VP的NUMBER相等

现在使用新的记法，把这个规则表示如下：

$$S \rightarrow NP\ VP$$

$$\langle NP\ \text{NUMBER}\rangle = \langle VP\ \text{NUMBER}\rangle$$

注意，如果在规则中有两个或多个成分具有同一个句法范畴，我们就给相应的成分加下标，并按顺序加以排列，如规则 $VP \rightarrow V\ NP_1\ NP_2$。

除了记法之外，重要的是应该注意，在这样的方法中，上下文无关语法的简单的生成特性由于这样的提升而发生了根本性的变化。普通的上下文无关语法基于“毗连”（concatenation）的简单概念，一个*NP*的后面跟着*VP*就是一个*S*。或者，用生成的概念来说，要生成一个*S*，所要做的一切就是把一个*NP*和一个*VP*毗连起来。在新的方法中，这种毗连必须伴随着成功的合一运算才能实现。这种情况自然就会导致合一运算的计算复杂性问题，以及合一运算对于这种新语法的生成能力的影响问题。我们将在第13章中讨论这些问题。

总之，这种新方法的基本点有两个：

- 上下文无关语法规则的成分增加了与它们相关的基于特征的约束。这反映了从原子式的单纯语法范畴到表示该成分的各种性质的更复杂范畴的转移。
- 与单个规则相联系的约束可以参照并处理与带有这些约束的规则部分相联系的特征结构。

下面几节将介绍合一约束在处理四种有趣的语言现象时的应用。这四种语言现象是：一致关系（agreement）、语法中心语（grammatical head）、次范畴化（subcategorization）和长距离依存关系（long-distancedependency）。

11.3.1 一致关系

我们在第9章中讨论过，英语中的不同地方都存在一致关系。本节介绍怎样使用合一来处理英语中的两种主要一致现象：主语-动词一致关系和限定词-名词性成分一致关系。我们在讨论中自始至终都使用如下的ATIS中的句子来说明这些现象：

This flight serves breakfast. (11.9)

Does this flight serve breakfast? (11.10)

Do these flights serve breakfast? (11.11)

注意，上面给出的用于说明主语-动词一致关系的约束的不足之处在于这种约束忽略了PERSON这个特征。下面的使用特征AGREEMENT的约束注意到了这个问题。

$$S \rightarrow NP\ VP$$

$$\langle NP\ \text{AGREEMENT}\rangle = \langle VP\ \text{AGREEMENT}\rangle$$

例句（11.10）和例句（11.11）中表示的主语-动词一致关系与例句（11.9）有些微小的差别。在这些yes-no疑问句中，主语*NP*必须与助动词一致，而不是与句子的主要动词一致，这时主要动词是以不定形式出现的。这种一致约束可以用如下规则处理：

$$S \rightarrow Aux\ NP\ VP$$

$$\langle Aux\ \text{AGREEMENT}\rangle = \langle NP\ \text{AGREEMENT}\rangle$$

在名词短语中，限定词和名词性成分之间的一致关系可以用相似的办法来处理。这种办法的基本任务是允许出现上面给定的形式，防止出现 **this flights* 和 **those flight* 这样的错误形式，其中限定词与名词性成分在数的特征上出现冲突。把这样的约束加到语法规则中，并把它们组合到一起即可构成一个完整的规则：

$$NP \rightarrow Det\ Nominal$$
$$\langle Det\ \text{AGREEMENT}\rangle = \langle Nominal\ \text{AGREEMENT}\rangle$$
$$\langle NP\ \text{AGREEMENT}\rangle = \langle Nominal\ \text{AGREEMENT}\rangle$$

这个规则的意思是，*Det* 的 AGREEMENT 特征必须与 *Nominal* 的 AGREEMENT 特征合一，并且，*NP* 的 AGREEMENT 特征也受到了约束，它必须与 *Nominal* 的 AGREEMENT 特征相同。

上面描述了加在主语 – 动词之间的一致关系以及限定词和名词性成分之间的一致关系上的约束，现在来研究使这样的约束能够发挥作用的机制。具体地说，我们必须考虑参与这种约束(*Aux*, *VP*, *NP*, *Det*, *Nominal*)的各个成分对于各种一致特征要求什么样的值。

首先，我们注意到这些约束既涉及到词汇成分，也涉及到非词汇成分。诸如 *Aux* 和 *Det* 这样的比较简单的词汇成分，可以直接从词典中得到它们各自的一致关系特征。如下面的规则所示：

$$Aux \rightarrow do$$
$$\langle Aux\ \text{AGREEMENT NUMBER}\rangle = \text{PL}$$
$$\langle Aux\ \text{AGREEMENT PERSON}\rangle = 3$$

$$Aux \rightarrow does$$
$$\langle S\ \text{AGREEMENT NUMBER}\rangle = \text{SG}$$
$$\langle S\ \text{AGREEMENT PERSON}\rangle = 3$$

$$Determiner \rightarrow this$$
$$\langle Determiner\ \text{AGREEMENT NUMBER}\rangle = \text{SG}$$
$$Determiner \rightarrow these$$
$$\langle Determiner\ \text{AGREEMENT NUMBER}\rangle = \text{PL}$$

我们再回到第一个 *S* 规则，首先研究非词汇成分 *VP* 的 AGREEMENT 特征。这个 *VP* 的成分结构可以用下面的规则来描述：

$$VP \rightarrow Verb\ NP$$

从规则可以看出，对于这个非词汇成分 *VP* 的一致关系的约束，必须以词汇成分 *Verb* 为基础。而从前面关于词汇成分的条目可以看出，这个 *Verb* 所要求的一致关系特征直接来自词典，如下面的规则所示：

$$Verb \rightarrow serve$$
$$\langle Verb\ \text{AGREEMENT NUMBER}\rangle = \text{PL}$$

$$Verb \rightarrow serves$$
$$\langle Verb\ \text{AGREEMENT NUMBER}\rangle = \text{SG}$$
$$\langle Verb\ \text{AGREEMENT PERSON}\rangle = 3$$

这种情况说明，父结点*VP*的一致关系特征所受的约束与它的动词成分所受的约束是相同的：

$$VP \rightarrow Verb\ NP$$

$$\langle VP\ \text{AGREEMENT}\rangle = \langle Verb\ \text{AGREEMENT}\rangle$$

换言之，非词汇语法成分至少可以从它们的组成成分获得某些特征的值。

同样的方法对于*NP*和*Nominal*范畴也是适用的。这些非词汇范畴的一致关系特征也可以从名词*flight*和*flights*的特征推导出来：

$$Noun \rightarrow flight$$

$$\langle Noun\ \text{AGREEMENT NUMBER}\rangle = \text{SG}$$

$$Noun \rightarrow flights$$

$$\langle Noun\ \text{AGREEMENT NUMBER}\rangle = \text{PL}$$

与此类似，非词汇成分*Nominal*的特征所受的约束与它的组成成分Noun具有相同的特征值：

$$Nominal \rightarrow Noun$$

$$\langle Nominal\ \text{AGREEMENT}\rangle = \langle Noun\ \text{AGREEMENT}\rangle$$

注意，本节的论述仅仅涉及英语一致关系系统的表层，是很粗略的，其他语言的一致关系系统可能会比英语中的情况复杂得多。

11.3.2 中心语特征

为了对于诸如名词短语、名词性成分（nominal）和动词短语这样的组成性语法成分的一致关系特征的构成方式做出解释，我们在前面引入了把所需的特征结构从子女结点复制（copy）到其父结点上的概念。这样的复制是基于约束的语法的更具概括性的现象的一个实例。具体地说，大多数语法范畴的特征都是从它们的子女结点复制到父结点上的。贡献特征的子女结点称为短语的中心语（head of the phrase），而被复制的特征称为中心语特征（head feature）。

为了理解这一点，我们来研究上面提到的三个规则：

$$VP \rightarrow Verb\ NP$$

$$\langle VP\ \text{AGREEMENT}\rangle = \langle Verb\ \text{AGREEMENT}\rangle$$

$$NP \rightarrow Det\ Nominal$$

$$\langle Det\ \text{AGREEMENT}\rangle = \langle Nominal\ \text{AGREEMENT}\rangle$$

$$\langle NP\ \text{AGREEMENT}\rangle = \langle Nominal\ \text{AGREEMENT}\rangle$$

$$Nominal \rightarrow Noun$$

$$\langle Nominal\ \text{AGREEMENT}\rangle = \langle Noun\ \text{AGREEMENT}\rangle$$

在这些规则中，把一致关系特征结构向上贡献给父结点的成分就是短语的中心语。更具体地说，动词是动词短语的中心语，名词性成分是名词短语的中心语，名词是名词性短语的中心语。此外，我们还可以说，一致关系特征结构是一个中心语特征。我们可以重新写规则，使其能够反映出上述的概括性。重写时把一致关系特征结构放到HEAD特征的下面，然后向上复制这个特征，其约束情况如下：

$$VP \rightarrow Verb\ NP$$

$$\langle VP\ \text{HEAD}\rangle = \langle Verb\ \text{HEAD}\rangle \tag{11.12}$$

$$NP \rightarrow Det\ Nominal$$
$$\langle NP\ \text{HEAD}\rangle = \langle Nominal\ \text{HEAD}\rangle$$
$$\langle Det\ \text{HEAD AGREEMENT}\rangle = \langle Nominal\ \text{HEAD AGREEMENT}\rangle \tag{11.13}$$

$$Nominal \rightarrow Noun$$
$$\langle Nominal\ \text{HEAD}\rangle = \langle Noun\ \text{HEAD}\rangle \tag{11.14}$$

与此类似，引入这些特征的词汇规则也必须反映这个 HEAD 的概念，规则重写如下：

$$Noun \rightarrow flights$$
$$\langle Noun\ \text{HEAD AGREEMENT NUMBER}\rangle = \text{PL}$$

$$Verb \rightarrow serves$$
$$\langle Verb\ \text{HEAD AGREEMENT NUMBER}\rangle = \text{SG}$$
$$\langle Verb\ \text{HEAD AGREEMENT PERSON}\rangle = 3$$

中心语的概念在语法中是非常有意义的，因为它提供了一种把句法规则与特定的单词联系起来的途径。由于中心语能够提供这样的途径，所以它在第 12 章将介绍的依存语法（dependency grammar）和词汇化语法（lexicalized grammar）以及在第 21 章将介绍的中心语转录机（head transducer）中，都起着十分重要的作用。

11.3.3　次范畴化

我们前面说过，次范畴化（subcategorization）这个概念是指动词对于与它一起出现的论元模式的严格要求。在第9章中，为了防止生成那些与动词和动词短语不匹配的不合语法的句子，我们不得不把动词的范畴分解成多个次范畴（sub-category）。然后，应用这些更特殊的动词范畴来定义允许与它们共同出现的特殊动词短语。如下面的规则所示：

$$Verb\text{-}with\text{-}S\text{-}comp \rightarrow think$$
$$VP \rightarrow Verb\text{-}with\text{-}S\text{-}comp\ S$$

显而易见，这样的方法将导致我们所不期望的范畴的大量增长，而解决各种问题的办法却相差不大。为了避免范畴的大量增长，可以导入特征结构来区分各种不同的动词范畴。把一个称为 SUBCAT 的原子特征与适当的值和词典中的每个动词结合起来，就可以实现这个目标。例如，在词典中可以给及物动词 *serves* 指派如下的特征结构：

$$Verb \rightarrow serves$$
$$\langle Verb\ \text{HEAD AGREEMENT NUMBER}\rangle = \text{SG}$$
$$\langle Verb\ \text{HEAD SUBCAT}\rangle = \text{TRANS}$$

这个特征 SUBCAT 说明，这个动词在动词短语中出现时，在语法上要求一个单独的名词短语作为它的论元。这样的约束还可以加上语法中所有动词短语规则的相应约束来进一步强化，这些约束如下：

$$VP \rightarrow Verb$$
$$\langle VP\ \text{HEAD}\rangle = \langle Verb\ \text{HEAD}\rangle$$
$$\langle VP\ \text{HEAD SUBCAT}\rangle = \text{INTRANS}$$

$$VP \rightarrow Verb\ NP$$
$$\langle VP\ \text{HEAD}\rangle = \langle Verb\ \text{HEAD}\rangle$$

$$\langle VP \text{ HEAD SUBCAT} \rangle = \text{TRANS}$$

$$VP \rightarrow Verb\ NP\ NP$$
$$\langle VP \text{ HEAD} \rangle = \langle Verb \text{ HEAD} \rangle$$
$$\langle VP \text{ HEAD SUBCAT} \rangle = \text{DITRANS}$$

在这些规则中的第一个合一约束，说明动词短语从它的动词组成成分中获得 HEAD 特征，而第二个约束说明，这个 SUBCAT 的值应该是什么。任何要求动词与一个不适合的动词短语一起使用的企图都是注定要失败的，因为这时*VP*的SUBCAT特征的值都不能与在第二个约束中给定的原子符号进行合一。注意，这种方法要求英语的 50~100 个动词短语中的每个都有一个特定的符号。

这个方法有点儿不透明，因为这些不可分析的SUBCAT符号，既不能对于动词期望采用的论元的数目直接进行编码，也不能对于动词期望采用的论元的类型直接进行编码。为了看清楚这一点，需要注意的是，不能以为只要简单地检查词典中的动词条目，就可以知道它的次范畴化框架是什么。必须间接地使用 SUBCAT 特征的值，把它作为一个指针，指向在语法上能够接受该动词的动词短语规则。

一个比较漂亮的解决方法是更好地利用特征结构的表达能力，允许动词条目能够直接说明它们所要求的论元的顺序和范畴。下面关于 *serves* 的条目是使用这种方法的一个例子。在这个条目中，动词的次范畴特征用它的宾语和补语的一个表（list）来表示。

$$Verb \rightarrow serves$$
$$\langle Verb \text{ HEAD AGREEMENT NUMBER} \rangle = \text{SG}$$
$$\langle Verb \text{ HEAD SUBCAT FIRST CAT} \rangle = \text{NP}$$
$$\langle Verb \text{ HEAD SUBCAT SECOND} \rangle = \text{END}$$

这个条目使用了FIRST这个特征来表示动词后的第一个论元必须是一个*NP*，又使用SECOND这个特征来表示这个动词只要求一个论元。像 *leave* 这样的动词（在 *leave Boston in the morning* 中）具有两个论元，它的条目如下：

$$Verb \rightarrow leaves$$
$$\langle Verb \text{ HEAD AGREEMENT NUMBER} \rangle = \text{SG}$$
$$\langle Verb \text{ HEAD SUBCAT FIRST CAT} \rangle = \text{NP}$$
$$\langle Verb \text{ HEAD SUBCAT SECOND CAT} \rangle = \text{PP}$$
$$\langle Verb \text{ HEAD SUBCAT THIRD} \rangle = \text{END}$$

当然，使用这样的方法对一个表进行编码显得太奢侈了；我们也可以使用 11.6 节中定义的类型（type）来定义一个类型表，这样会更清楚一些。

现在，单独的动词短语规则必须检查动词所要求的成分是不是都准确地出现了。下面是关于动词及物性的规则：

$$VP \rightarrow Verb\ NP$$
$$\langle VP \text{ HEAD} \rangle = \langle Verb \text{ HEAD} \rangle$$
$$\langle VP \text{ HEAD SUBCAT FIRST CAT} \rangle = \langle NP \text{ CAT} \rangle \qquad (11.15)$$
$$\langle VP \text{ HEAD SUBCAT SECOND} \rangle = \text{END}$$

在这个规则约束中的第二个约束说明，动词的SUBCAT表中的第一个成分必须与直接处于动词之后的成分的范畴相匹配。第三个约束说明这个动词短语规则只要求一个单独的论元。

前面的例子只代表了一些非常简单的动词的次范畴化结构。事实上，动词的次范畴化可表示为非常复杂的次范畴化框架（例如*NP PP*，*NP NP*或者*NP S*），并且这些框架可以成为很多不同的短语类型的组成部分。为了给英语的动词构造可能的次范畴化框架表，首先需要列出可以构成这些框架的可能的短语类型的表。图11.3是英语中构成动词次范畴化框架的一个可能的短语类型的简单的表。这个表是根据FrameNet课题(Johnson, 1999; Baker et al., 1998)中的动词次范畴化框架修改而成的，表中包括诸如*there*和*it*这样的动词特殊主语的短语类型，以及宾语和补语的短语类型。

Noun Phrase Types		
There	nonreferential there	**There** *is still much to learn*
It	nonreferential it	**It** *was evident that my ideas*
NP	noun phrase	*As he was relating* **his story**
Preposition Phrase Types		
PP	preposition phrase	*couch their message* **in terms**
PPing	gerundive PP	*censured him* **for not having intervened**
PPpart	particle	*turn it* **off**
Verb Phrase Types		
VPbrst	bare stem VP	*she could* **discuss it**
VPto	to-marked infin. VP	*Why do you want* **to know***?*
VPwh	wh-VP	*it is worth considering* **how to write**
VPing	gerundive VP	*I would consider* **using it**
Complement Clause types		
Finite Clause		
Sfin	finite clause	*maintain* **that the situation was unsatisfactory**
Swh	wh-clause	*it tells us* **where we are**
Sif	whether/if clause	*ask* **whether Aristophanes is depicting a**
Nonfinite Clause		
Sing	gerundive clause	*see* **some attention being given**
Sto	to-marked clause	*know* **themselves to be relatively unhealthy**
Sforto	for-to clause	*She was waiting* **for him to make some reply**
Sbrst	bare stem clause	*commanded* **that his sermons be published**
Other Types		
AjP	adjective phrase	*thought it* **possible**
Quo	quotes	*asked* **"What was it like?"**

图 11.3　可以构成动词的潜在次范畴化框架的潜在短语类型集合。根据FrameNet的标记集修改而成(Johnson, 1999; Baker et al., 1998)。样本句子的片断来自英国国家语料库（British National Corpus，简称BNC）

为了在合一语法中使用图 11.3 所示的短语类型，每个短语类型都要使用特征来描述。例如，**VPto**这个短语类型形式在动词*want*次范畴化时可以表示如下：

$$\begin{aligned} &\textit{Verb} \rightarrow \textit{want} \\ &\quad \langle \textit{Verb}\ \text{HEAD SUBCAT FIRST CAT} \rangle = \text{VP} \\ &\quad \langle \textit{Verb}\ \text{HEAD SUBCAT FIRST FORM} \rangle = \text{INFINITIVE} \end{aligned}$$

英语中的50~100个动词的可能的次范畴化框架将被描述为短语类型的集合，而所有的短语类型都来自上述的这些短语类型。我们以带有两个补语的动词*want*为例来说明这里存在着两种不同的可能标记方法。第一种可能方法是采用带尖括号"〈"和"〉"的表来标记，第二种可能的方法是不使用带路径等式的重写规则来标记，而把词汇条目表示为一个单独的特征结构：

$$\begin{bmatrix} \text{ORTH} & \text{WANT} \\ \text{CAT} & \text{VERB} \\ \text{HEAD} & \begin{bmatrix} \text{SUBCAT} & \langle \boxed{\text{CAT NP}}, \begin{bmatrix} \text{CAT VP} \\ \text{HEAD} \boxed{\text{VFORM INFINITIVE}} \end{bmatrix} \rangle \end{bmatrix} \end{bmatrix}$$

即使我们把短语类型的一个有限集合组合起来，其结果也会形成一个非常庞大的可能的次范畴化框架集合。而且，每个动词还可能具有不同的次范畴化框架。例如，下面是动词*ask*的次范畴化框架的一部分，例句来自BNC语料库：

次范畴	例句
Quo	asked [$_{Quo}$ "What was it like?"]
NP	asking [$_{NP}$ a question]
Swh	asked [$_{Swh}$ what trades you're interested in]
Sto	ask [$_{Sto}$ him to tell you]
PP	that means asking [$_{PP}$ at home]
Vto	asked [$_{Vto}$ to see a girl called Evelyn]
NP Sif	asked [$_{NP}$ him] [$_{Sif}$ whether he could make]
NP NP	asked [$_{NP}$ myself] [$_{NP}$ a question]
NP Swh	asked [$_{NP}$ him] [$_{Swh}$ why he took time off]

现在已经有一些具有相当规模的次范畴化框架的标记集，例如COMLEX标记集(Macleod et al., 1998)，这个标记集包括动词、形容词和名词的次范畴化框架；又如ACQUILEX标记集(Sanfilippo, 1993)，这个标记集包括动词的次范畴化框架。很多次范畴化的标记集还增加了补语的其他信息，诸如关于在没有明显主语的下层动词短语中主语的等同性信息，这种信息称为控制信息（control information）。例如，句子*Temmy promised Ruth to go*（至少在某些方言中可以这样说）的含义是*Temmy*将做*going*（去）这个动作，而句子*Temmy persuaded Ruth to go*的含义则是Ruth将做*going*（去）这个动作。一个动词可能具有的若干个次范畴化框架可以部分地通过动词本身的语义预测出来，例如，对于很多具有及物语义的动词（*give*, *send*, *carry*），可以预示它们可能具有*NP NP*和*NP PP*两个次范畴化框架：

NP NP　sent FAA Administrator James Busey a letter

NP PP　sent a letter to the chairman of the Armed Services Committee

动词的各个类的次范畴化框架之间的这些关系称为“论元结构交替”（argument-structure alternation）。在第16章中讨论动词论元结构的语义时将进一步研究这个问题。在动词所允许的不同次范畴化框架之间，一般都存在优先关系（preference）。第12章将介绍模拟这种优先关系的概率方法。

11.3.4 其他词类的次范畴化

次范畴化的概念通常称为配价（valence），这个概念本来是为研究动词提出来的，但是最近发现动词之外的很多其他词类也表现出与配架相似的行为。我们来对比介词*while*和*during*的如下不同用法：

Keep your seatbelt fastened while we are taking off.　(11.16)

*Keep your seatbelt fastened *while takeoff.　(11.17)

Keep your seatbelt fastened during takeoff. (11.18)

*Keep your seatbelt fastened during we are taking off. (11.19)

尽管这些单词看起来似乎是相似的，但它们对于各自的论元的要求却有很大不同。作为练习，请读者把具有不同次范畴化框架的这两个介词的特征结构表示出来。

很多形容词和名词也具有次范畴化框架。这里是形容词*apparent*, *aware*, *unimportant*以及名词*assumption*, *question*次范畴化框架的一些例子。

It was **apparent** [$_{Sfin}$ that the kitchen was the only room. . .]
It was **apparent** [$_{PP}$from the way she rested her hand over his]
aware [$_{Sfin}$ he may have caused offense]
it is **unimportant** [$_{Swheth}$ whether only a little bit is accepted]
the **assumption** [$_{Sfin}$ that wasteful methods have been employed]
the **question** [$_{Swheth}$ whether the authorities might have decided]

请参看 Macleod et al.(1998)和 Johnson(1999)关于名词和形容词的次范畴化框架的描述。

动词要表示出对于它们的主语以及补语的次范畴化约束。例如，我们要表示动词*seem*能够以Sfin作为主语的词汇事实（*That she was affected seems obvious*），而动词*paint*却不能。我们可以用SUBJECT这个特征来表示这样的限制。

11.3.5　长距离依存关系

迄今为止，我们研制的次范畴化模型有两个组成部分。每个中心语都有一个SUBCAT特征，这个特征包括一个由该中心语所期望的补语组成的表。然后像规则（11.16）这样的*VP*规则把实际的组成成分与SUBCAT表中所期望的补语相匹配。当动词的补语与在动词短语中所发现的事实一致时，这样的机制会工作得相当好。

但是，有时动词次范畴化的成分在局部范围内并不出现，它们与谓语之间保持着一种长距离关系。这里是这种长距离依存关系（long-distance dependency）的一些例子：

What cities does Continental service?
What flight do you have from Boston to Baltimore?
What time does that flight leave Atlanta?

在第一个例子中，*what cities*这个成分是动词*service*次范畴化的补语，但是由于这个句子是*wh*非主语疑问式结构，宾语要放在句子前面。从第9章中我们知道，*wh*非主语疑问式结构的（简单）短语结构规则如下：

$$S \rightarrow \textit{wh-NP Aux NP VP}$$

现在我们有了特征，可以增强这个短语结构规则，要求*Aux*与*NP*保持一致关系（因为*NP*是主语）。但是，我们也需要某种办法来增强规则，使规则能够将*wh-NP*填充到*VP*的某个次范畴化的槽（slot）中。这种长距离依存关系的表示是一个非常困难的问题，因为动词的次范畴化要求填充的成分与填充位置之间的距离可能会相当长。在下面的例子中，*wh-*短语*which flight*要填充到动词*book*次范畴化要求的位置，而它们之间隔着两个动词（*want*和*have*）之遥，相距甚远。

Which flight do you want me to have the travel agent book?

在合一语法中提出了表示长距离依存关系的不少解决办法，其中一个办法是使用间隔表（gap list），这种间隔表体现为特征 GAP。在剖析树中，特征 GAP 从一个短语转移到另一个短语，从而处理长距离依存关系的问题。对于这种策略的详细解释，可以参阅 Sag and Wasow(1999)，其中也讨论了给这种长距离依存关系建模的很多其他复杂问题。

11.4 合一的实现

前面讨论过，合一算子取两个特征结构作为输入，如果合一成功则返回一个合并后的单独特征结构，如果两个输入不相容则宣告失败。输入的特征结构用非成圈有向图（DAG）来表示。在DAG中，特征作为标记记录在有向的边上，特征值或者是原子符号，或者是DAG。我们知道，这种合一运算的实现是一个相对简单的递归图匹配算法，但是要对这种算法加以适当裁剪，使之满足合一的各种要求。粗略地说，算法要把一个输入中的特征都走一遍，试着发现在另一个输入中相匹配的特征。如果所有特征都匹配，则合一成功。只要有一个特征不匹配，则合一失败。显而易见，为了正确地对那些以特征结构为其值的特征进行匹配，需要使用递归。

这种算法的一个不寻常的方面在于，它不是使用由两个参与合一项目的全部信息合一而成的信息来构造新的输出特征结构，而是破坏性地改变这些项目，使得最后它们恰好指向相同的信息。因此，成功地调用合一运算的结果也必定要适当地改变它的项目（失败的合一也会引起项目的改变，11.5 节中会进一步讨论）。我们原来假定，特征结构可以用DAG图来表示，在下一节中将讨论，这种算法的破坏性的性质要求我们适当地扩充特征结构 DAG 的图形表示方法。

11.4.1 合一的数据结构

为了对付算法的这种破坏性的归并，我们增加了代表输入特征结构的DAG的复杂性。在使用DAG来表示特征结构时，增加一个边或域（field）。具体地说，每个特征结构包括两个域：一个称为内容域（CONTENT field），一个称为指针域（POINTER field）。内容域可以为空，或者可以包含一个普通的特征结构。类似地，指针域可以为空，或者包含一个指向其他特征结构的指针。如果DAG的指针域为空，那么DAG的内容域就包含实际被处理的特征结构。另一方面，如果指针域不为空，则指针的方向就代表了实际被处理的特征结构。合一运算的合并可以通过在处理过程中改变DAG的指针域来实现。

为了具体说明我们的方法，下面讨论特征结构的扩充DAG表示方法。

$$\begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix} \tag{11.20}$$

这个矩阵图的扩充DAG表示要把CONTENT域和POINTER域作为普通的特征来处理，这种扩充DAG可以表示为如下矩阵：

$$\begin{bmatrix} \text{CONTENT} & \begin{bmatrix} \text{NUMBER} & \begin{bmatrix} \text{CONTENTS} & \text{SG} \\ \text{POINTER} & \text{NULL} \end{bmatrix} \\ \text{PERSON} & \begin{bmatrix} \text{CONTENTS} & 3 \\ \text{POINTER} & \text{NULL} \end{bmatrix} \end{bmatrix} \\ \text{POINTER} & \text{NULL} \end{bmatrix} \tag{11.21}$$

图11.4是这样的扩充DAG表示图。注意，这个扩充DAG表示在特征的最上层既包括内容域，也包括与内容域相连接的指针域。对于每个嵌入的特征结构，也采用同样的表示方法，一直表示到原子值。

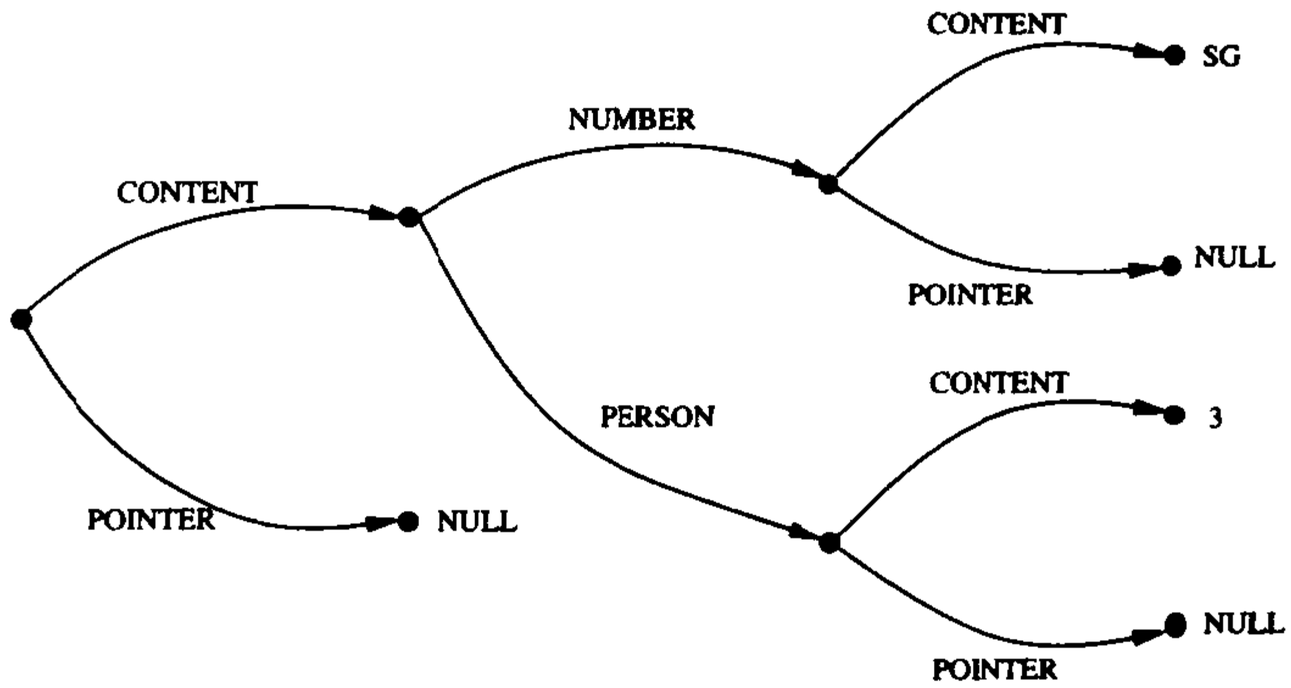


图 11.4　例（11.20）和例（11.21）的扩充 DAG 表示

在研究合一算法的细节之前，将采用这样的扩充DAG来表示如下的例子。这个合一项目的原来的扩充表示如图 11.5 所示。

$$\begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix} \sqcup \begin{bmatrix}\text{PERSON} & 3\end{bmatrix} = \begin{bmatrix}\text{NUMBER} & \text{SG} \\ \text{PERSON} & 3\end{bmatrix} \tag{11.22}$$

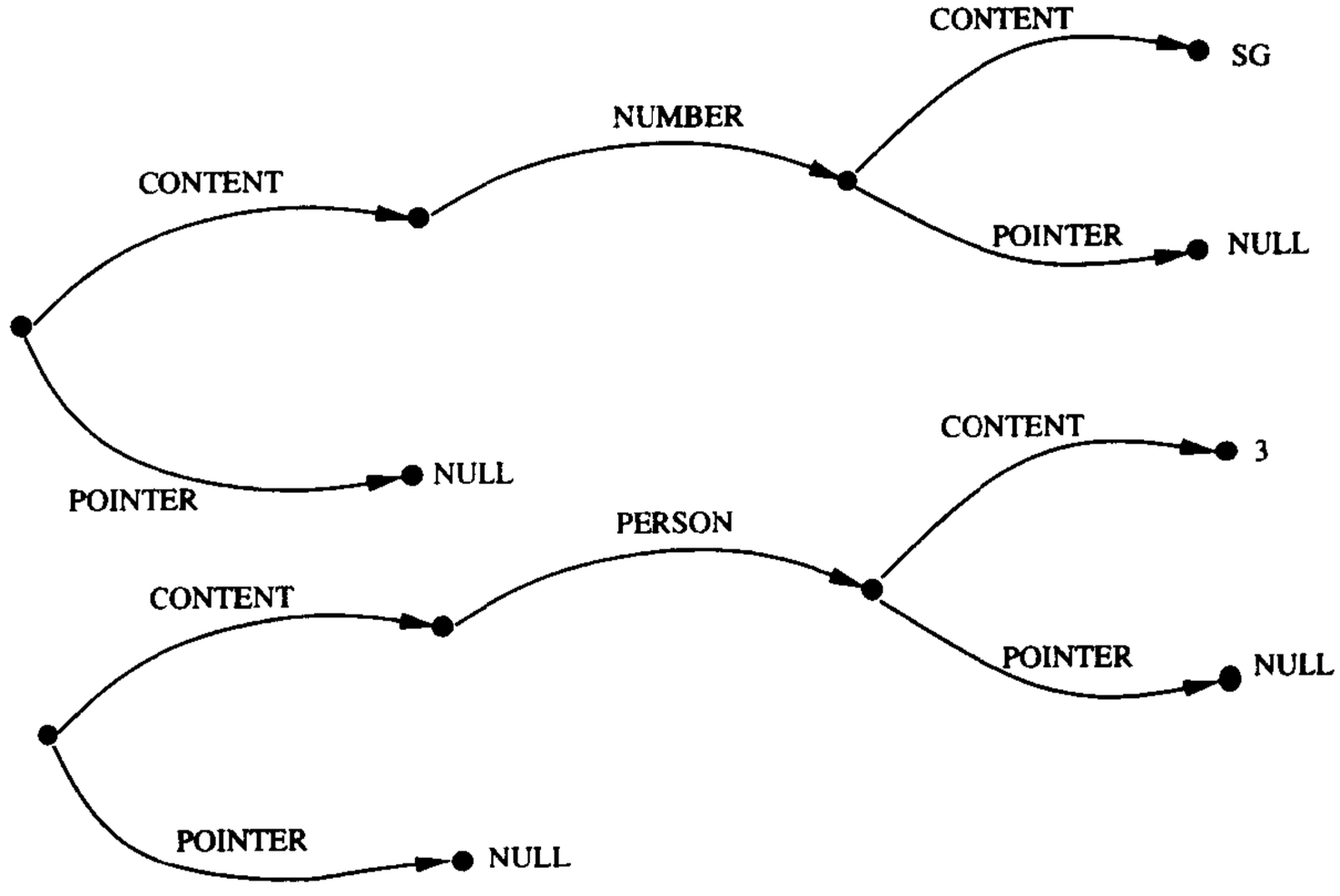


图 11.5　例（11.22）的原来的合一项目

在最高一层，我们只是说，合一的结果造出了一个新的结构，这个结构包含来自原来的两个项目的信息的并。从扩充DAG表示可以看出，怎样对原来的项目进行相加，怎样把某些指针从一个结构改变到另一个结构中，使得它们最后都包含相同的内容。在这个例子中，首先给第一个项目加上一个PERSON特征，然后把它的指针指向第二个项目中的适当位置并填充第一个项目的PERSON特征的指针域，从而给它指派一个值。

但是，处理过程还没有完成。从图 11.6 可以清楚地看出，第一个项目现在已经包含了所有的正确信息，但第二个项目还没有包含所有的正确信息，这个项目还缺少 NUMBER 特征。当然，这时我们大概可以把指针指向第一个项目中的适当位置，给第二个项目增加NUMBER特征。这样改变的结果将引起两个项目具有从合一得到的全部正确信息。可惜这样的解决办法是不恰当的，因为

这种办法并不能满足把两个项目真正合一的要求。由于这两个项目在最顶一层没有完全地合一，其中一个项目的进一步的合一将不会在另一个项目中表现出来。对于这个问题的解决办法是，简单地把第二个项目的指针域指向第一个项目的顶端。这样做的结果使得两个项目中进一步的任何改变都可以马上在它们两者中同时得到反映。这种改变的结果如图 11.7 所示。

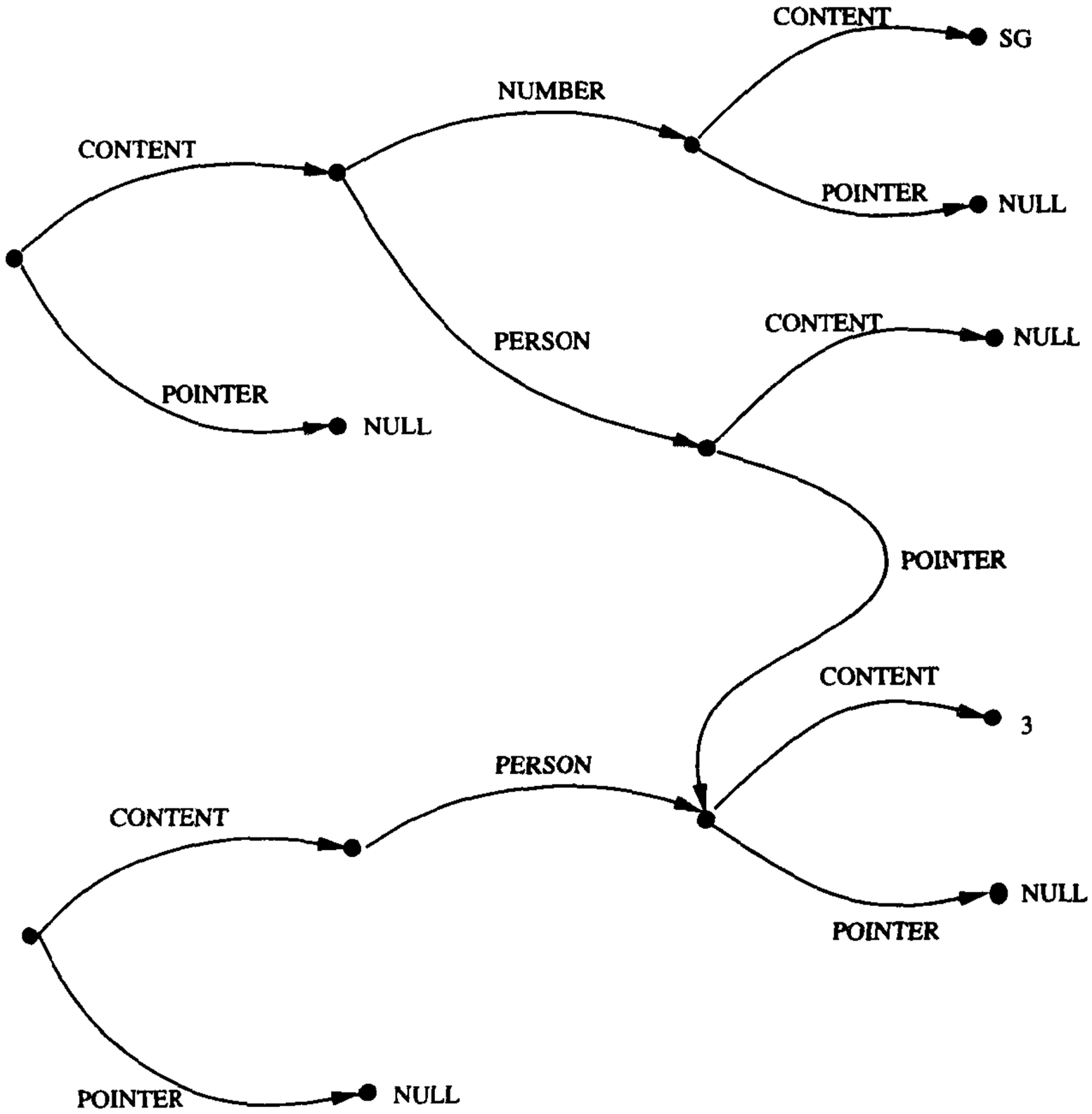


图 11.6　把第一个项目中新的 PERSON 特征指派到第二项目中的适当的值

11.4.2　合一算法

我们刚才讨论的这种算法如图 11.8 所示。这个算法接受用扩充DAG表示的两个特征结构。从代码可以知道，这个算法可以返回两个项目中的任何一个，但这样做会有些不可靠，因为这个算法的实际效果是两个输入的破坏性的合一。

这个算法的第一步是获取两个项目的真实内容。我们知道，如果在扩充的特征结构中的指针域为非空，那么这个结构的真实内容就要顺着在指针域中的指针去查找。变量 *f1-real* 和 *f2-real* 就是顺着指针查找的结果，这种结果通常称为解除参照（dereferencing）。

就像所有递归算法一样，下一步是在进入原来项目某个部分的递归调用之前检查递归的各种基本情况。这时，存在三种可能的基本情况：

- 两个项目或者其中的一个项目的值为零
- 两个项目相同
- 项目是非复杂的和非等同的

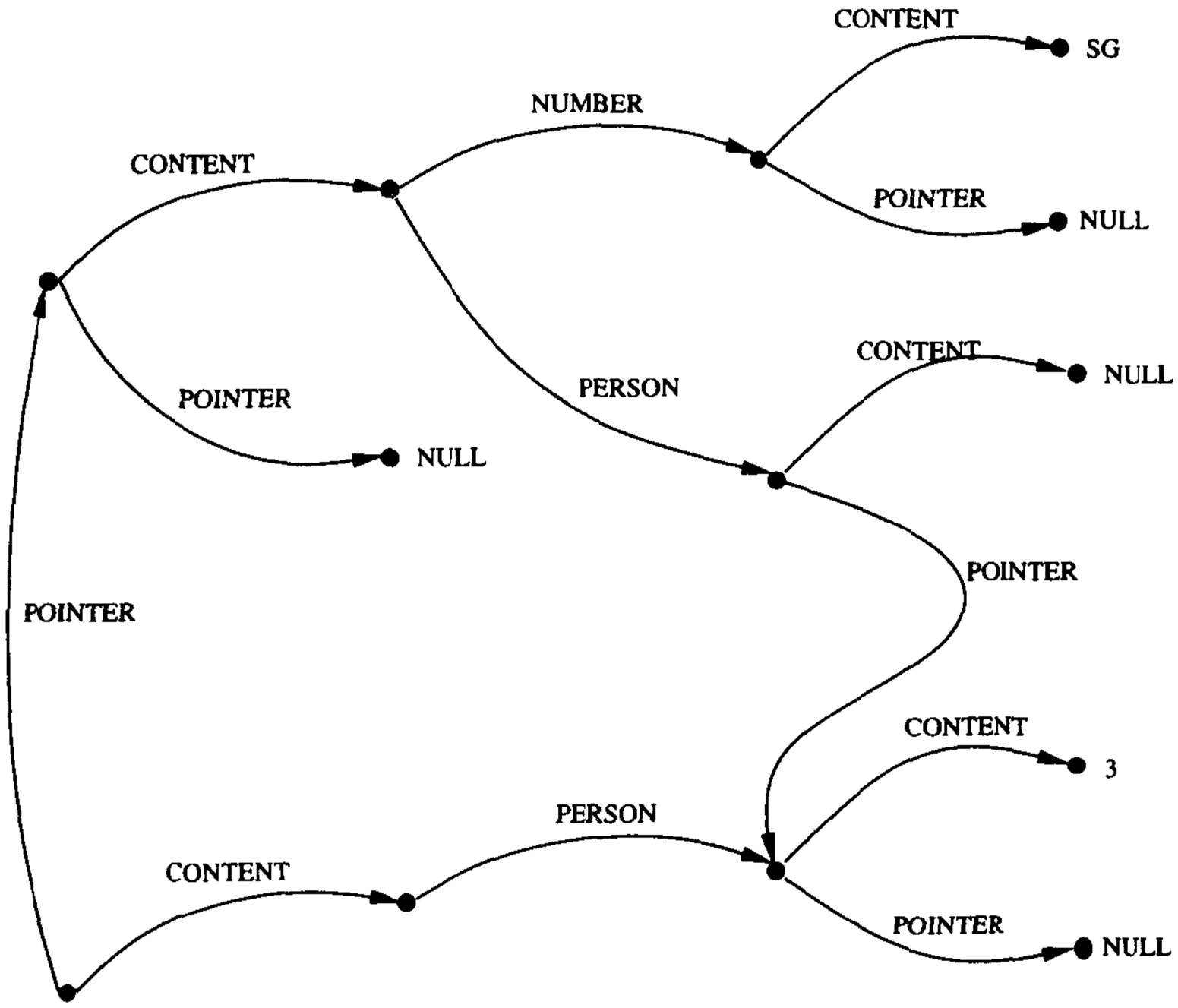


图 11.7　F1 与 F2 合一的最后结果

```
function UNIFY(f1, f2) returns fstructure or failure

  f1-real ← Real contents of f1
  f2-real ← Real contents of f2

  if f1-real is null then
    f1.pointer ← f2
    return f2
  else if f2-real is null then
    f2.pointer ← f1
    return f1
  else if f1-real and f2-real are identical then
     f1.pointer ← f2
     return f2
  else if both f1-real and f2-real are complex feature structures then
     f2.pointer ← f1
     for each feature in f2-real do
       other-feature ← Find or create
                          a feature corresponding to feature in f1-real
       if UNIFY(feature.value, other-feature.value) returns failure then
         return failure
     return f1
  else return failure
```

图 11.8　合一算法

在有一个项目为空的场合，空项目的指针域将改变为指向另一个项目，然后再返回。结果是两个结构都指向相同的值。

如果两个结构相同，那么第一个项目的指针将指向第二个项目，并且返回第二个项目。重要的是要理解为什么在这种情况下要改变这个指针。由于项目毕竟是相等的，返回其中的任何一个看起来都可以。在单一的合一时这是可行的，但我们是想让两个项目使用合一算子进行真正的合一运算。指针的改变之所以必要是因为我们想让两个项目都参与真正的合一运算，使得而后进行合一时，只要任何一个项目中添加了信息，这个信息都可以同样地添加到每个项目中。

如果前面两种情况的检查结果都不行，那么就存在两个可能：或者它们是不等同的原子值，或者它们是不等同的复杂特征结构。如果它们是不等同的原子值，就意味着这两个项目不相容，这将导致算法返回失败的值。如果它们是不等同的复杂特征，则需要进行递归调用，以确认这些复杂特征中的组成成分是否相容。在这个过程中，递归的关键是对第二个项目 *f2* 中的所有特征进行循环（loop）。这种循环的目的在于把 *f2* 中的每个特征的值与 *f1* 中相应的特征进行合一。在这个循环中，如果 *f2* 中的特征在 *f1* 中不存在，就把这个特征加到 *f1* 中，并且给该特征赋 NULL 值。这个过程继续进行，就像刚开始时处理特征那样。如果所有的合一都获得成功，那么 *f2* 的指针域就指向 *f1*，完成结构的合一运算，并把合一的值返回到 *f1*。

我们注意到，这样的算法的不足之处在于它可能产生包含圆圈（cycle）的特征结构。这种情况发生在我们要求算法把一个结构与第二个结构合一，而第二个结构的子部分中又包含第一个结构时。避免这种结构的途径是使用称为状况检查（occur check）的方法(Robinson, 1965)。状况检查分析输入的 DAG，当一个项目被作为子部分包含在另一个项目中时，就返回特征。实际上，由于状况检查的计算开支太大，在大部分应用系统中都省去了这样的检查。

示例

为了说明这个算法，让我们把下列的例子走一遍。

$$\begin{bmatrix} \text{AGREEMENT} & \boxed{1}\begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix} \\ \text{SUBJECT} & \begin{bmatrix}\text{AGREEMENT} & \boxed{1}\end{bmatrix} \end{bmatrix} \sqcup \begin{bmatrix} \text{SUBJECT} & \begin{bmatrix}\text{AGREEMENT} & \begin{bmatrix}\text{PERSON} & 3\end{bmatrix}\end{bmatrix}\end{bmatrix} \tag{11.23}$$

图 11.9 是这个合一运算的项目的扩充表示。注意，在第一个项目中重入结构是通过使用指针域（PTR 域）来表示的。

这两个原来的项目既不是等同的，又不是空的，也不是原子的，所以进入主循环。在特征 *f2* 上循环时，算法试图递归地把相应于 SUBJECT 特征 *f1* 和 *f2* 的值进行合一。

$$\begin{bmatrix}\text{AGREEMENT} & \boxed{1}\end{bmatrix} \sqcup \begin{bmatrix}\text{AGREEMENT} & \begin{bmatrix}\text{PERSON} & 3\end{bmatrix}\end{bmatrix}$$

这两个项目也不是等同的，又不是空的，也不是原子的，所以再次进入循环，导致递归地检查特征 AGREEMENT 的值。

$$\begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix} \sqcup \begin{bmatrix}\text{PERSON} & 3\end{bmatrix}$$

在循环检查第二个项目的特征时，发现了在第一个项目中没有 PERSON 特征。因此在第一个项目中加上一个PERSON特征，开始时置这个特征的初值为NULL。这样就可以把前面的合一改成如下形式：

$$\begin{bmatrix} \text{NUMBER} & \text{SG} \\ \text{PERSON} & \text{NULL} \end{bmatrix} \sqcup \begin{bmatrix} \text{PERSON} & 3 \end{bmatrix}$$

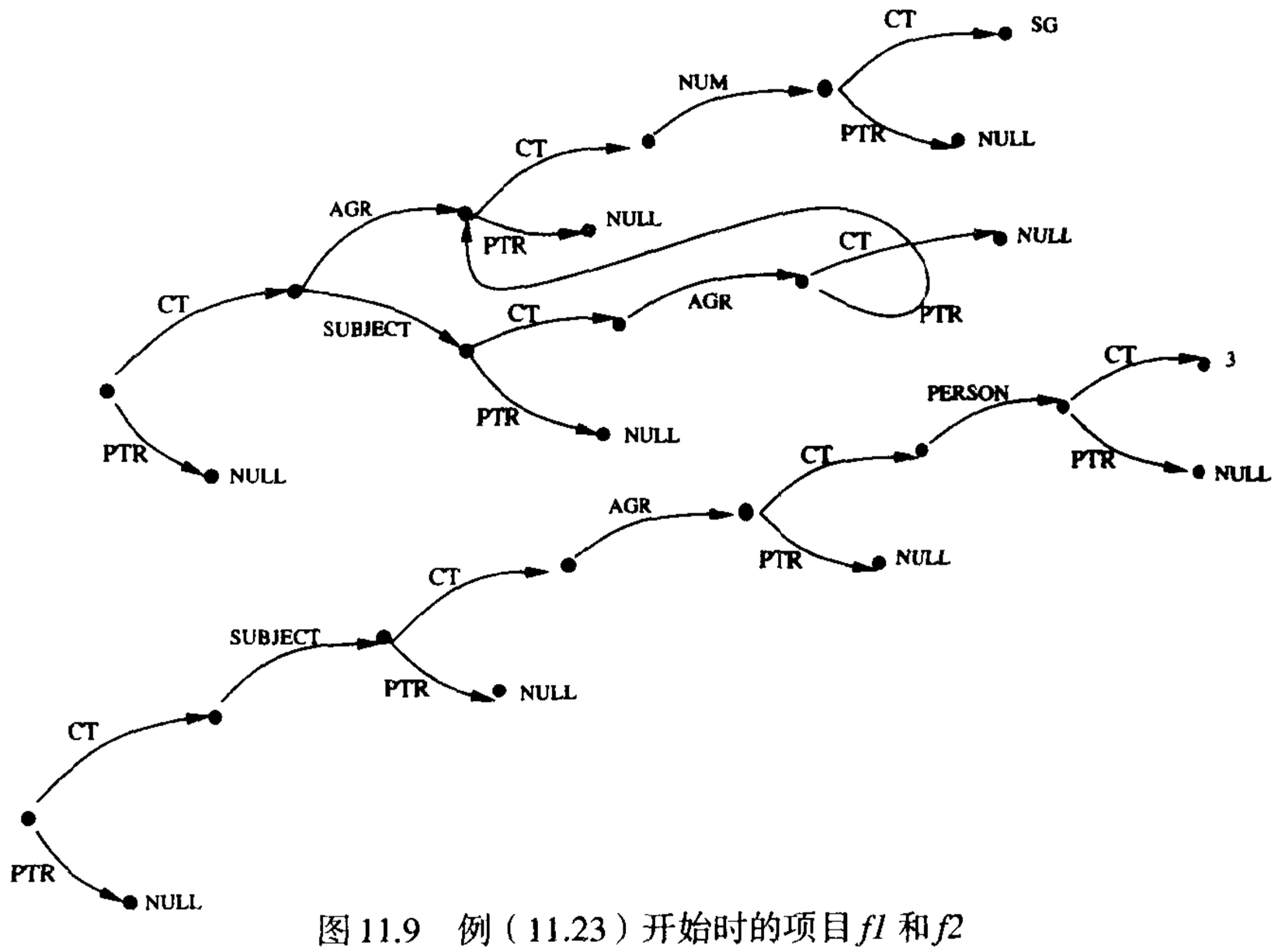


图 11.9　例（11.23）开始时的项目 *f1* 和 *f2*

在创建了这个新的PERSON特征之后，下面一个递归调用将导致把在第一个项目中新特征的值NULL与第二个项目中的3这个值进行合一。这个递归调用的结果是把*f2*中的3这个值指派到第一个项目的指针域中，如图 11.10 所示。

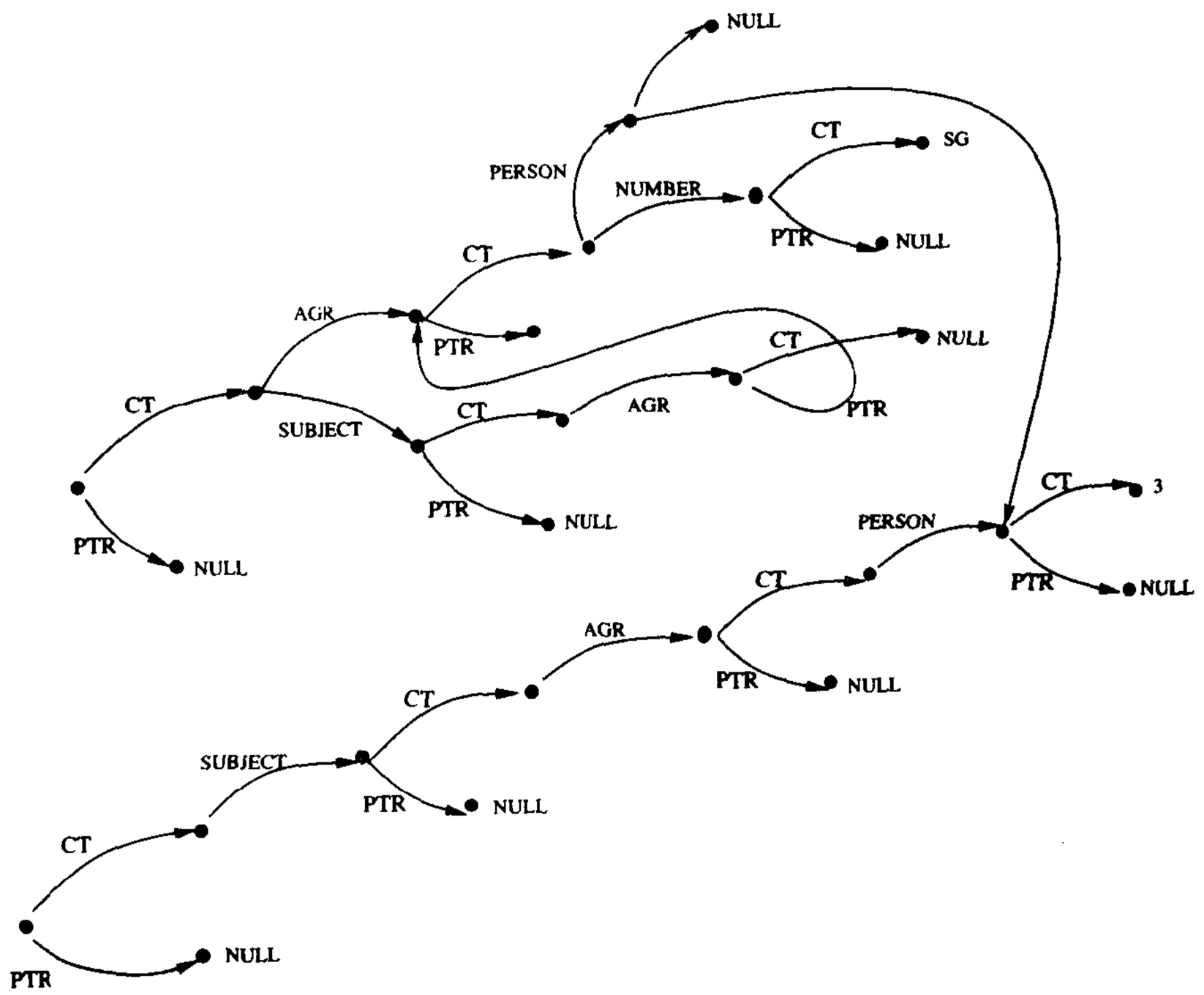


图 11.10　在递归地添加了新特征 PERSON 的值之后 *f1* 和 *f2* 的情况

由于在项目*f2*中的任何递归层级没有检查出更多的特征，在递归的每一轮都把项目*f2*的指针指向项目*f1*，并返回到*f1*。所有的这些指派的结果如图11.11所示。

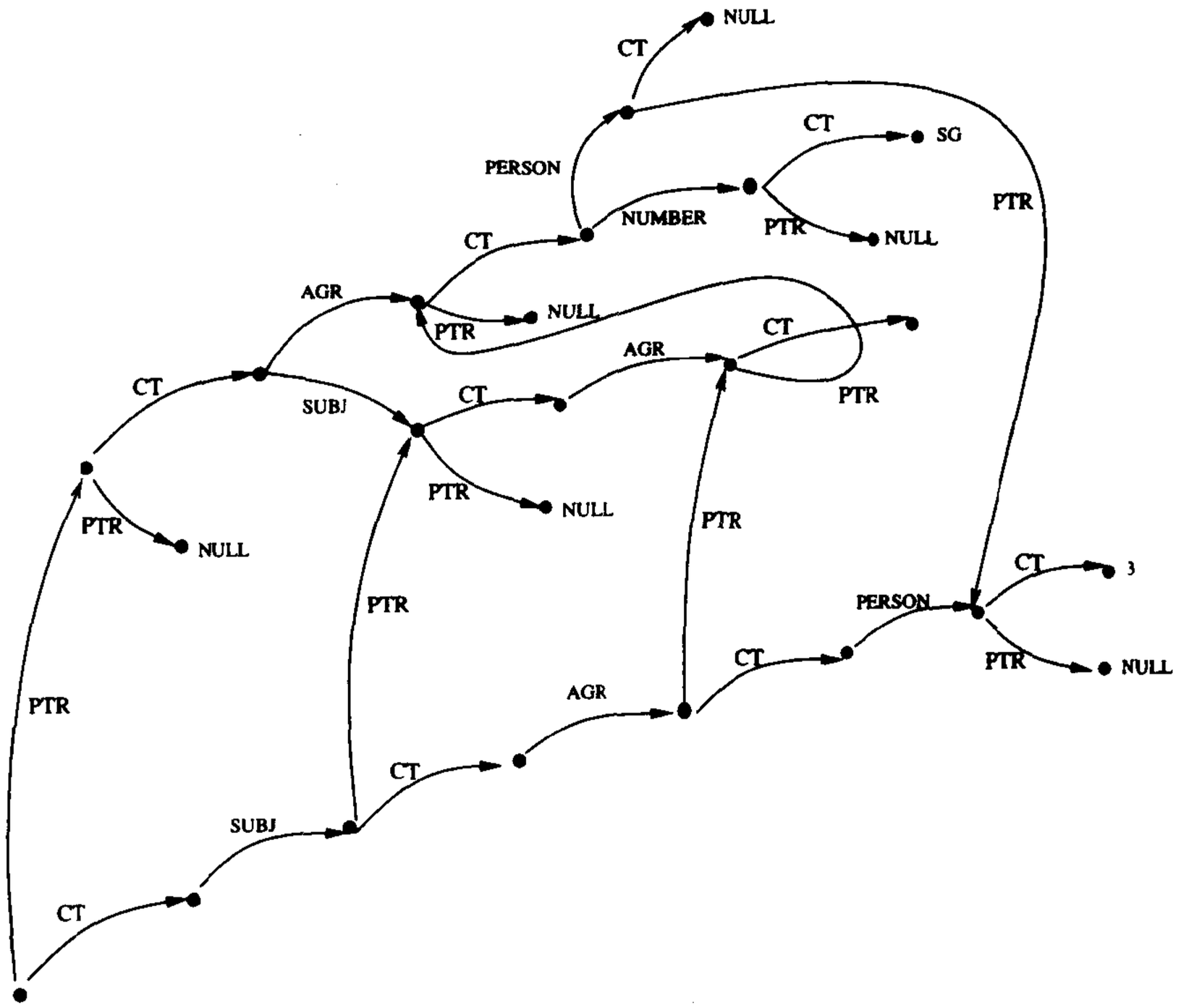


图11.11 *f1*和*f2*的最终结构

11.5 带有合一约束的剖析

现在，我们已经有足够的必要知识把特征结构与合一结合在一个剖析器中了。幸运的是，合一运算是与合一的顺序无关的，合一运算的这种特性使我们可以放心大胆地不考虑剖析器中使用的具体搜索策略。只要有与语法的上下文无关规则相联系的合一约束，以及与搜索状态有关的特征结构，就可以使用第10章中描述的任何标准搜索算法。

当然，这种情况给我们留下了非常广阔的空间来实现各种各样的策略。例如，我们可以在使用规则的上下文无关成分之前先进行简单的剖析，然后根据事实来建立结果树的特征结构，对于包含合一特征的那些剖析进行过滤。尽管使用这种方法最后形成的结构都是良构的（well-formed），但它不能以合一的力量来减少在搜索过程中剖析器的搜索空间。

下一节描述的方法可以更好地利用合一的力量，把合一约束直接结合到Earley算法的剖析过程中，当非良构的结构出现时就被很快地清除。这种方法只要求对于Earley的基本算法做最小限度的改变。然后，简单地介绍一种与上下文无关方法差别较大的合一剖析方法。

11.5.1 把合一结合到Earley剖析器中

把特征结构与合一运算结合到Earley算法中的目的有两个：第一，使用特征结构，从而为剖析的组成成分提供更丰富的表达方式；第二，阻止成分进入破坏合一约束的非良构成分的线图中。只要最小限度地改变原来的Earley算法，就可以达到这样的目的。

第一个改变是改变在原来的代码中使用的各种表达方法。Earley算法在运算时要使用一套无装饰的上下文无关语法规则，来填充一种称为线图的数据结构，这种线图由状态的集合组成。在剖析的终点，构成线图的状态可以表示对于输入的一切可能的剖析结果。因此，我们要进行的改变首先从改变上下文无关语法规则和线图中的状态开始。

在改变规则时，除了它们当前的成分之外，还包括从它们的合一约束推导出的特征结构。更具体地说，我们将使用由规则列出的约束来建立特征结构，把特征结构表示为 DAG，以便在剖析过程中与规则一起使用它们。

我们来研究如下的带有合一约束的上下文无关规则：

$$S \rightarrow NP\ VP$$
$$\langle NP\ \text{HEAD AGREEMENT}\rangle = \langle VP\ \text{HEAD AGREEMENT}\rangle$$
$$\langle S\ \text{HEAD}\rangle = \langle VP\ \text{HEAD}\rangle$$

把这些约束转变为特征结构，得到如下的结构：

$$\begin{bmatrix} \text{S} & \begin{bmatrix}\text{HEAD} & \boxed{1}\end{bmatrix} \\ \text{NP} & \begin{bmatrix}\text{HEAD} & \begin{bmatrix}\text{AGREEMENT} & \boxed{2}\end{bmatrix}\end{bmatrix} \\ \text{VP} & \begin{bmatrix}\text{HEAD} & \boxed{1}\begin{bmatrix}\text{AGREEMENT} & \boxed{2}\end{bmatrix}\end{bmatrix} \end{bmatrix}$$

在这个推导中，我们首先对上下文无关规则中的每个部分创建顶层特征，在这个例子中，这些顶层特征是 S，NP 和 VP；使用这样的方法，把各种不同的约束结合到一个单独的结构中。然后，顺着约束中的路径等式，进一步再把各种成分加到这个结构中。注意，这只是一种纯粹的表记法的变换，DAG 和约束等式中包含的信息都是完全一样的。但是，把各种约束捆绑到一个单独的特征结构中，这样的形式就可以直接放到合一算法中。

第二个改变是改变用于表示 Earley 算法中局部剖析的状态。原来的状态包括所用的上下文无关规则、表示规则已经完成部分的点的位置、状态的开始和终止位置以及表示状态中已完成子部分的其他状态的一个表。现在再加上一个附加的区域，这个区域包含表示与状态相应的特征结构的 DAG。注意，当一个规则首次被 PREDICTOR 用来建立一个状态时，与状态相联系的 DAG 将由规则中检索出来的 DAG 组成。例如，当 PREDICTOR 使用上面的 S 规则而进入到线图中的一个状态时，上面给出 DAG 将是初始的 DAG。我们把状态表示如下，其中 DAG 表示上面给出的特征结构：

$$S \rightarrow \bullet\ NP\ VP,\ [0,0],[],Dag$$

给出了这些新增加的表示之后，就可以来改变算法本身了。在行动方面的最重要的改变是：当通过扩充现有状态建立一个新状态时，在 COMPLETER 程序中发生了变化。我们知道，当一个完成的成分加到线图中时，就要调用 COMPLETER。COMPLETER 的任务是发现并扩充线图中现有状态，这个状态正在查找与新完成的成分相容的成分。因此，COMPLETER 是一种结合两个状态中的信息建立新状态的功能，它要使用合一的操作。

更具体地说，COMPLETER 通过发现现有状态的方式，把一个新的状态加到线图中，并把现有状态中的“ • ”由新的完成状态向前推进。当直接在“ • ”后面的成分的范畴与新完成的成分

的范畴相匹配时，就可以把“ • ”向前推进。为了便于使用特征结构，我们可以改变这样的做法，而把与新的完成状态相联系的特征结构与被推进的特征结构中的适当部分进行合一。如果这样的合一取得成功，那么新状态的DAG就接受合一后的结构，并进入线图中。如果合一失败，则没有新的状态进入线图中。COMPLETER的这种改变情况如图11.12所示。

```
function EARLEY-PARSE(words, grammar) returns chart

  ENQUEUE((γ → • S, [0,0], dag_γ), chart[0])
  for i ← from 0 to LENGTH(words) do
   for each state in chart[i] do
     if INCOMPLETE?(state) and
              NEXT-CAT(state) is not a part of speech then
        PREDICTOR(state)
     elseif INCOMPLETE?(state) and
              NEXT-CAT(state) is a part of speech then
        SCANNER(state)
     else
        COMPLETER(state)
   end
  end
  return(chart)

procedure PREDICTOR((A → α • B β, [i, j], dag_A))
   for each (B → γ) in GRAMMAR-RULES-FOR(B, grammar) do
       ENQUEUE((B → • γ, [j, j], dag_B), chart[j])
   end

procedure SCANNER((A → α • B β, [i, j], dag_A))
   if B ⊂ PARTS-OF-SPEECH(word[j]) then
      ENQUEUE((B → word[j], [j, j + 1], dag_B), chart[j+1])

procedure COMPLETER((B → γ •, [j,k], dag_B))
   for each (A → α • B β, [i, j], dag_A) in chart[j] do
     if new-dag ← UNIFY-STATES(dag_B, dag_A, B) ≠ Fails!
       ENQUEUE((A → α B • β, [i,k], new − dag), chart[k])
   end

procedure UNIFY-STATES(dag1, dag2, cat)
  dag1-cp ← COPYDAG(dag1)
  dag2-cp ← COPYDAG(dag2)
  UNIFY(FOLLOW-PATH(cat, dag1-cp), FOLLOW-PATH(cat, dag2-cp))

procedure ENQUEUE(state, chart-entry)
   if state is not subsumed by a state in chart-entry then
      PUSH(state, chart-entry)
   end
```

图11.12　修改Earley算法使之包括合一运算

我们通过剖析 *That flight* 来研究这个过程。假设在 *That flight* 中 *That* 已经被识别出来，当时的状态如下：

$$NP \rightarrow Det \bullet Nominal\,,\ [0,1],[S_{Det}],Dag_1$$

$$Dag_1\begin{bmatrix}\text{NP} & \begin{bmatrix}\text{HEAD} & \boxed{1}\end{bmatrix} \\ \text{DET} & \begin{bmatrix}\text{HEAD} & \begin{bmatrix}\text{AGREEMENT} & \boxed{2}\begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix}\end{bmatrix}\end{bmatrix} \\ \text{NOMINAL} & \begin{bmatrix}\text{HEAD} & \boxed{1}\begin{bmatrix}\text{AGREEMENT} & \boxed{2}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

现在来研究下面的情况，这时剖析器已经处理了*flight*，并由此产生了如下的状态：

$$Nominal \rightarrow Noun\bullet,[1,2],[S_{Noun}],Dag_2$$

$$Dag_2\begin{bmatrix}\text{NOMINAL} & \begin{bmatrix}\text{HEAD} & \boxed{1}\end{bmatrix} \\ \text{NOUN} & \begin{bmatrix}\text{HEAD} & \boxed{1}\begin{bmatrix}\text{AGREEMENT} & \begin{bmatrix}\text{NUMBER} & \text{SG}\end{bmatrix}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

为了推进*NP*规则，分析器把在Dag_2的NOMINAL特征中发现的特征结构与在*NP*的Dag_1的NOMINAL特征中发现的特征结构进行合一。正如在原来的算法中那样，我们建立起一个新状态来表示现有状态已经被向前推进这一事实。这个新状态的DAG也就是上面合一的结果所形成的DAG。

对于原来算法的最后一个改变是检查已包含在线图中的状态。在原来的算法中，ENQUEUE这个功能拒绝进入线图中的与该线图已有的状态“等同”（identical）的任何状态。所谓“等同”意味着规则相同，开始位置和结束位置相同，点“•”的位置相同。这种检查使得算法可以避免与左递归规则相关的无穷递归问题。

在带合一约束的剖析中，我们的状态当然会更复杂，因为这时的状态中具有与其相关联的特征结构。在原来的标准看来似乎等同的状态，现在事实上可能是不同的，因为与它们联系的DAG可能有所不同。要解决这样的问题显然只需要扩充等同性的检查，使得这种检查包含与状态有关联的DAG，不过我们还可以改善这种解决办法。

之所以要求改善，是因为要进行等同性的检查。这种改善的目的在于阻止把无用状态加到线图中，以免造成浪费，这种改善对于剖析的影响可由现有状态来实现。换言之，我们希望阻止那些造成重复工作的状态进入线图中，这些工作最终可以由其他状态来完成。当然，很显然这是关于等同状态的问题，但这也是线图中的状态的问题，这个问题比研究新状态更具有一般性。

我们来研究下面的情况，其中线图包含的状态的*Dag*没有对*Det*进行约束。

$$NP \rightarrow \bullet Det\ NP,[i,i],[],Dag$$

这样的状态只是说，它期待在位置*i*有一个*Det*，任何的*Det*都可以做到这一点。

现在来研究如果剖析器试图把一个新状态插入到线图中，而线图中的状态与这个状态完全等同，只是DAG限制*Det*为单数时的情况。在这样的场合，尽管有关状态不等同，但是在线图中添加新状态不能得到什么东西，因此这样的插入被阻止。

为了看到这一点，让我们来考虑所有的情况。如果添加一个新状态，则单数*Det*在两个状态中都是匹配的，而且两个状态都要向前推进。由于特征的合一，两个状态都有DAG指出它们的*Det*都是单数，其结果是线图中的状态成倍增加。如果遇到复数的*Det*，那么新状态将拒绝这个*Det*，并且不向前推进，而这时老的规则将向前推进，进入线图中的一个新状态。另一方面，如果不把新状态放入线图中，复数或单数的*Det*将与更一般的状态相匹配，并且向前推进，这将导致把一个新状

态加到线图中。注意，除了避免成倍增加状态的数目之外，这种情况与把新状态加到线图中的情形是完全一样的。总而言之，把一个比在线图中的状态更加特殊的状态加到线图中，最后还是一无所获。

可喜的是，我们可以使用前面讨论过的“蕴涵于”（subsumption）的概念，形式化地描述特征结构中的“一般化”（generalization）和“特殊化”（specialization）之间的关系。这意味着，改变ENQUEUE的一种合适的办法是检查一个新建立的状态是不是蕴涵于线图的任何现有状态中。如果是，就不允许它进入线图中。更具体地说，如果一个新状态与现有状态的规则相同，开始和结束位置相同，子部分相同，点“ • ”的位置相同，并且如果存在状态的DAG蕴涵于新状态的DAG（也就是说，$Dag_{old} \sqsubseteq Dag_{new}$），那么这个新状态就不能加到线图中。对于原来的Earley的ENQUEUE过程的必要改变，可参见图11.12。

11.5.2 复制的必要性

在UNIFY-STATE过程中调用COPYDAG需要费些力气。我们知道，Earley算法（以及一般的动态规划算法）的一个长处在于，当状态进入线图之后，它们可以作为不同推导的部分被一而再、再而三地使用，包括最后不能导致剖析成功的部分。这种能力使我们认识到，当那些已经处于线图中的状态还没有得到更新，以反映它们的点“ • ”的进展情况时，我们可以先复制它们，然后再进行更新，让原来的状态保持不变，以便它们能够在进一步的推导中得到使用。

由于合一算法具有破坏性，在UNIFY-STATE中的COPYDAG要求保持这样的做法。如果只是单纯地把与现有状态相关的DAG进行合一，这些状态将会由于合一而发生改变，由于使用了COMPLETER功能，因此后来在同样的形式中就不能使用。注意，不论合一是成功还是失败了，这都是一种负面的后果，因为原来状态的任何情况都被改变了。

我们来研究一下，在下面的例子中，当前面的合一尝试失败了，如果不能调用COPYDAG将会发生什么样的情况。

Show me morning flights. (11.24)

假定对于双及物动词*show*，剖析器含有如下的条目以及如下的及物动词规则和双及物动词规则：

Verb → *show*

⟨*Verb* HEAD SUBCAT FIRST CAT⟩ = NP

⟨*Verb* HEAD SUBCAT SECOND CAT⟩ = NP

⟨*Verb* HEAD SUBCAT THIRD⟩ = END

VP → *Verb NP*

⟨*VP* HEAD⟩ = ⟨*Verb* HEAD⟩

⟨*VP* HEAD SUBCAT FIRST CAT⟩ = ⟨*NP* CAT ⟩

⟨*VP* HEAD SUBCAT SECOND⟩ = END

VP → *Verb NP NP*

⟨*VP* HEAD⟩ = ⟨*Verb* HEAD⟩

⟨*VP* HEAD SUBCAT FIRST CAT⟩ = ⟨NP_1 CAT ⟩

⟨*VP* HEAD SUBCAT SECOND CAT⟩ = ⟨NP_2 CAT ⟩

⟨*VP* HEAD SUBCAT THIRD⟩ = END

当读到单词 *me* 时，表示及物动词短语的状态将被完成，因为它的点已经移动到终点。因此，在把这个完成状态加入到线图中之前，COMPLETER 要调用 UNIFY-STATES。这将会失败，因为这两条规则的SUBCAT结构不能合一。当然，这正是我们所想到的，因为 *show* 是一个双及物动词。遗憾的是，因为合一算法具有破坏性，这时已经改变了附着在表示 *show* 的状态上的DAG，以及附着在VP的状态上的DAG，这样就破坏了这些状态，使得它们没有可能在之后的正确动词短语结构中得到使用。因此，为了保证这些状态能够在多次推导中一而再、再而三地得到使用，在试图对这些状态进行合一运算之前，必须对与这些状态相关的 DAG 进行复制。

进行各种这样的复制的开支是很高昂的，要付出很高的代价。为了把复制的开支降低到最低限度，研制了各种不同的技术(Pereira, 1985; Karttunen and Kay, 1985; Tomabechi, 1991; Kogure, 1990)。Kiefer et al.(1999)描写了用于加快大型的基于语料库的剖析系统的一系列有关技术。

11.5.3　合一剖析

让我们来观察表示增强语法规则的另一种方法，这是一种在剖析中使用合一运算的更根本的办法。我们来研究下面的 *S* 规则，这个规则在本章中自始至终都在使用：

$$S \rightarrow NP\ VP$$
$$\langle NP\ \text{HEAD AGREEMENT}\rangle = \langle VP\ \text{HEAD AGREEMENT}\rangle$$
$$\langle S\ \text{HEAD}\rangle = \langle VP\ \text{HEAD}\rangle$$

改变这个规则中的上下文无关部分的一个有趣办法是改变语法范畴的表示方法。具体地说，我们可以把关于规则部分的范畴信息放在特征结构内，而不是放在规则的上下文无关部分内。下面是这种方法的一个典型的具体实例(Shieber, 1986)。

$$X_0 \rightarrow X_1\ X_2$$
$$\langle X_0\ \text{CAT}\rangle = S$$
$$\langle X_1\ \text{CAT}\rangle = NP$$
$$\langle X_2\ \text{CAT}\rangle = VP$$
$$\langle X_1\ \text{HEAD AGREEMENT}\rangle = \langle X_2\ \text{HEAD AGREEMENT}\rangle$$
$$\langle X_0\ \text{HEAD}\rangle = \langle X_2\ \text{HEAD}\rangle$$

现在只是集中讨论规则中的上下文无关成分。这个规则的含义是成分 X_0 包含两个组成成分 X_1 和 X_2，成分 X_1 直接位于成分 X_2 的左侧。关于这些成分的实际范畴的信息放在规则的特征结构内。在这种情况下，要指出 X_0 是一个 S，X_1 是一个 NP，X_2 是一个 VP。修改 Earley 算法来处理这种记法上的改变是轻而易举的。这时不必在规则的上下文无关成分中查找成分的范畴，只需简单地在与规则相关的 DAG 的 CAT 特征中进行查找。

当然，由于这两种规则包含着完全相同的信息，这样的改变究竟有什么明显好处，一时可能还看不太清楚。为了明白这样的改变带来的潜在好处，我们来研究下面的例子：

$$X_0 \rightarrow X_1\ X_2$$
$$\langle X_0\ \text{CAT}\rangle = \langle X_1\ \text{CAT}\rangle$$
$$\langle X_2\ \text{CAT}\rangle = PP$$

$$X_0 \rightarrow X_1\ and\ X_2$$
$$\langle X_1\ \text{CAT}\rangle = \langle X_2\ \text{CAT}\rangle$$
$$\langle X_0\ \text{CAT}\rangle = \langle X_1\ \text{CAT}\rangle$$

第一个规则试图把已经见过的规则 $NP \rightarrow NP\ PP$ 和 $VP \rightarrow VP\ PP$ 等加以概括。这个规则的意思是任何范畴的后面都可以接一个介词短语 PP，而最后构成的新成分与原来的成分具有相同的范畴。类似地，第二个规则试图把规则 $S \rightarrow S\ and\ S$ 和 $NP \rightarrow NP\ and\ NP$ 等加以概括[①]。这个规则的意思是任何成分可以与范畴相同的成分结合起来，形成具有相同类型的新范畴。这些规则的共同之处在于，它们使用上下文无关规则，而这些规则包含带有约束但没有具体说明的范畴，因此这些东西是旧规则格式不能实现的。

由于这些规则使用了CAT这个特征，其效果在旧格式中当然可以通过简单地枚举各种实例的方式来近似表示。当存在这样的规则或结构时，它们所包含的成分，使用任何现有句法范畴都很不容易描述出来，在这种情况下，这种新方法就更加引人注目了。

我们来研究下面来自华尔街日报语料库WSJ的关于英语中HOW-MANY结构的例子(Jurafsky, 1992)：

How early does it open? (11.25)

How deep is her Greenness? (11.26)

How papery are your profits? (11.27)

How quickly we forget? (11.28)

How many of you can name three famous sporting Blanchards? (11.29)

正如例子中所显示的，HOW-MANY结构包含两个成分：一个成分是词汇项目 *how*，一个成分是很难使用句法描述的某个项目或短语。我们感兴趣的是这里的第二个成分。从例子中可以看出，这个成分可以是形容词、副词或某种表示数量的短语(尽管并不是其中的所有成分都可以从语法上加以说明)。显而易见，比较好的办法是把第二个成分作为一个“分等级的概念”来处理，我们把它记为SCALAR，使用特征结构可以把这样的约束表示如下：

$$X_0 \rightarrow X_1\ X_2$$
$$\langle X_1\ \text{ORTH} \rangle = \langle how \rangle$$
$$\langle X_2\ \text{SEM} \rangle = \langle\ \text{SCALAR} \rangle$$

这种包含语义成分的规则的完全说明将在第14章中介绍。此刻最关键的是理解使用特征结构，一个语法规则可以对它的成分加上各种约束，还可以包括那些用句法范畴的概念不能描述的约束。

处理这类规则当然要改变原来的剖析方案。过去研究的所有剖析方法都是由输入中各种成分的句法范畴驱动的。尤其是它们的基本方法都是把预测到的范畴与已发现的范畴相匹配。例如，我们来考虑图11.12中的COMPLETER功能的操作。这个功能在线图中搜索能够由新完成的状态向前推进的状态。把新完成的状态的范畴与现有状态中跟随在点“•”之后的成分的范畴相匹配，就可以实现这样的功能。显而易见，当不存在这样的范畴可以匹配时，这种方法就会遇到麻烦。

补救COMPLETER这个问题的办法是，在线图中搜索那些其DAG与新完成状态的DAG进行合一的状态。这样就不必要求状态或规则必须有范畴。PREDICTOR按同样的方式来修改，使它把状态加到线图的状态中，其中成分 X_0 DAG可以与跟随在所预测状态的点“•”之后的成分相匹配。作为练习，请读者对图11.12中的伪代码进行一些改变，以便实现这种风格的剖析。

① 考虑到具体的语言现象，这些规则还可以加以改进或完善。

11.6　类型与继承

I am surprised that ancient and modern writers have not attributed greater importance to the laws of inheritance...[①]

Alexis de Tocqueville, *Democracy in America*, 1840

迄今为止，我们介绍过的基本特征结构有两个问题，因此需要把这种形式化方法加以扩充。第一个问题是这样的特征结构无法把一个约束准确地放到它应该具有的特征值的位置。例如，我们曾经含蓄地假定，NUMBER 这个特征只可以取 SG 和 PL 作为值。但是，在现在的系统中，这并不能阻止 NUMBER 取 3RD 或 FEMININE 作为它的值：

$$\begin{bmatrix}\text{NUMBER} & \text{FEMININE}\end{bmatrix}$$

这个问题导致很多基于合一的语法理论添加了各种机制，试图约束特征可能取的值。例如，功能合一语法（Functional Unification Grammar, FUG）(Kay, 1979, 1984, 1985)和词汇功能语法（Lexical Functional Grammar, LFG）(Bresnan, 1982)都设法使像 *sneeze*（打喷嚏）这样的不及物动词不与直接宾语进行合一（*Marin sneezed Toby*）。在 FUG 中，增加了一个特殊的原子 none，这个原子不允许进行任何合一。在 LFG 中增加了连贯性条件（coherence），这个条件要说明在什么时候不能填写特征。广义短语结构语法（Generalized Phrase Structure Grammar, GPSG）(Gazdar et al., 1985, 1988)增加了一种特征共现限制（feature co-occurrence restriction）来阻止带有某种动词特征的名词等。

这种简单特征结构的第二个问题是这样的特征结构无法从众多的特征中捕捉到贯穿这些特征的一般性的东西。例如，前面在讨论次范畴化时描述了英语动词短语的很多类型，这些类型共享了很多特征，因此我们可以给英语动词做出很多种次范畴化框架。句法研究者正在探索表达这种一般性的途径。

上述这两个问题的总的解决办法是使用“类型”（type）。合一语法的类型系统具有如下的特点：

1. 每个特征结构用一个类型来标记。
2. 反之，每个类型具有适切性条件（appropriateness condition）来表示什么样的特征对于这个类型是适切的。
3. 各种类型被组织成一个类型层级体系（type hierarchy）。在这个层级体系中，比较具体的类型继承比较抽象的类型的性质。
4. 对合一运算进行修改，使得合一运算除了对特征和特征值进行合一之外，还能对特征结构的类型进行合一。

在这种类型化的特征结构系统中，类型是一种新的类别，它与标准特征结构中的属性和值很相似。类型可以分为两种：一种是简单类型（又称为原子类型），一种是复杂类型。我们先来讨论简单类型。一个简单类型是一个原子符号，如 sg 或 pl（我们使用粗体字母来表示类型），我们用这种简单类型来代替标准特征结构中的简单原子值。所有的类型组织成一个有多种继承关系的类型层级体系，即一个偏序（partial order）或格（lattice）。图 11.13 是新类型 agr 的类型层级体系，这是一个原子类型，可以作为特征 AGREE 的值。

---

① 我感到非常惊讶的是，无论是古代的作家还是现代的作家，都没有注意到继承法的极端重要性。

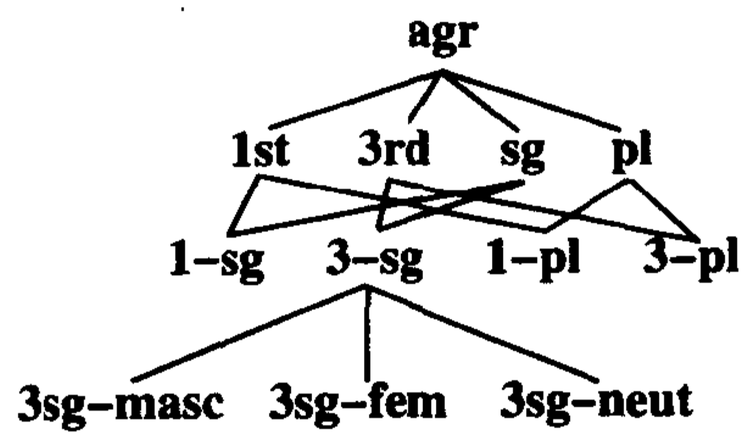


图 11.13 类型 agr 的子类型的一个简单的类型层级体系，agr 可以作为特征 AGREE 的值(Carpenter, 1992)

在图 11.13 的层级体系中，3rd 是 agr 的子类型，3-sg 是 3rd 和 sg 的子类型。在类型层级体系中类型可以进行合一，任意两个类型的合一的结果是更具体的类型，比输入的两个类型更加具体。因此，有：

$$\textbf{3rd} \sqcup \textbf{sg} = \textbf{3sg}$$
$$\textbf{1st} \sqcup \textbf{pl} = \textbf{1pl}$$
$$\textbf{1st} \sqcup \textbf{agr} = \textbf{1st}$$
$$\textbf{3rd} \sqcup \textbf{1st} = \textit{undefined}$$

没有定义的合一项的两个类型进行合一，其结果是不确定的，尽管我们也可以使用符号“⊥”(Aït-kaci, 1984)来表示这个失败的类型。

第二种类型是复杂类型，它要说明：

- 与类型相适应的一组特征
- 对于这些特征的值的限制
- 在这些值之间的相等性约束

我们来研究对于复杂类型 verb 的一种简化的表示，这种表示正好可以表达一致关系和动词形态形式的信息。定义一个动词时，要定义两个相应的特征，一个特征是 AGREE，一个特征是 VFORM，还要定义这两个特征的值的类型。我们假定特征 AGREE 取在图 11.13 中定义的类型 agr 的值，特征 VFORM 取类型 vform 的值（这里假定 vform 有 7 个子类型：finite，infinitive，gerund，base，present-participle，past-participle 以及 passive-participle）。这样，verb 可定义如下（这里按照惯例把类型记录在 AVM 的顶部，或者记录在左括号的左下部）：

$$\begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{agr} \\ \text{VFORM} & \textbf{vform} \end{bmatrix}$$

与此相反，复杂类型 noun 可以用特征 AGREE 来定义，但是不用特征 VFORM 来定义：

$$\begin{bmatrix} \textbf{noun} & \\ \text{AGREE} & \textbf{agr} \end{bmatrix}$$

除了运算组成成分特征合一的值之外，只需要能够对两个结构的类型也能进行合一运算，把合一运算提升到能够运算类型化的特征结构就可以了。

$$\begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{1st} \\ \text{VFORM} & \textbf{gerund} \end{bmatrix} \sqcup \begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{sg} \\ \text{VFORM} & \textbf{gerund} \end{bmatrix} = \begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{1-sg} \\ \text{VFORM} & \textbf{gerund} \end{bmatrix}$$

复杂类型也是类型层级体系的一部分。复杂类型的子类型继承了其父代的全部特征以及对其值的约束。例如，Sanfilippo(1993)使用类型层级体系来给词表的层级结构进行编码。图 11.14 是这个层级体系的一个很小的部分，这个部分为要求句子补语的动词的各个次范畴建立模型；它们可以分为及物动词和不及物动词两种。及物动词带直接宾语（*ask yourself whether you have become better informed*），不及物动词不带直接宾语（*Monsieur asked whether I wanted to ride*）。类型 trans-comp-cat 可导入所要求的直接宾语，这个直接宾语被约束为类型 noun-phrase，而类型 sbase-comp-cat 则导入动词的基本形式补语（光杆的词干），并约束它的 vform 必须是动词的基本形式。

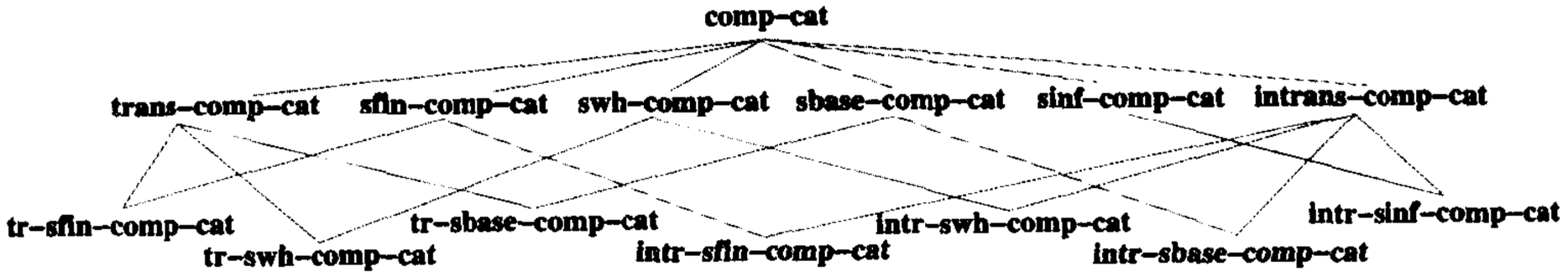


图 11.14　动词类型 verb-cat 的类型层级体系的一部分，说明了类型 comp-cat 的子类型。它们全都是要求句子补语的动词的次范畴(Sanfilippo, 1993)

11.6.1　类型的扩充

如果允许带默认值（default）的继承，就可以扩充类型化的特征结构。前面说过，默认值系统主要用于词汇类型的继承，以便进行一般化的编码和处理不十分正规的例外。在早期的默认值合一运算中，运算是与顺序有关的，而且这种运算基于优先的并运算（priority union operation）(Kaplan, 1987)。最近的一些体系结构，例如 Lascarides and Copestake(1997)的用于类型化特征结构的默认值合一，则是与顺序无关的，读者可以参阅 Young and Rounds(1993)，这样的研究与 Reiter 的默认值逻辑（default logic）有联系(Reiter, 1980)。

很多基于合一的语法理论，包括HPSG(Pollard and Sag, 1987, 1994)和LFG(Bresnan, 1982)，除了使用继承关系来处理词汇的一般化问题之外，还使用了一种附加的机制：词汇规则（lexical rule）。词汇规则使用一个小词汇量的词表，因此是一种冗余度很少的词表，词汇的扩充是通过规则自动进行的。Jackendoff(1975)最先提出了词汇规则的思想，关于现代的词汇规则可参阅 Pollard and Sag(1994)，关于计算复杂性的讨论可参阅 Carpenter(1991)，关于实现词汇规则的一些最新研究，可参阅 Meurers and Minnen(1997)。有些学者提出使用类型层级体系来代替词汇规则，或者添加一些抽象类型和某些析取规则(Krieger and Nerbonne, 1993)，或者对类型进行次级规范（type underspecification）和动态分类（dynamic typing）。其中，经过次级规范的类型结合起来，可以在线地（online）形成新的类型(Koenig and Jurafsky, 1995)。

类型也可以用来表示组成关系。前面的规则（11.13）使用了一般的短语结构规则模板，再通过路径等式的方法加上特征。不使用这样的办法也可以把整个短语结构规则表示为类型。为此，根据 Sag and Wasow(1999)提出的方法，我们采用 phrase 这个类型，这个类型有一个称为 DTRS（"daughters"，"女儿"）的特征，这个特征的值是一个 phrase 的表（list）。例如，短语 *I love New York* 可以表示如下（只说明 DTRS 特征）：

$$\left[\begin{array}{l}\textbf{phrase}\\ \text{DTRS}\left\langle\left[\begin{array}{l}\text{CAT PRO}\\ \text{ORTH I}\end{array}\right],\left[\begin{array}{l}\text{CAT VP}\\ \text{DTRS}\left\langle\left[\begin{array}{l}\text{CAT V}\\ \text{ORTH LOVE}\end{array}\right],\left[\begin{array}{l}\text{CAT NP}\\ \text{ORTH NEW YORK}\end{array}\right]\right\rangle\end{array}\right]\right\rangle\end{array}\right]$$

11.6.2 合一的其他扩充

除了类型之外，合一还有很多其他方面的扩充，其中包括路径不等式（path inequation）(Moshier, 1988; Carpenter, 1992; Carpenter and Penn, 1994)、否定（negation）(Johnson, 1988, 1990)、集合值特征（set-valued feature）(Pollard and Moshier, 1990)、析取（disjunction）(Kay, 1979; Kasper and Round, 1986)等。在一些合一系统中，这些运算被用到特征结构中。与此相反，Kasper and Round(1986)等则在一种用于描写特征结构的元语言中实现了这些运算。这种思想来自Pereira and Shieber(1984)以及Kaplan and Bresnan(1982)更早期的工作，他们都把描写特征结构的元语言和实际的特征结构本身区分开来。这样的描述可以使用否定和析取来描写特征结构的集合（也就是说，一个确定的特征不一定必须包含一个确定的值，也可以包含值的集合中的任意一个值），但是满足这种描述的特征结构的实际例子不能有否定或析取的值。

11.7 小结

本章介绍了特征结构和把特征结构彼此结合起来的合一运算。

- 特征结构是特征-值偶对的集合，其中特征是来自某个有限集合的不可分析的原子符号，而特征的值或者是原子符号，或者是特征结构。特征结构或者用属性-值矩阵（AVM）表示，或者用有向非成圈图（DAG）表示。在DAG中，特征是有向的、有标记的边，而特征的值是图中的结点。
- 合一是一种运算，这种运算既可以结合信息（把两个特征结构的信息内容合并），也可以比较信息（拒绝合并不相容的信息）。
- 增强和提升短语结构规则的方法有两个：一个方法是使用特征结构，另一个方法是使用特征约束，特征约束要表示短语结构规则成分的特征结构之间的关系。次范畴化约束可以表示为关于中心语动词（或其他谓词）的特征结构。一个动词的次范畴化成分或者可以出现在动词短语中，或者可以离开动词很远，成为一种远距离依存关系。
- 特征结构可以类型化。所形成的类型化的特征结构把约束放在给定特征结构能够取值的类型上，也可以把它们组织到类型层级体系中，以捕捉贯穿这些类型的一般化的东西。

11.8 文献和历史说明

从源头上说，在语言学理论中使用特征起源于音系学。Anderson(1985)指出，Jakobson(1939)首先把特征（称为区别特征）作为他的理论中的一种知识本体类型（ontological type）来使用，在他之前曾经有Trubetskoi(1939)等使用过特征这个术语。此后不久就开始在语义学中使用特征，关于语义的组成分析，参见第16章。句法中的特征是20世纪50年代建立起来的，Chomsky(1965)使特征得到了推广。

语言学中的合一运算是分别由Kay(1979)和Colmerauer(1970, 1975)独立地发展起来的，Colmerauer提出了“合一”这个术语，参见1.6节。Kay和Colmerauer都从事机器翻译研究，都试图探索一种形式化方法把语言信息结合起来，并且要求这种结合是可逆的。Colmerauer的原来的Q系统是一个自底向上的剖析器，它是建立在包含逻辑变量的一系列重写规则基础上的。使用Q系统，Colmerauer设计了一个英语到法语的机器翻译系统。这样的重写规则是可逆的，既可以用于剖析，也可以用于生成。Colmerauer，Fernand Didier，Robert Pasero，Philippe Roussel和Jean Trudel

设计了 Prolog 语言。Prolog 语言的基础是扩充为基于 Robinson(1965)的“决定原则”（resolution principle）的Q系统。在Prolog语言的基础上，他们还实现了一个法语分析器(Colmerauer and Roussel, 1996)。在自然语言中使用 Prolog 与合一，提出了定子句语法（Definite Clause Grammar）。定子句语法的基础是 Colmerauer(1975)的变形语法（metamorphosis grammar），而定子句语法本身是由 Pereira and Warren(1980)提出和命名的。与此同时，Martin Kay 和 Ron Kaplan 在研制扩充转移网络（Augmented Transition Network，简称 ATN）语法。ATN 是经过改进的递归转移网络（Recursive Transition Network，简称RTN），其中的结点用特征寄存器来加以扩充。在用ATN分析英语被动式句子时，第一个NP首先被指派到主语寄存器中，然后当遇到被动式动词时，它的值就被移动到宾语寄存器中。为了使这个过程成为可逆的，他们对寄存器的指派进行限制，使得某些寄存器只能填写一次。因此，在写了一次之后就不能再重新写了。他们还研究了逻辑变量的概念，但是没有实现它。Kay原来的合一算法是针对特征结构而不是针对项目设计的(Kay, 1979)。在 11.5 节中介绍的把合一运算结合到 Earley 算法中的内容主要依据 Shieber(1985b)。

关于合一算法的详细介绍参见 Shieber(1986)。关于合一运算的多学科介绍参见 Knight(1989)。

在 KL-ONE 知识表示系统的背景下(Brachman and Schmolze, 1985)，Bobrow and Webber(1980)首先提出了语言知识的继承和适切性条件。一些学者提出了没有适切性条件的简单继承关系，早期使用这种继承关系的是 Jacobs(1985, 1987)和 Flickinger et al.(1985)。Aït-Kaci(1984)从逻辑程序学界借用了合一运算中的继承概念。包括继承和适切性条件的特征结构的类型化，是由 Calder(1987)，Pollard and Sag(1987)和 Elhadad(1990)首先提出的。类型化的特征结构是由 King(1989)和 Carpenter(1992)形式化的。关于在语言学中如何使用类型继承，特别是关于如何捕捉词汇的一般化内容的问题，存在着大量的文献；除了前面讨论过的文章之外，有兴趣的读者可以参阅更多的文献；关于 DATR 语言的描述（这种语言是为了定义语言知识表示的继承网络而设计的），参阅 Evans and Gazdar(1996)；关于在依存语法中继承关系的使用情况，参阅 Fraser and Hudson(1992); Daelemans et al.(1992)对此做了一个总的评述。通过类型化的特征结构实现基于约束的语法的形式化方法和系统，包括使用 TDL 语言的 PAGE 系统(Krieger and Schäfer, 1994)，ALE(Carpenter and Penn, 1994)和 ConTroll(Götz et al., 1997)。

基于合一的语法理论有词汇功能语法（Lexical Functional Grammar，简称LFG）(Bresnan, 1982)，中心语驱动的短语结构语法（Head-Driven Phrase Structure Grammar，简称 HPSG）(Pollard and Sag, 1987, 1994)，构造语法（Construction Grammar）(Kay and Fillmore, 1999)和合一范畴语法（Unification Categorial Grammar）(Uszkoreit, 1986)。

第12章　词汇化剖析与概率剖析

Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth...
Robert Frost, *The Road Not Taken* ①

Damon Runyon的短篇小说中的人物喜欢对任何问题都打赌，正如Runyon在《Sarah Brown小姐的田园诗》中关于Sky Masterson所说的那样，他们背靠背地从奇数中猜测结果是幺点的概率，他们也猜测把一颗花生米从第二个地基扔到房屋平台上的概率。“根据足够多的知识，我们能够求出任何事物的概率”，这是语言处理中的一条格言。前面的三章介绍了关于句法结构以及剖析的复杂模型。在这一章中将说明，我们也可以给这些复杂的句法信息建立概率模型，并且在有效的概率剖析中使用这些概率信息。

概率语法和概率剖析器有什么用处呢？概率剖析器的一个重要贡献是它能够进行歧义消解（disambiguation）。我们知道，句子可能具有很多歧义，第10章中介绍的Earley算法可以有效地表示这种歧义，但是它没有给我们提供手段来消解歧义。概率语法可以给我们提供对这个问题的解决办法：从歧义中选择概率最大的解释。由于歧义非常普遍，因此概率剖析器在大多数剖析或自然语言理解的工作中起着重要的作用。

概率语法的另一个重要用处是为语音识别或增强通信建立语言模型。我们已经看到，在帮助语音识别方面，在预测即将来临的单词方面，在帮助约束单词的搜索方面，*N*元语法都是非常重要的。更复杂的概率语法可以给语音识别提供更多的预测能力。因为人也要像语音识别器那样处理同样的歧义问题，所以可以寻找心理学的证据来证明，在人的语言处理中人们也使用类似的概率语法，这也是非常有意义的事情。

把句法的复杂结构模型和概率模型结合起来，是概率剖析领域的一个前沿问题。因为这个问题很新，还没有一个单独的模型成为标准，但是上下文无关语法已成为了非概率句法的一个标准。我们将介绍对于上下文无关语法在概率上的一些增强方法，说明如何使用它们进行剖析，对该领域应该采取的方向提出建议，以揭示这个领域的面貌。本章首先介绍概率上下文无关语法（Probabilistic Context-Free Grammar，简称PCFG），这种语法是上下文无关语法在概率方面的提升，同时还介绍CYK算法（CYK algorithm），这是用于PCFG剖析的一种标准的动态规划算法。然后，我们介绍对于PCFG在两个方面的扩充，以处理概率次范畴化信息以及处理概率词汇依存关系（lexical dependency），为剖析器的评测给出一个评测基准，然后介绍一些先进的研究结果，并对人的语言处理进行某些讨论。

---

① 这首诗的意思是：
道路在黄树林中分叉为两条，遗憾的是，我不能同时走两条路，
我是一个单独的旅行者，只好茫然地久久伫立，我尽自己之所能向下看其中的一条路，
看着它倾斜在大树下的丛林中……

12.1　概率上下文无关语法

对于上下文无关语法的最简单的提升就是概率上下文无关语法(PCFG),这种语法又称为随机上下文无关语法（ Stochastic Context-Free Grammar，简称 SCFG ），这种语法最早是由 Booth(1969)提出来的。

我们知道，上下文无关语法 G 是由四个参数(N, Σ, P, S)来定义的：

1. 非终极符号（或变量）的集合 N
2. 终极符号的集合 Σ（与 N 不相交）
3. 产生式的集合 P，每个产生式的形式为 $A \to \beta$，其中 A 是单个的非终极符号，β 是从无限的符号串（$\Sigma \cup N$)* 中的符号构成的符号串
4. 指定的初始符号 S

概率上下文无关语法给产生式P中的每个规则都加上了一个条件概率，从而增强了这些规则：

$$A \to \beta \ [p] \tag{12.1}$$

这样，PCFG 就是一个五元组 $G = (N, \Sigma, P, S, D)$，其中 D 的功能是给 P 中的每个规则指派一个概率。这个功能表示，把给定的非终极符号 p 展开为符号序列 β 时的概率，这个概率通常表示为：

$$P(A \to \beta)$$

或者表示为：

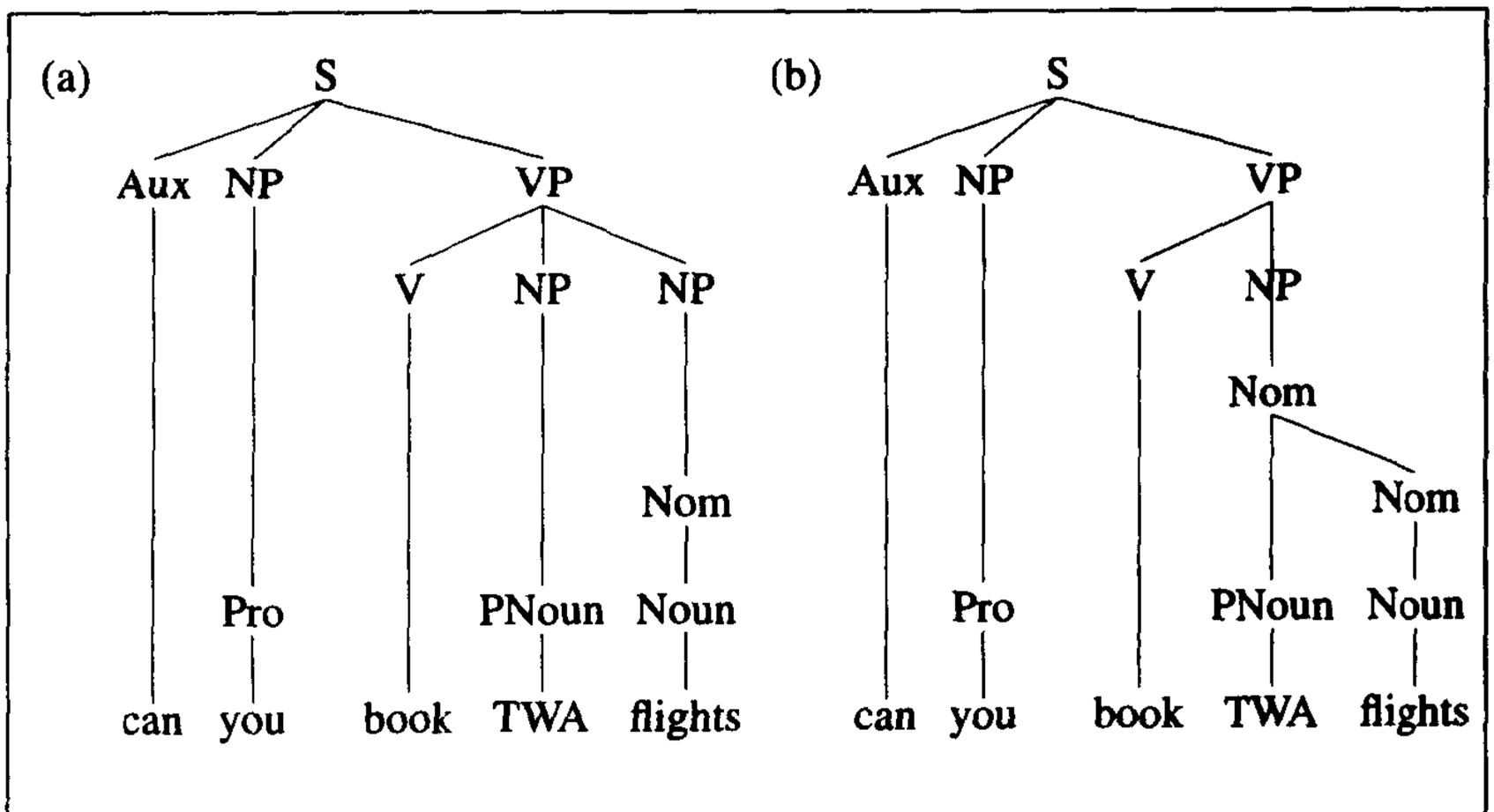
$$P(A \to \beta | A)$$

从形式上讲，这是对于给定的左手边的非终极符号A进行给定的展开时的条件概率。因此，如果我们考虑一个非终极符号的所有可能展开，它们的概率之和就必定等于1。图12.1中英语的一个微型语法的PCFG样本，其中只有三个名词和三个动词。注意，一个非终极符号的一切可能展开概率的总和为1。显而易见，在真实的语法中，每个非终极符号存在着比这多得多的规则，因此任何特定规则的概率都比这个微型语法中的规则的概率小一些。

$S \to NP\ VP$	[0.80]	$Det \to that$ [0.05] | the [0.80] | a [0.15]	
$S \to Aux\ NP\ VP$	[0.15]	$Noun \to book$	[0.10]
$S \to VP$	[0.05]	$Noun \to flights$	[0.50]
$NP \to Det\ Nom$	[0.20]	$Noun \to meal$	[0.40]
$NP \to Proper\text{-}Noun$	[0.35]	$Verb \to book$	[0.30]
$NP \to Nom$	[0.05]	$Verb \to include$	[0.30]
$NP \to Pronoun$	[0.40]	$Verb \to want$	[0.40]
$Nom \to Noun$	[0.75]	$Aux \to can$	[0.40]
$Nom \to Noun\ Nom$	[0.20]	$Aux \to does$	[0.30]
$Nom \to Proper\text{-}Noun\ Nom$	[0.05]	$Aux \to do$	[0.30]
$VP \to Verb$	[0.55]	$Proper\text{-}Noun \to TWA$	[0.40]
$VP \to Verb\ NP$	[0.40]	$Proper\text{-}Noun \to Denver$	[0.40]
$VP \to Verb\ NP\ NP$	[0.05]	$Pronoun \to you$ [.40] | I [.60]	

图 12.1　图 10.2 的英语微型语法和词表用概率增强后形成的 PCFG。这些概率不是从语料库统计出来的，只是为了说明这种语法而编出来的

怎样来使用这些概率呢？PCFG可以用来估计关于一个句子及其剖析树的有用概率的数量。例如，一个PCFG可以对于一个句子S的每个剖析树T（也就是每个推导结果）都指派一个概率。PCFG的这个性质在歧义消解（disambiguation）中是非常有用的。例如，我们来研究歧义句子Can you book TWA flights 的两个剖析结果。一个意思是 Can you book flights on behalf of TWA（你能以 TWA 公司的名义预订飞机票吗？），另一个意思是 Can you book flights run by TWA（你能预订 TWA 公司经营的飞机票吗？），这两个剖析树如图 12.2 所示。



Rules			P	Rules			P
S	→	Aux NP VP	0.15	S	→	Aux NP VP	0.15
NP	→	Pro	0.40	NP	→	Pro	0.40
VP	→	V NP NP	0.05	VP	→	V NP	0.40
NP	→	Nom	0.05	NP	→	Nom	0.05
NP	→	PNoun	0.35	Nom	→	PNoun Nom	0.05
Nom	→	Noun	0.75	Nom	→	Noun	0.75
Aux	→	Can	0.40	Aux	→	Can	0.40
NP	→	Pro	0.40	NP	→	Pro	0.40
Pro	→	you	0.40	Pro	→	you	0.40
Verb	→	book	0.30	Verb	→	book	0.30
PNoun	→	TWA	0.40	Pnoun	→	TWA	0.40
Noun	→	flights	0.50	Noun	→	flights	0.50

图 12.2 歧义句子的两个剖析树。剖析树(a)对应的意思是 Can you book flights on behalf of TWA，剖析树(b)对应的意思是 Can you book flights which are run by TWA

一个特定的剖析T的概率定义为在该剖析树中用来展开每个结点n的所有规则r的概率的乘积：

$$P(T,S)=\prod_{n\in T}p(r(n)) \tag{12.2}$$

作为结果的概率$P(T,S)$既是剖析和句子的联合概率，又是剖析$P(T)$的概率。怎样证实这个公式的正确性呢？

首先，根据联合概率的定义有：

$$P(T,S)=P(T)P(S|T) \tag{12.3}$$

但是，因为剖析包含了句子中的所有单词，所以 $P(S|T)$等于 1。因此有：

$$P(T,S) = P(T)P(S|T) = P(T) \tag{12.4}$$

图 12.2 中的每个剖析树的概率的计算，只要把在推导中使用的每个规则概率相乘就可以得到计算结果。例如，图 12.2(a)中左侧的剖析树（称为 T_l）的概率以及图 12.2(b)中右侧的剖析树（称为 T_r）的概率可以计算如下：

$$\begin{aligned} P(T_l) &= 0.15*0.40*0.05*0.05*0.35*0.75*0.40*0.40*0.40 \\ &\quad *0.30*0.40*0.50 \\ &= 1.5\times10^{-6} \end{aligned} \tag{12.5}$$

$$\begin{aligned} P(T_r) &= 0.15*0.40*0.40*0.05*0.05*0.75*0.40*0.40*0.40 \\ &\quad *0.30*0.40*0.50 \\ &= 1.7\times10^{-6} \end{aligned} \tag{12.6}$$

我们可以看出，在图 12.2(b)中右侧的剖析树具有比较高的概率。如果歧义消解算法选择具有最大 PCFG 概率的剖析，那么这个剖析便可以通过这样的歧义消解算法选择正确结果。

可以看出，选择具有最大概率的剖析是进行歧义消解的正确方法。现在，让我们把这样的直觉加以形式化。歧义消解算法在句子 S 的剖析树集合（我们称之为 $\tau(S)$）中选择对于这个句子 S 的最佳的树作为剖析结果。我们想得到的是对于给定的句子 S 的最佳剖析树 T。

$$\hat{T}(S) = \underset{T\in\tau(S)}{\operatorname{argmax}} P(T|S) \tag{12.7}$$

根据定义，概率 $P(T|S)$可以改写为 $P(T,S)/P(S)$，这样可以得出：

$$\hat{T}(S) = \underset{T\in\tau(S)}{\operatorname{argmax}} \frac{P(T,S)}{P(S)} \tag{12.8}$$

因为我们要最大限度地考虑同一个句子可能有的一切剖析树，所以对于每个树，$P(S)$将是一个常数，我们可以删除它，得到：

$$\hat{T}(S) = \underset{T\in\tau(S)}{\operatorname{argmax}} P(T,S) \tag{12.9}$$

还有，由于我们曾经说明 $P(T,S) = P(T)$，所以选择最佳剖析的最后的等式巧妙地简化为选择具有最大概率的剖析：

$$\hat{T}(S) = \underset{T\in\tau(S)}{\operatorname{argmax}} P(T) \tag{12.10}$$

PCFG 的另一个特性是它可以给构成句子的单词符号串指派一个概率。在语音识别、拼写检查和增强通信中，这个特性对于语言建模（language modeling）有重要意义。非歧义句子的概率等于 $P(T,S) = P(T)$，或者说这个概率恰好是该句子的单个剖析树的概率。歧义句子的概率等于该句子所有剖析树的概率之和：

$$P(S) = \sum_{T\in\tau(S)} P(T,S) \tag{12.11}$$

$$= \sum_{T\in\tau(S)} P(T) \tag{12.12}$$

PCFG 对语言建模的另一个有用特征是它可以给句子中的子符号串指派一个概率。例如，Jelinek and Lafferty(1991)提出了能够有效计算一个句子前面部分（prefix）的概率的算法。这个概

率也就是语法生成句子的初始子符号串为 $w_1 w_2 \cdots w_i$ 的概率。Stolcke(1995)说明了怎样把标准的Earley算法提升到能够计算这些句子前面部分的概率，Jurafsky et al.(1995)描写了采用这种算法作为语言模型在语音识别中的应用。

在一个PCFG中，如果一种语言的所有句子的概率之和为1，就可以说这个PCFG是坚固的(consistent)。有些递归规则会引起语法变得不坚固，因为这时它对某些句子要进行无限循环的推导。例如，概率为1的规则 $S \rightarrow S$ 将会导致概率量的丧失，因为推导永远不会终止。关于坚固语法和非坚固语法的详细情况，可以参阅Booth and Thompson(1973)。

12.1.1 PCFG的概率CYK剖析

PCFG的剖析问题是对于一个给定的句子产生最佳剖析树的问题，也就是计算：

$$\hat{T}(S) = \underset{T \in \tau(S)}{\operatorname{argmax}} P(T) \tag{12.13}$$

幸运的是，计算最佳剖析的算法只是标准剖析算法的简单扩充。第10章中介绍了怎样使用Earley算法来对于给定的输入句子和给定的上下文无关语法找出所有剖析。我们完全可能提升Earley算法，使它能够计算每个剖析的概率，从而找出最佳的剖析。但是，这里不介绍概率Earley算法，而是介绍概率CYK算法(Cocke-Younger-Kasami algorithm)。之所以这样做，一是因为概率Earley算法介绍起来比较复杂，二是因为CYK算法很值得我们理解和学习，而我们现在还没有学习过这种算法。因此，关于概率Earley算法的介绍，建议读者阅读Stolcke(1995)。

Earley算法主要是一种自顶向下的剖析算法，使用动态规划表来有效地存储中间结果；CYK算法主要是一种自底向上的剖析算法，也使用同样的动态规划表。CYK算法的这种自底向上的性质，使得它在处理词汇化的语法时非常有效，这一点以后将会看到。

概率CYK算法首先是由Ney(1991)描述的，但是现在采用的概率CYK算法的版本来自Collins(1999)和Aho and Ullman(1972)。首先，我们要假定PCFG是具有Chomsky范式的。从9.10节中可以知道，如果一个语法是 ε-free 的，并且如果每个产生式的形式或者为 $A \rightarrow B\ C$，或者为 $A \rightarrow a$，那么这个语法就是具有Chomsky范式(CNF)的语法。CYK算法假定如下的输入、输出和数据结构：

- **输入**
  - Chomsky范式的PCFG $G = (N, \Sigma, P, S, D)$。假定非终极符号 $|N|$ 的索引号为 $1, 2, \cdots, |N|$，初始符号的索引号为1。
  - n 个单词为 $w_1 \cdots w_n$。
- **数据结构** 动态规划数组 $\pi[i, j, a]$ 表示跨在单词 $i \cdots j$ 上的非终极索引号为 a 的成分的最大概率。在这个区域上的反向指针用于存储剖析树中成分之间的链接。
- **输出** 最大概率剖析将是 $\pi[1, n, 1]$：剖析树的根是 S，剖析树跨在单词 $w_1 \cdots w_n$ 构成的整个符号串上。

正如其他动态规划算法那样(最小编辑距离算法、向前算法、Viterbi算法和Earley算法)，CYK算法采用归纳法来填充概率数组。为了便于描写，我们用 w_{ij} 来表示从单词 i 到单词 j 的单词符号串。根据Aho and Ullman(1972)，我们有：

- **基底** 考虑长度为1的输入符号串(也就是一个单词 w_i)。在Chomsky范式中，给定的非终极符号 A 展开为一个单词 w_i 的概率必定只来自规则 $A \rightarrow w_i$，因为当且仅当 $A \rightarrow w_i$ 是一个产生式时，有 $A \overset{*}{\Rightarrow} w_i$)。

● **递归**　对于长度大于 1（length > 1）的单词符号串，当且仅当至少存在一个规则 $A \rightarrow BC$ 以及某个 k，$1 \leqslant k < j$ 时，$A \stackrel{*}{\Rightarrow} w_{ij}$，使得 B 推导出 w_{ij} 的开头 k 个符号串，C 推导出 w_{ij} 的后面 $j-k$ 个符号串。因为这些符号串都比原来的符号串 w_{ij} 短，它们的概率已经被存储在矩阵 π 中，我们把这两个片断的概率相乘，计算出 w_{ij} 的概率。当然，这时 w_{ij} 也可能会出现多个剖析，所以要选择在所有可能的剖析中概率最大的剖析（也就是在所有可能的 k 值和所有可能的规则中进行选择）。

图 12.3 给出了这个概率 CYK 算法的伪代码，也取自 Collins(1999)和 Aho and Ullman(1972)。

```
function CYK(words,grammar) returns best_parse

  Create and clear p[num_words,num_words,num_nonterminals]

  # base case
  for i = 1 to num_words
     for A = 1 to num_nonterminals
        if A → w_i is in grammar then
           π[i,i,A] = P(A → w_i)

  # recursive case
  for j = 2 to num_words
     for i = 1 to num_words-j+1
        for k = 1 to j-1
           for A = 1 to num_nonterminals
           for B = 1 to num_nonterminals
           for C = 1 to num_nonterminals
              prob = π[i,k,B] × p[i+k,j-k,C] × P(A→BC)
              if (prob > π[i,j,A]) then
                 π[i,j,A] = prob
                 B[i,j,A] = {k,A,B}
```

图 12.3　概率 CYK 算法。对于给定的具有 Chomsky 范式规则 *num_rule* 的 PCFG 语法，该算法用于找出由单词 *num_words* 组成的符号串的最大概率剖析。B 是反向指针的数组，用于恢复最佳的剖析(Collins, 1999; Aho and Ullman, 1972)

12.1.2　PCFG 概率的学习

PCFG的概率是从哪里来的？给语法指派概率有两种途径。最简单的途径是使用句子已经得到剖析的语料库。这样的语料库称为树库（treebank）。例如，语言数据联盟（Linguistic Data Consortium）发布的 Penn 树库(Marcus et al., 1993)包括 Brown 语料库的剖析树，规模有 1 百万单词，语料来自《华尔街日报》，部分语料来自 Switchboard 语料库。给定一个树库，一个非终极符号的每个展开概率都可以通过展开发生的次数进行计算，然后将其归一化。

$$P(\alpha \rightarrow \beta|\alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\sum_{\gamma}\text{Count}(\alpha \rightarrow \gamma)} = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)} \tag{12.14}$$

当没有现成的树库可供使用时，计算 PCFG 概率所需的数据可以通过先剖析一个语料库来生成。如果句子是没有歧义的，做法就很简单：剖析语料库，在剖析中为每个规则都添加一个计数器，然后进行归一化处理，就可以得到概率。但是，由于大多数句子都是有歧义的，实际上我们

必须为一个句子的每个剖析都分别保持一个记数，并且根据剖析的概率给每个局部的记数加权。这种计算的标准算法称为向内－向外算法（Inside-Outside algorithm），这个算法是Baker(1979)作为第7章中介绍过的向前－向后算法的一般化算法提出的。这种算法的完全描述可参阅Manning and Schütze(1999)。

12.2 PCFG的问题

由于概率上下文无关语法是上下文无关语法的自然扩充，这样的语法在概率估计方面会出现一些问题。正因为这些问题，所以当前大多数概率剖析模型都使用某些增强了的PCFG，而不使用普通的PCFG。本节将总结PCFG在模拟结构依存（structural dependency）和模拟词汇依存（lexical dependency）中的问题。

PCFG的一个问题来自基本的独立性假设（independence assumption）。根据定义，CFG假定任何一个非终极符号的展开与任何其他非终极符号的展开是独立的。这种独立性假设也带到了PCFG中；每个PCFG规则被假定为独立于其他每个规则，这样，规则的概率才能相乘。然而，英语句法统计检查的结果说明，有时一个结点展开的选择取决于该结点发在剖析树中的位置。例如，我们来研究代词与实词名词短语在句子中的不同位置的分布情况。从Kuno(1972)开始，很多语言学家指出，在英语中（在很多其他语言中也是如此）存在着一个强烈的倾向：一个句子的句法主语往往是代词(Givón, 1990)。代词是谈论旧信息的一种手段，而非代词（实词性）的名词或名词短语往往用来引入新信息。例如，Francis等人指出，在Switchboard语料库的31 021个陈述句的主语中，91%都是代词，参见例句（12.15a），只有9%是实词性的名词或名词短语，参见例句（12.15b）。反之，在7489个宾语中，只有34%是代词，参见例句（12.16a），而66%是实词性的名词或名词短语，参见例句（12.16b）。

(a) **She's** able to take her baby to work with her. （12.15）

(b) Uh, **my wife** worked until we had a family.

(a) Some laws absolutely prohibit **it**. （12.16）

(b) All the people signed **confessions**.

通过规则 $NP \rightarrow Pronoun$，我们可以把*NP*展开为代词，通过规则 $NP \rightarrow Det\ Noun$，我们可以把*NP*展开为实词性名词或名词短语。如果*NP*展开为代词对应于*NP*展开为实词性名词或名词短语的概率取决于*NP*究竟是主语还是宾语，那么，我们就可以捕捉到这样的依存关系，然而这种概率依存关系恰恰是PCFG所不允许的。

PCFG更重要的问题是这种语法缺乏对单词的敏感性。PCFG中的词汇信息只能通过前终极结点（*Verb, Noun, Det*）展开为单词的概率来表示。但是，还有一些其他词汇依存关系对于句法概率的模拟也很重要。例如，一些研究者指出，在选择有歧义的介词短语附着的正确剖析时，词汇信息就起着重要作用(Ford et al., 1982; Whittemore et al., 1990; Hindle and Rooth, 1991; et al.)。我们来研究来自Hindle and Rooth(1991)的例子：

Moscow sent more than 100, 000 soldiers into Afghanistan ... （12.17）

在这个句子中，介词短语*into Afghanistan*或者附着于*more than 100, 000 soldiers*这个*NP*，或者附着于以*sent*为中心语的动词短语。在PCFG中，附着关系的选择归结为在下面两个规则之间的选择：一个规则是 $NP \rightarrow NP\ PP$（*NP*附着），一个规则是 $VP \rightarrow VP\ PP$（*VP*附着）。这两个规则的概率依

赖于训练语料库；Hindle and Rooth(1999)报告，根据对*AP*新闻专线1300万单词的语料库统计，*NP*附着占67%，*VP*附着占33%；Collins(1999)报告，在混合了《华尔街日报》和《IBM计算机手册》的语料库中，*NP*附着占了52%。无论*NP*附着的优先性是52%还是67%，关键的是在PCFG中这个优先性是纯结构的，而且对于所有动词都是相同的。

但是，在例句（12.17）中，*PP*的正确附着却是在动词上，是*VP*附着。在这种情况下，因为动词*send*的次范畴需要表示“方向”，而“方向”正好通过介词*into*来表示。在Penn树库的Brown语料库和WSJ语料库关于动词的语料中，与主要动词*send*有关的介词*into*的所有歧义的PP附着全都附着于动词。在这种情况下，只有针对不同的动词建立不同的词汇依存关系（lexical dependency）的统计模型，才能正确地选择剖析的结果。

词汇依存关系是选择正确剖析的关键，这一点还表现在联合结构歧义（coordination ambiguity）的问题上。图12.4是来自Collins(1999)的一个例子，短语*dogs in houses and cats*有两个不同的剖析。由于*dogs*在语义上更容易与*cats*结合，而不与*houses*结合（并且也由于*dogs*不可能处于*cats*的体内），所以*[dogs in [NP houses and cats]]*在直觉上就是不自然的，因而不能选择为剖析结果。然而，图12.4中的两个剖析使用的PCFG规则完全相同，所以PCFG将给它们指派相同的概率。

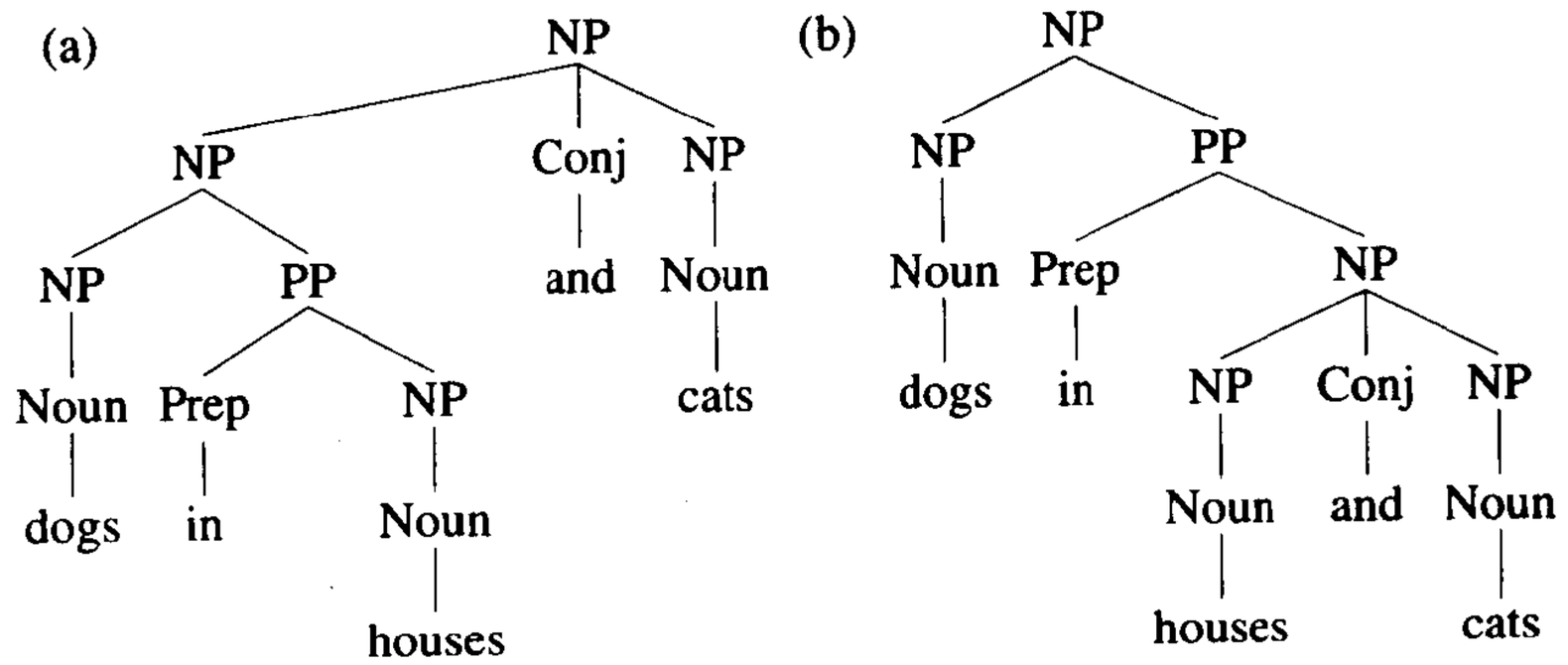


图12.4　联合结构歧义的一个例子。尽管左边的结构在直觉上是正确的，但是PCFG将给左右两个结构都指派相同的概率，因为两个结构使用的规则完全相同(Collins，1999)

总而言之，概率上下文无关语法作为句法的一个概率模型存在着一系列不足。下一节将简要介绍当前提升PCFG的一些新方法，以处理这些问题。

12.3　概率词汇化的CFG

在第11章中讲过，每个句法成分可以与一个词汇中心语（lexical head）联系起来。每个成分都有一个中心语的这种思想可以追溯到Bloomfield(1914)，但首先使用这种思想来扩充PCFG模型的是Black et al.(1992)。像Charniak(1997)和Collins(1999)那样在剖析器中使用词汇中心语的概率表示，比第11章介绍的那种复杂的中心语特征模型简单一些。在这种比较简单的概率表示中，剖析树中的每个非终极符号都要标上一个单词作为它的词汇中心语。图12.5中是这种树的一个例子，这个例子来自Collins(1999)，其中每个非终极符号都标上了它们的中心语。*workers dumped sacks into a bin*这个句子是WSJ语料库中的句子经过简化得到的。

为了生成这样的树，每个PCFG的规则必须增强，使它能够鉴别一个右手边的成分是中心语女儿结点。然后，一个结点上的中心词被置为这个中心语女儿结点的中心词。从教材的例子中，找出这些中心语女儿结点是很简单的（*NN*是*NP*的中心语），但是对于大多数短语来说，这却是很复杂

的，当然也是容易引起争议的。在一个不定动词短语中，它的中心语是标补语*to*，还是动词呢？现代语言学的句法理论一般都包括确定中心语这样的内容(Pollard and Sag, 1994)。Collins(1999)也给由Magerman修订的Penn树库语法提出了一套实用的中心语规则；例如查找NP的中心语的规则是：如果已经标注为POS（possessive，主有代词），就返回NP中最后的单词，否则就在NP中从右到左搜索第一个儿子结点，这些结点的标记是NN，NNP，NNPS，NNS，NX，POS或JJR；如果从右到左搜索不成功，就从左到右搜索标记为NP的第一个儿子结点。

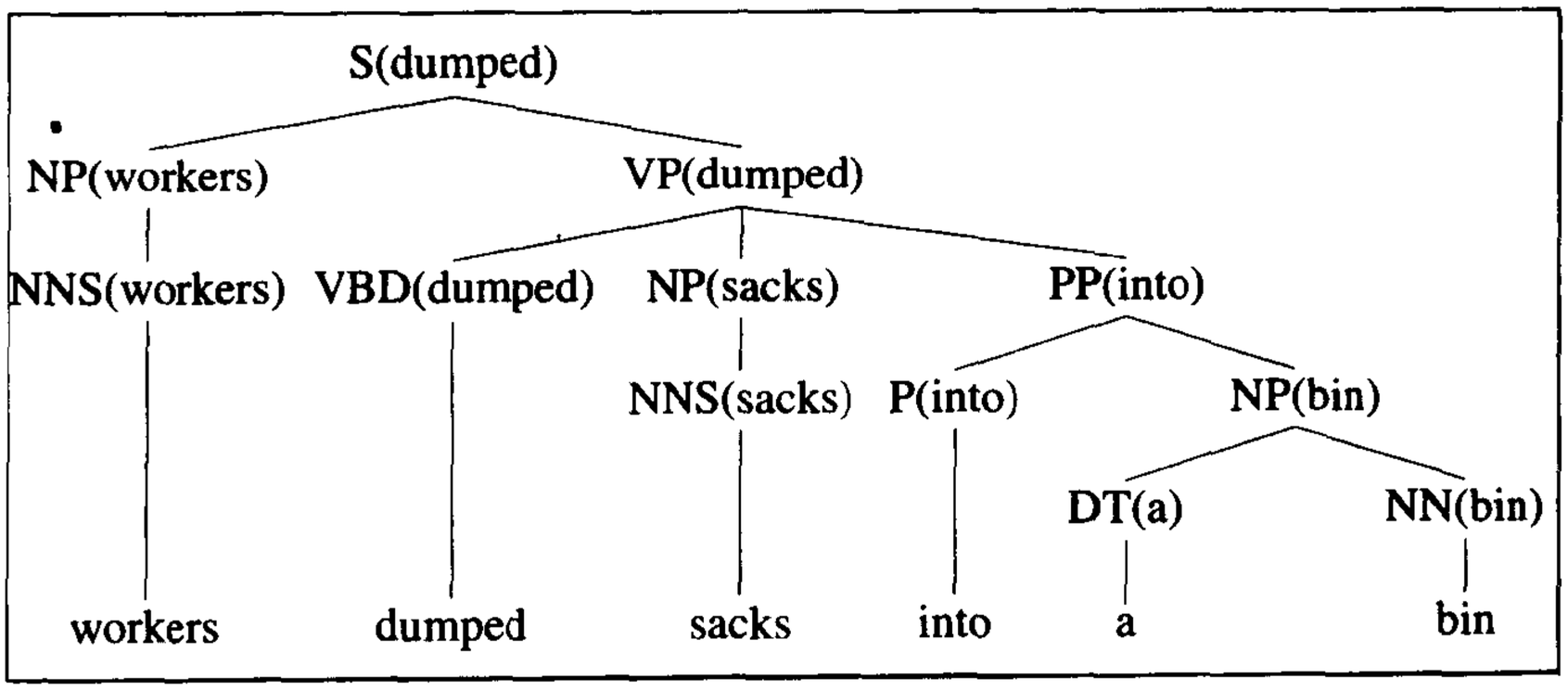


图 12.5　词汇化的树，来自 Collins(1999)

一个办法是把这些中心语特征看成合一语法中简化的中心语特征，这种简化的中心语特征不使用非常复杂的重入特征值，只允许一个属性有一个单独的值，而这个单独的值来自一个有限的集合（事实上就是词汇中单词的集合）。从技术上讲，每个结点用一个非递归的特征来标记的语法称为属性语法（attribute grammar）。

另一个办法是把词汇化的语法看成是带有很多规则的简单上下文无关语法；这就像我们对每个规则都做了很多个副本，一个副本表示一个成分中的一个中心词；Schabes et al.(1988)和Schabes(1990)根据这种思想建立了一个扩充的词汇化语法。一般来说，把这样的规则集中起来，它们的数量将会相当多，用这种办法建立的词汇化语法使得我们可以使用标准的CFG剖析算法来剖析它们。

现在来研究怎样使用概率来增强这些词汇化的语法，怎样做才能表示第9章中讨论过的词汇依存关系。假定我们是把概率词汇化语法当成一个正规而又庞大的PCFG来处理的。这样，将对每个“规则/中心语”的结合体存储一个概率，下面是几个稍微人为化的例子：

$$\begin{aligned}
VP(dumped) &\rightarrow VBD(dumped)\ NP(sacks)\ PP(into)\ [3\times10^{-10}] \\
VP(dumped) &\rightarrow VBD(dumped)\ NP(cats)\ PP(into)\ [8\times10^{-11}] \\
VP(dumped) &\rightarrow VBD(dumped)\ NP(hats)\ PP(into)\ [4\times10^{-10}] \\
VP(dumped) &\rightarrow VBD(dumped)\ NP(sacks)\ PP(above)\ [1\times10^{-12}] \\
&\cdots
\end{aligned} \tag{12.18}$$

显而易见，使用这个方法的问题是：我们还没有足够大的语料库来训练这样的概率。训练标准的PCFG概率时，对于几乎所有的规则都会引起零计数。为了避免零计数，需要做出某些简化的独立性假设，以便把某些计数进行聚类。

各种现代统计剖析器的主要不同点可能正是在于它们做出不同的独立性假设。在本节的其他部分，我们将介绍Charniak(1997)中的剖析器，但是也可以选择其他类似的基于依存关系的统计剖析器，例如Magerman(1995)，Collins(1999)和Ratnaparkhi(1997)。

正如其他基于依存的统计剖析器那样，Charniak的剖析器加入了词汇依存信息，把短语的中心语和它们的成分联系起来。Charniak的剖析器还加入了句法次范畴化信息，从而说明了在给定的规则中，非终极符号在扩充非终极符号的中心语时的概率条件。让我们来研究这两种统计策略的稍微简化了的例子。简化之后条件的因素比 Charniak 原来的完全算法中的因素少一些。

我们知道，在普通的 PCFG 中，被一个规则 r 展开的结点 n 的概率，其条件只有一个因素，就是结点 n 的句法范畴。为简单起见，我们用 n 来表示结点 n 的句法范畴。现在给它再加上一个条件因素：结点 n 的中心词 $h(n)$。这样就可以计算概率：

$$p(r(n)|n, h(n)) \tag{12.19}$$

例如，我们来研究在图 12.5 中规则 r 展开结点 *VP* 的概率，这个规则是：

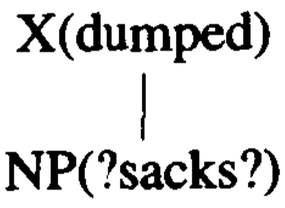
$$VP \rightarrow VBD\ NP\ PP$$

当回答“把以 *dumped* 为中心词的 *VP* 展开为 *VBD NP PP* 的概率是多少”这个问题时，我们可以说这个概率是 $p(r|VP, dumped)$。这样就能够抓住关于 *dumped* 的次范畴化信息；例如，一个以 *dumped* 为中心词的 *VP* 与一个以 *slept* 为中心词的 *VP* 相比较，以 *dumped* 为中心词的 *VP* 更乐意展开为 *NP* 加 *PP*。

我们已经把中心语作为一个条件因素，现在需要决定怎样来计算中心语的概率。如果把所有的中心语的概率都看成是相等的，也就是说，一个结点的中心语是 *sacks* 的概率与中心语是 *racks* 的概率是一样的，这样的假设就是无效的假设，没有多大用处。结点的句法范畴应该是有差别的（名词可能比动词具有种类差异更大的中心语），而且相邻的中心语也应该是有差别的。中心语为 h 的结点 n 的概率有两个条件因素：一个因素是结点 n 的句法范畴，一个因素是该结点的母结点的中心语 $h(m(n))$。其概率如下：

$$p(h(n) = word_i|n, h(m(n))) \tag{12.20}$$

例如，我们来研究图 12.5 中 *VP* 的第二个结点 *NP* 的中心语为 *sacks* 的概率。这个中心语的概率是 $p(head(n) = sacks|n = NP, h(m(n)) = dumped)$。这个概率可以回答问题：“中心语为 *sacks* 而且母结点为 *dumped* 的 *NP* 的概率是多少？”我们可以将其简单地表示如下：



这个图表明，这个中心语概率真正抓住了在单词 *dumped* 和单词 *sacks* 之间的依存关系信息。怎样使用这两个概率来计算整个剖析的概率呢？我们不再使用把PCFG规则的概率相乘的办法来计算剖析的概率，而是修改式（12.2），给每个规则加上它的中心语条件：

$$P(T,S) = \prod_{n \in T} p(r(n)|n, h(n)) \times p(h(n)|n, h(m(n))) \tag{12.21}$$

我们来观察歧义剖析的一个样本，以了解这些词汇化的概率在歧义消解中的用处。图 12.6 是句子 *Workers dumped sacks into a bin* 的另一个（不正确的）剖析，这个样本也来自 Collins(1999)。在这个不正确的剖析中，*into a bin* 这个 *PP* 是修饰 *sacks* 这个 *NP* 的，而不是修饰以 *dumped* 为中心语的 *VP* 的。这个剖析之所以不正确，是因为 *into a bin* 这个 *PP* 很难成为这个 *NP* 的修饰语，而它却更乐意像图 12.5 中那样来修饰动词 *dumped*。

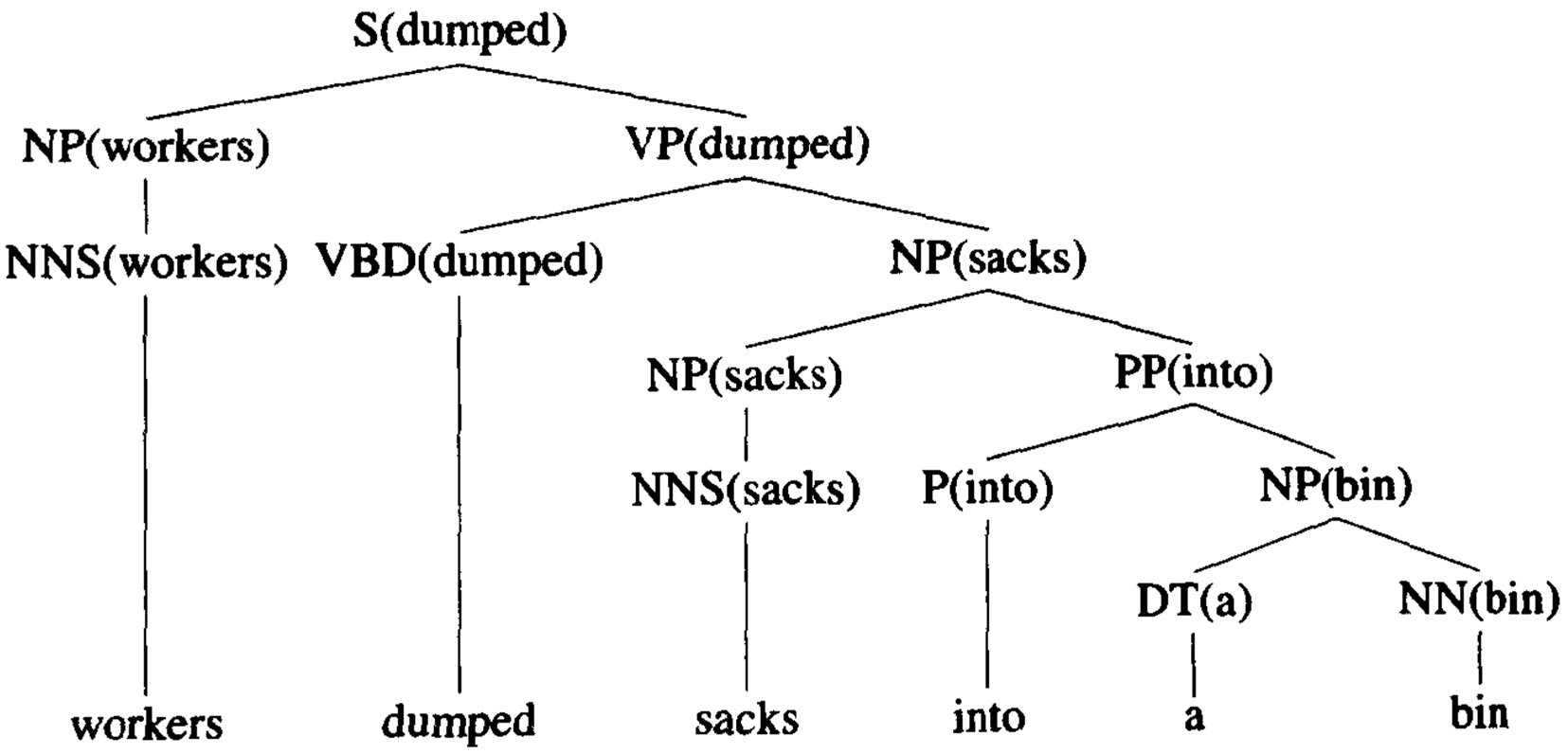


图 12.6　图 12.5 中的句子的一个不正确的剖析(Collins, 1999)

在式（12.21）中，中心语 - 中心语和中心语 - 规则的概率有助于正确地选择 *VP* 附着（见图 12.5），而不选择 *NP* 附着（见图 12.6）。这两个树的区别在于：在正确的树中 *VP*(*dumped*)展开为 *VBD NP PP*，而在不正确的树中 *VP*(*dumped*)展开为 *VBD NP*。让我们根据 Penn 树库中的 Brown 语料库中统计的计数来计算这两种剖析的概率。第一个规则的计算如下：

$$\begin{aligned} p(VP \rightarrow VBD\,NP\,PP | VP, dumped) &= \frac{C(VP(dumped) \rightarrow VBD\,NP\,PP)}{\sum_{\beta} C(VP(dumped) \rightarrow \beta)} \\ &= \frac{6}{9} = 0.67 \end{aligned} \tag{12.22}$$

第二个规则在 Brown 语料库中从未出现过。在实际工作中，需要对零值做某些平滑，但这里只需要知道第一个规则更优先。这样的结论并不奇怪，因为 dump 这个动词会引起移动到一个新位置：

$$\begin{aligned} p(VP \rightarrow VBD\,NP | VP, dumped) &= \frac{C(VP(dumped) \rightarrow VBD\,NP)}{\sum_{\beta} C(VP(dumped) \rightarrow \beta)} \\ &= \frac{0}{9} = 0 \end{aligned} \tag{12.23}$$

中心语的概率是多少呢？在正确的剖析中，*PP* 的母结点的中心语为 *dumped*，*PP* 的中心语是 *into*。在不正确的剖析中，*PP* 的母结点的中心语是 *sacks*，*PP* 的中心语是 *into*。这样，我们再使用 Penn 树库的 Brown 语料库中统计的计数进行计算：

$$\begin{aligned} p(into | PP, dumped) &= \frac{C(X(dumped) \rightarrow \ldots PP(into) \ldots)}{\sum_{\beta} C(X(dumped) \rightarrow \ldots PP \ldots)} \\ &= \frac{2}{9} = 0.22 \end{aligned} \tag{12.24}$$

$$\begin{aligned} p(into | PP, sacks) &= \frac{C(X(sacks) \rightarrow \ldots PP(into) \ldots)}{\sum_{\beta} C(X(sacks) \rightarrow \ldots PP \ldots)} \\ &= \frac{0}{0} = ? \end{aligned} \tag{12.25}$$

中心语的概率再次正确地预见了动词 *dumped* 比名词 *sacks* 更愿意被 *into* 修饰。

当然，仅仅一个例子还不能证明一个方法一定比其他方法好。另外，我们上面提到的概率词汇语法只是Charniak的实际算法的简化版本。他还增加了一些附加的条件因素（例如，某结点的祖父结点句法范畴的规则展开概率），并提出了各种回退和平滑算法，不过，现有语料库要获取这些统计数字还是显得太小了。另一些统计剖析器包括更多的因素，例如区分论元成分（argument）与附属成分（adjunct），对于树形图中那些比较接近的词汇依存关系比那些比较疏远的词汇依存关系给予更大的权重(Collins, 1999)，在给定成分中的三个最左的词类(Magermen and Marcus, 1991)以及一般的结构依存关系（例如，英语中右分支结构优先）(Briscoe and Carroll, 1993)。这些统计剖析器很多都进行过评测（根据同样的语料），评测时使用了下面所述的方法。作为练习，读者可以把CYK算法加以扩充，使它能处理词汇化的概率。

方法论探讨：剖析器的评测

评测剖析器和语法的标准技术称为PARSEVAL测度，PARSEVAL测度是Black et al.(1991)根据前面章节中介绍过的信号探测理论的思想提出的。在最简单的场合，测试集（例如Penn树库）中的一个特定剖析被定义为正确的剖析，把它作为测试集的“黄金标准”。对于给定的“黄金标准”，句子s的候选剖析c中的一个给定成分被标记为“正确的”，如果在树库中存在一个成分，与测试集中的这个成分具有相同的起点、终点和非终极符号，就可以对每个句子s测量其准确率（precision）、召回率（recall）和一个新的度量（交叉括号）：

$$\text{标记召回率：} = \frac{\#s\text{的候选剖析中的正确成分数}\#}{\#s\text{的树库中的正确成分数}\#}$$

$$\text{标记准确率：} = \frac{\#s\text{的候选剖析中的正确成分数}\#}{\#s\text{的候选剖析中的全部成分数}\#}$$

交叉括号：交叉括号的数量［例如，某一结构在树库中括号表示为((A B) C)，但在候选剖析中的括号表示却为(A (B C))，同一个结构的括号表示出现交叉，就算有一个交叉括号］。

使用一部分WSJ树库作为测试集，对于像Charniak(1997)和Collins(1999)这样的剖析器，其召回率在90%以下，准确率在90%以下，每个句子的交叉括号成分为1%。

为了比较使用不同语法的剖析器，PARSEVAL测度还包括一个规范化算法，以除去那些看起来在语法上太特殊的信息（如助动词、动词不定式前面的“to”等），并计算出一个简化的记分。有兴趣的读者可参阅Black et al.(1991)。此外，还有对于依存语法的相关评测度量方法(Collins et al., 1999)以及可用于任何剖析结构的基于依存的度量方法(Lin, 1995; Carroll et al., 1998)。

对于语法检查，我们可以不计算这种任务的准确率和召回率，而计算剖析器拒绝不合语法的句子（或识别合乎语法的句子）的频繁程度。

12.4　依存语法

前一节指出，基于成分的语法可以使用中心语之间的概率关系来增强，并且指出词汇依存信息在模拟中心语（如动词）对于它们的论元或修饰语的词汇约束时是非常重要的。

语法形式化的一个重要类别就是纯粹建立在词汇依存信息基础上的。在这些依存语法中，成分和短语结构规则没有任何根本性的作用。依存语法不用成分和短语结构规则，一个句子的句法结构完全由单词和这些单词之间的二元语义或句法关系来描写。依存语法主要来自Tesnière(1959)的工作，而“依存”这个词可以假定为首先由David Hays使用。不过，语法的词汇依存概念事实上比相对现代的短语结构或组成性语法早一些，这样的概念在古希腊和印度的语言学传统中就有了。传

统语法关于“把一个句子剖析为主语和谓语”的概念显然是建立在词汇关系的基础上，而不是建立在成分关系的基础上的。

图 12.7 是使用 Järvinen and Tapanainen(1997)和 Karlsson et al.(1995)的依存语法形式化方法来剖析句子 *I gave him my address* 的结果。注意，这里没有非终极结点或短语结点；在剖析树中的每个链接只在两个词汇结点之间存在（用了一个特殊的结点<ROOT>来增强这种表示方法）。这些链接是从包含35种关系的一个固定清单中提取的，这些关系中的大多数大致表示了语法功能或很普通的语义关系。其他基于依存的计算语法，例如链语法（link grammar）(Sleator and Temperley, 1993)，使用了不同的但大致交迭的链接。下面的表列出了 Järvinen and Tapanainen(1997)使用的一些依存关系：

依存关系	描述
subj	句法主语
obj	直接宾语（包括句子补语）
dat	间接宾语
pcomp	介词补语
comp	谓语体词（系词补语）
tmp	时间状语
loc	地点状语
attr	前修饰（定语）体词（属格等）
mod	体词后修饰语（介词短语等）

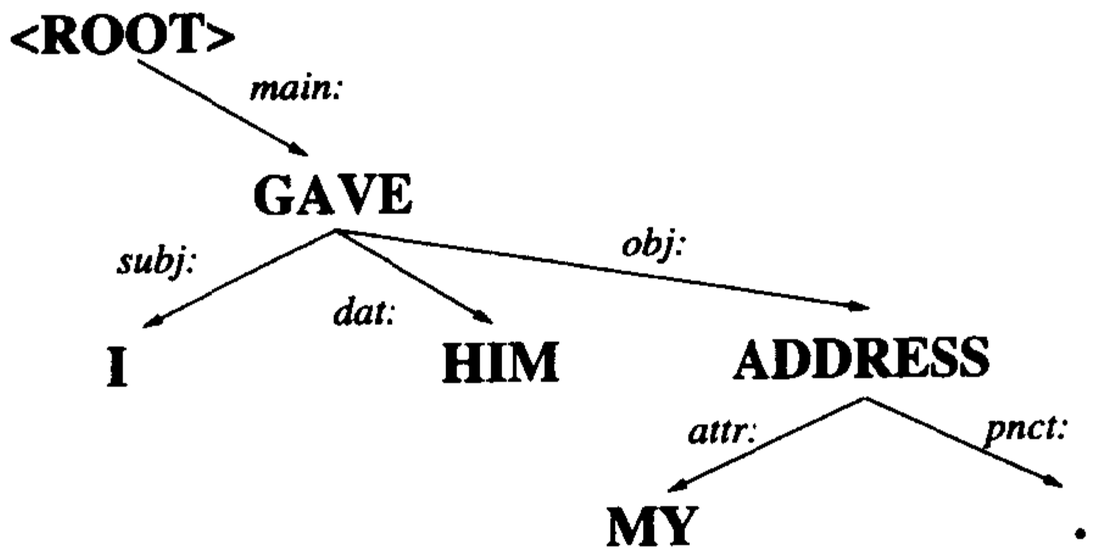


图 12.7 使用了Karlsson et al.(1995)的依存形式化方法的依存语法剖析的一个样本(Järvinen and Tapanainen, 1997)

我们已经讨论了为什么依存关系信息是很重要的。只使用依存信息而忽略组成性，是否还有其他任何好处呢？依存语法的研究者主张，纯粹的依存语法的一个最主要的好处是它们有能力来处理具有相对自由词序的语言。例如，像捷克语这样的语言的词序比英语灵活得多；一个宾语可以处于地点状语或谓语体词 comp 之前，也可以处于它们之后。当剖析树中出现这种状语短语时，对于每个可能的不同位置，短语结构语法都需要分别用不同的规则来处理。而对于这种特定的状语关系，依存语法只需要一个链接类型就可以代表了。可见，依存语法对不同的词序变化进行了抽象，只表示对于剖析必需的那些信息。

已经有一些在计算机上实现依存语法的研究成果；链语法（Link Grammar）(Sleator and Temperley, 1993)和约束语法（Constraint Grammar）(Karlsson et al., 1995)都是使用方便、覆盖面广的依存语法和英语的剖析器。依存语法也用于处理其他语言。例如，Hajic(1998)描写了含有 500 000 个单词的捷克语的 Prague 依存树库，这个树库被用于训练概率依存剖析器(Collins et al., 1999)。

12.4.1　范畴语法

范畴语法（category grammar）是首先由 Adjukiewicz(1935)提出来的，后来 Bar-Hillel(1953)，Lambek(1958)，Dowty(1979)，Ades and Steedman(1982)和 Steedman(1989)等都对范畴语法进行过修改。关于范畴语法的导论可以参阅 Bach(1988)，关于范畴语法基本模型的发展可以参阅 Oehrle et al.(1988)。这里只介绍 Steedman(1989)关于组合范畴语法（combinatory categorical grammar）的一个简化版本。范畴词表（category lexicon）把每个单词与一个句法语义范畴联系起来。组合规则（combinatory rule）把函数与主项（argument）结合起来。范畴有两种类型：一种类型是函子（functor），一种类型是主项。名词就是主项，它们只具有诸如 N 这样的简单范畴。动词和限定词是函子，它们可以被看成是函数。例如，限定词可以想像成一个函数，这个函数应用于它右侧的名词 N，形成一个 NP。这样的复杂范畴可以使用运算符 X/Y 和 $X\backslash Y$ 来建立。X/Y 是从 Y 到 X 的一个函数，其含义是：某种东西的右侧有一个 Y，它和这个 Y 结合之后形成一个 X。因此，限定词的范畴可以定为 NP/N，其含义是：限定词在其右侧与 N 相结合，形成一个 NP。及物动词的范畴为 VP/NP，其含义是：及物动词在其右侧与 NP 相结合，形成一个 VP。像 *give* 这样的双及物动词的范畴是 $(VP/NP)/NP$，其含义是：双及物动词在其右侧与一个 NP 相结合，形成一个及物动词 (VP/NP)。最简单的结合规则是：把一个 X/Y 在其右侧与 Y 结合起来，形成一个 X；把一个 $X\backslash Y$ 在其左侧与 Y 结合起来，形成一个 X。

我们来研究来自 Steedman(1989)的一个简单句子 *Harry eats apples*。我们不像过去那样使用 VP 这个原始范畴，而是假设定式动词短语 *eats apples* 的范畴是 $(S\backslash NP)$，其含义是：这个定式动词短语在其左侧与一个 NP 结合起来，形成一个句子 S。*Harry* 和 *apples* 两者都是 NP。*eats* 是一个定式及物动词，在其右侧与一个 NP 结合，形成一个定式动词短语 VP，故写为 $(S\backslash NP)/NP$。句子 S 的推导过程如下：

$$\dfrac{\dfrac{\underline{\text{Harry}}}{\text{NP}} \quad \dfrac{\underline{\text{eats}} \qquad \underline{\text{apples}}}{\dfrac{\text{(S\NP)/NP} \qquad \text{NP}}{\text{S\NP}}}}{\text{S}} \tag{12.26}$$

现代的范畴语法包含更复杂的组合规则，以满足处理联合结构和其他语言现象的需要。现代范畴语法还包括语义范畴以及句法范畴的合成。关于语义合成（semantic composition）的讨论，可以参阅第 15 章，关于范畴语法的更详尽的介绍，可以参阅上面提到的那些参考材料。

12.5　人的剖析

人是怎样来进行剖析的呢？我们有没有任何证据来说明，人也使用前面四章中讨论的语法模型和剖析方法吗？对于人的剖析的研究（通常称为人的句子剖析）是一个比较新的领域，因此我们还不能对这些问题做出完满的回答。但是，在过去的 20 年中，我们已经有了关于人的剖析的很多研究成果；在这一节中将对某些新理论进行简要介绍。这些成果相对而言还是比较新的，关于什么才是模拟人的剖析的正确途径，目前还有不同的看法，因此读者在阅读这部分内容时，请多加思考。

人的剖析的一个重要内容是歧义辨识。怎样才能知道，人们在一个句子存在两种歧义剖析时如何进行选择呢？在本章和第 9 章中已经指出，尽管在某种意义上说差不多每个句子都是有歧义的，但是人们很少会注意到这样的歧义，他们似乎只是从句子中看到了一个解释。Fodor(1978)和

Ford et al.(1982)使用下面的事实来说明人们在句子处理时对于词汇次范畴化的优先性（lexical subcategorization preference）是敏感的。他们给受试者显示如例句（12.27）和例句（12.28）中的歧义句子。在这些句子中，介词短语 *on the beach* 可能附着在名词短语 *the dogs* 上，也可能附着在动词短语上。在试验时，他们要求受试者读句子并检查一个小盒子，说明他们首先感到的是两个解释中的哪一个。受试者对于每个句子的检查结果如下所示：

The women kept the dogs on the beach （12.27）

- The women kept the dogs which were on the beach. 5%
- The women kept them (the dogs) on the beach. 95%

The women discussed the dogs on the beach （12.28）

- The women discussed the dogs which were on the beach. 90%
- The women discussed them (the dogs) while on the beach. 10%

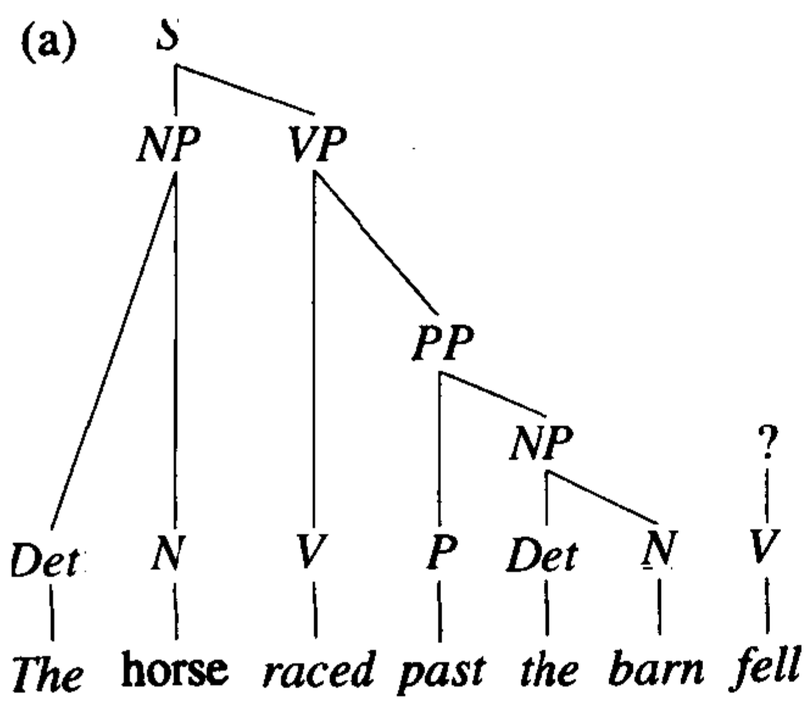
试验的结果是：对于动词 *keep*，受试者优先选择 *VP* 附着；对于动词 *discuss*，受试者优先选择 *NP* 附着。这意味着，*keep* 的次范畴化优先选择带有三个成分的 *VP*($VP \rightarrow V\ NP\ PP$)，而 *discuss* 的次范畴化优先选择带有两个成分的 *VP*($VP \rightarrow V\ NP$)，尽管这两种次范畴化都是这两个动词可以允许的。

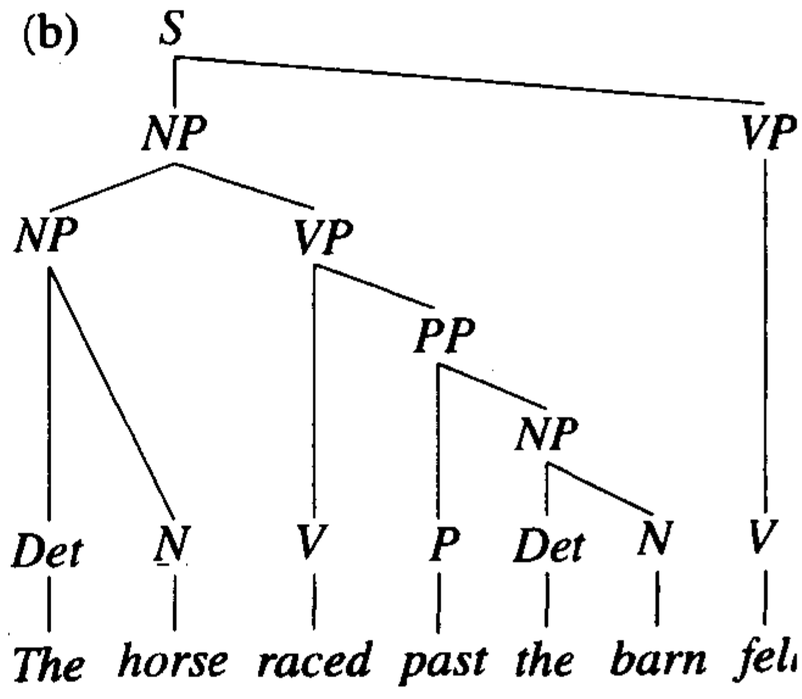
现在对于歧义辨识的很多研究都与一种称为花园幽径句（garden-path sentence）的特殊的临时歧义句有关。Bever(1970)首先描述了花园幽径句，这种句子的构造很巧妙，具有三个性质。这三个性质结合起来，使人们在剖析这种花园幽径句时感到非常困难。这三个性质是：

1. 花园幽径句是临时的歧义句，整个句子没有歧义，但是句子的前一部分是有歧义的。
2. 对于人的剖析机制来说，在句子的前一部分的两个或多个歧义剖析之间有优先性。
3. 但是，优先性低的剖析结果恰恰是这个句子的正确剖析结果。

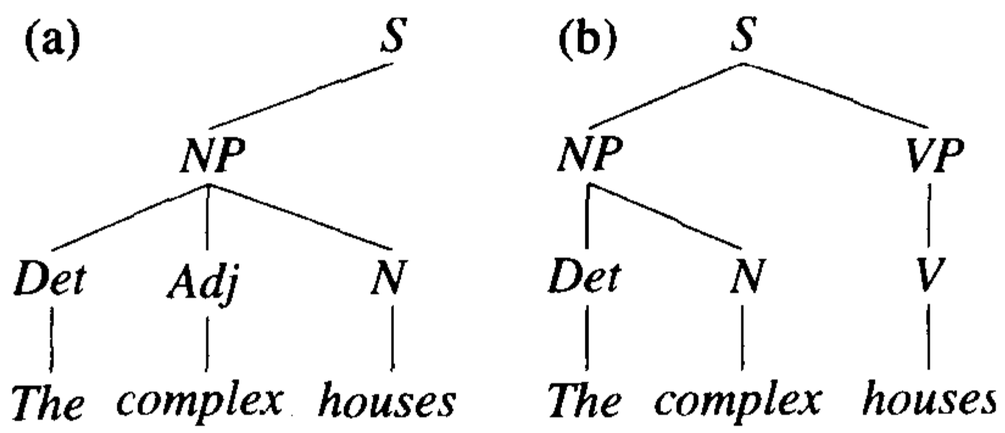
这三个性质综合作用的结果，使得人们在理解这样的句子时，一开始就像进入了花园幽径中，朝着错误的剖析走去。当他们发现这是错误的剖析时，会感到迷惑不解。有时，这样的迷惑是有意识的，例如 Bever 的例句（12.29）；事实上，这样的句子很难剖析，以至于通常都要给读句子的人显示正确的结构。在正确的结构中，raced 是简化的关系从句的一部分，修饰 *the horse*，意思是 *The horse [which was raced past the barn] fell*；这种类型的结构也出现在句子 *Students taught by the Berlitz method do worse when they get to France* 中。

The horse raced past the barn fell. （12.29）

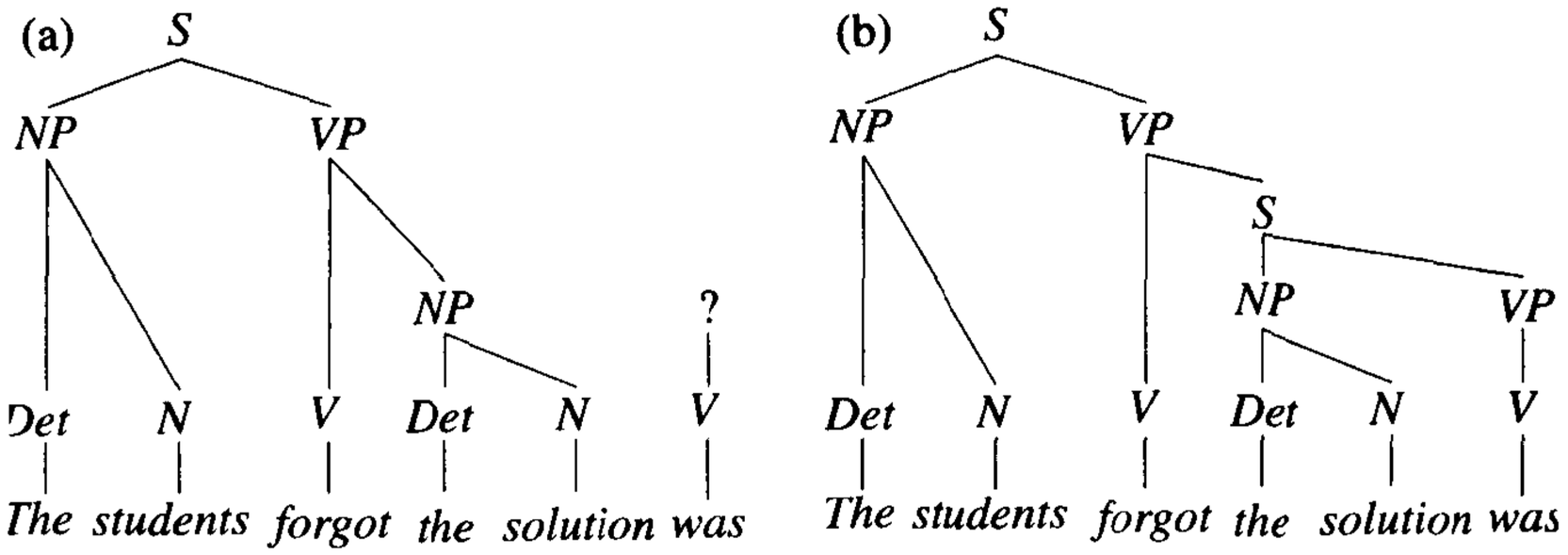




The complex houses married and single students and their families.　（12.30）



The student forgot the solution was in the back of the book.　（12.31）



花园幽径句造成的这种迷惑使人感到扑朔迷离，以至于在读这样的句子时不得不多花些时间才能正确理解它们。例句（12.31）是来自 Trueswell et al.(1993)的一个例子，这个例子在 Ferreira and Henderson(1991)的试验中有些修改，读这个句子的人往往会错误地把*the solution*剖析为*forgot*的直接宾语，而不把它剖析为嵌入句的主语。这是次范畴化优先性的另一个例子，这时*forgot*优先选择直接宾语($VP \rightarrow V\ NP$)，而不是优先选择句子补语($VP \rightarrow V\ S$)。不过，这样的差别还是令人捉摸不定，我们只能在受试者花很多时间来读单词*was*时，才能注意到这种扑朔迷离的情况。怎样了解到受试者究竟用了多长时间来读一个单词或短语呢？一个办法是把一个句子在计算机的屏幕上展开，在同一时间展开一个单词或短语；另一个办法是使用眼睛追踪器（eye-tracker）来追踪受试者的眼睛在每个单词上徘徊的时间。Trueswell et al.(1993)在不同的试验中使用这两种方法进行测试。这种“微型花园幽径”的效应对于单词*was*的影响说明，在剖析这个句子时受试者首先选择直接宾语的剖析，误入了花园幽径中。这样，受试者就不得不重新进行分析，重新安排他们原来的剖析，最后终于认识到应该把剖析引导到一个句子补语中，从而得到正确的剖析结果。与此相反，对于那些优先要求句子补语的动词（如*hope*），就不会像*was*这样要多余地花费阅读时间。

这样的花园幽径句不限于英语中。Gilboy and Sopena(1996)给出了西班牙语中的花园幽径句的例子，见例句（12.32）。在这个例子中，*que*（与英语的*that*类似）是有歧义的，它可以作为关系从句的标志，又可以作为句子补语的标志。所以，在读到短语*dos hijas*时，读的人认为这个句子的意思是“这个男子对这个女子说，他有两个女儿”；当读到第二个*que*的时候，读的人必须重新剖析*que tenia dos hijas*，把它剖析为名词*mujer*的修饰语，而不是作为动词*dijo*的补语。为了便于理解，在西班牙语单词的下面已经注出了相应的英语单词。

El	*hombre*	*le*	*dijo*	*a*	*la*	*mujer*	*que*	*tenía*	*dos*	*hijas*
The	man	her	told	to	the	woman	that	had	two	daughters
que	*la*	*invitaba*	*a*	*cenar.*						
that	her	he invited	to	dinner.						

（这个男子对这个有两个女儿的女子说，他想邀请她吃晚饭。）　（12.32）

例句（12.33）是 Mazuka and Itoh(1995)中提出的日语中的花园幽径句。在这个句子中，一直到读到*mikaketa*（看）时，读的人都认为这个句子的意思是：Yoko看到那个在十字路口的小孩，但是当他们读到 *takusii-ni*（*taxi-DAT*，出租汽车充当给予格）时，则要重新分析，不是把 *kotomo-o*（小孩）分析为 *mikaketa*（看）的直接宾语，而是把它分析为 *noseta*（放到）的直接宾语，而 *takusii-ni* 则分析为 *noseta* 的间接宾语。为了便于理解，在日语单词的下面已经注出了相应的英语单词：

Yoko-ga	*kodomo-o*	*koosaten-de*	*mikaketa*	*takusii-ni*	*noseta.*
Yoko-NOM	child-ACC	intersection-LOC	saw	taxi-DAT	put on

（12.33）

（Yoko把这个小孩放到她在十字路口看到的出租汽车上。）

在西班牙语和日语以及例句（12.29）和例句（12.31）中，花园幽径是由动词的次范畴化的优先性引起的。花园幽径和其他一些方法还用来研究除了次范畴化的优先性之外的很多其他类型的优先关系。Jurafsky(1996)给出的例句（12.30）表明，有时这些优先关系是词类的优先关系（例如，*houses* 在这里看成一个动词还是名词）。很多这样的优先关系是有概率的，它们与本章描述的概率的类别是有关系的。MacDonald(1993)指出，人的语言处理对于一个名词是作为成分的中心语还是非中心语是很敏感的，对于单词与单词之间的搭配概率也是很敏感的。Mitchell et al.(1995)指出，句法的短语结构频度（诸如关系从句的频度）在人的语言处理中起着一定的作用。Juliano and Tanenhaus(1993)指出，人的语言处理对于词汇与短语结构的频度的结合也是很敏感的。

除了语法知识以外，人的语言处理还受到许多其他因素的影响，我们将在以后描述这些因素，包括资源的约束（例如记忆的局限性，详见第13章）、主题结构（例如一个动词是不是要求语义施事或受事，详见第16章）、语义和话语以及其他上下文约束（详见第15章和第17章）。人的句子剖析所用的知识源一般是一致的，但这些知识源使用的时间进程（time course）却缺乏一致性。Frazier和他的同事们指出，在人的剖析中，开始解释时使用纯句法知识，然后才使用语义知识、主题知识和话语知识。这种观点常常被称为“模块论”（modularist perspective），最新的研究可参阅 Frazier and Clifton(1996)。持这种立场的研究人员一般认为，在人的头脑中，句法知识是一个独特的模块。其他很多文献，包括(MacDonald, 1994; MacWhinney, 1987; Pearlmutter and MacDonald, 1992; Tabor et al., 1997; Trueswell and Tanenhaus, 1994; Trueswell et al., 1994)，持“互动论”（interactionist）的观点。互动论的研究人员认为，在人的语言处理中，人们递增地使用多种信息。对于互动论者来说，人的语言剖析是一个互动过程，在这个过程中，不同的知识源交互地对解释过程进行限制。

包括MacDonald(1993)和Jurafsky(1996)在内的一些文献中指出，无论语言知识的使用是否具有时间进程的差异，所有这些约束从根本上讲都是概率性的。例如，Jurafsky(1996)的贝叶斯句法处理模型显然是概率性的，Narayanan and Jurafsky(1998)使用概率来解释上面的例句（12.29）至例句（12.31）以及类似的花园幽径句中的难点，也是概率性的。在他们的模型中，人的语言处理取一个歧义的输入句子，计算出若干个不同的并行解释。对于每个这样的解释指派一个概率，使用贝叶斯信念网络（belief network），把 PCFG 概率、句法概率、主题概率和次范畴化概率以及其他上下文概率结合起来。例如，当人在处理例句（12.29）时，使用图 12.8 中的信念网络，在看了前面的少数几个单词（*The horse raced*）之后，贝叶斯模型给例句（12.29a）和例句（12.29b）中的树指派主要动词（Main Verb，简称MV）解释以及简化关系从句（Reduced-Relative，简称RR）解释的概率。这样要把多个概率事实的资源结合起来，例如动词 *raced* 是及物动词的概率（即 *raced* 带直接宾语），*horse* 是奔跑事件（racing event）的主题 *Theme* 的概率，名词短语包含一个简化关系从句的句法概率。由于 *race* 更倾向于作为及物动词（带一个直接宾语）而不倾向于作为不及物动词，又

由于简化关系从句本身的概率很低，如例句（12.29a）和例句（12.29b）所示，模型预言 *raced* 为主要动词 MV 解释的概率是为简化关系从句 RR 解释的概率的 300 倍。

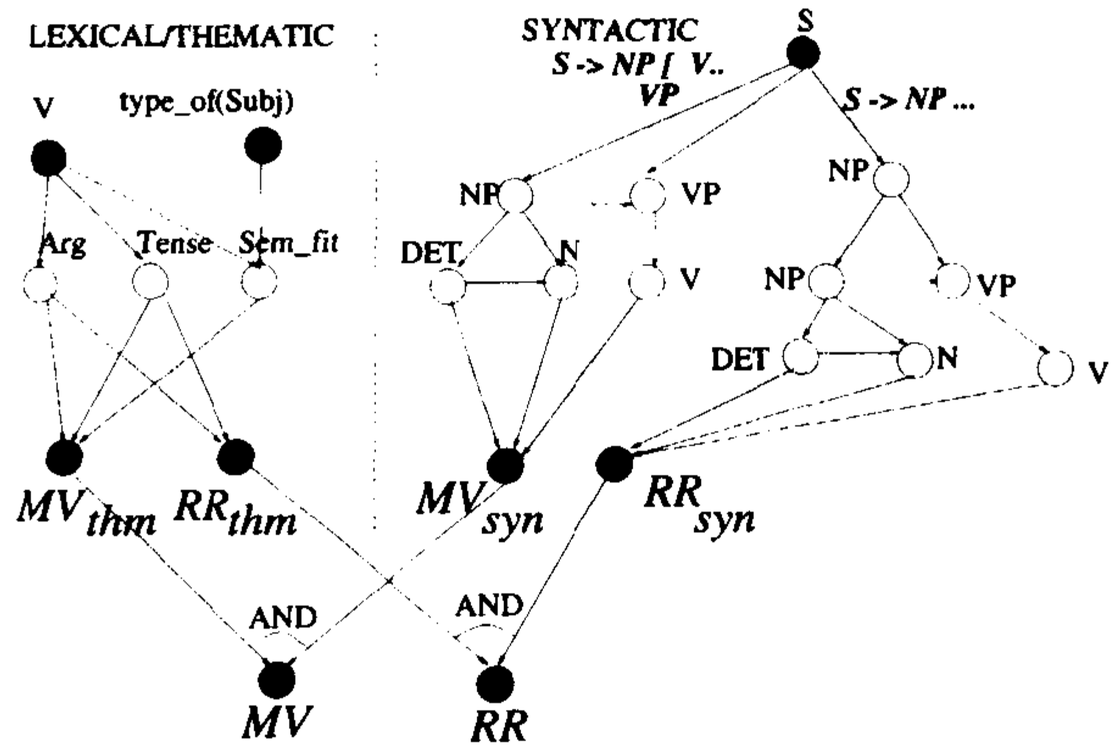


图 12.8　把PCFG和次范畴化概率、主题概率以及其他词汇概率结合起来的信念网络，表示对于输入的样本句子究竟为主要动词（MV）解释还是为简化关系从句（RR）解释的支持程度

这个模型进一步假定人们不能在同一时间内保持很多的解释。这或者是由于人们的记忆有限，或者是由于他们强烈地希望选择一个简单的解释，这样就把那些级别低的解释修剪掉了（关于“修剪”，见第 7 章）。Jurafsky(1996)和 Narayanan and Jurafsky(1998)假定，如果某一个概率比最高的概率解释低 5 倍，那么这样的解释就要被修剪掉。其结果使得人们有时会把那些正确的解释修剪掉，而保持了那些得分很高但实际上不正确的解释。正是因为这个原因，在例句（12.29）中，当人剖析前一段时，概率低（但“正确”）的简化关系从句 RR 解释就被修剪掉了。

这个模型也可以解释为什么一些表面上看起来相似的句子不会造成花园幽径句。例如，在例句（12.34）中(Pritchett, 1988; Gibson, 1991)，由于动词 *found* 是一个及物动词，所以 *found* 的简化关系从句的解释的概率大大高于 *race* 的简化关系从句的概率。

The bird found in the room died.　　（12.34）

图 12.9 说明了 *The horse raced* 和 *The horse found* 在 MV/RR 概率比值上的差别。

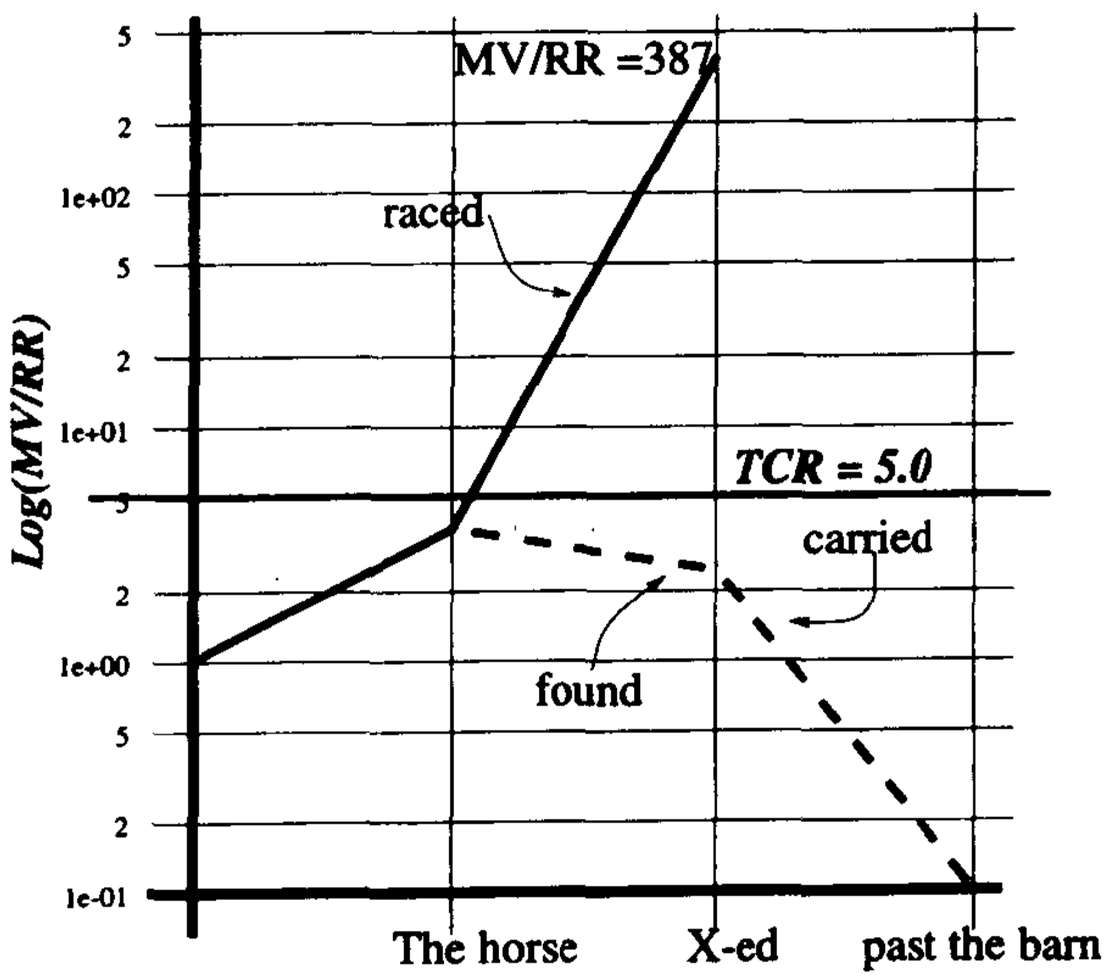


图 12.9　*raced* 的后概率比值 MV/RR 处于阈值之上，因此解释 *RR* 被修剪。*found* 和 *carried* 的 MV/RR 都处于歧义消解区域内(Narayanan and Jurafsky, 1998)

使用同样的概率区分方法，这个模型也可以解释例句（12.30）中的花园幽径效应；例如，$P(N \rightarrow houses)$大于$P(V \rightarrow houses)$，等等。Narayanan 和 Jurafsky 的模型还是很初步的，这个模型主要根据相对简单的独立性假设和很粗糙的概率模型；当前研究的关键焦点是确切地理解当人处理句子时究竟存储了什么概率？这些概率是怎样动态结合起来的？

12.6 小结

在本章中，我们给概率剖析的基础画出了一个轮廓，介绍了概率上下文无关语法和概率词汇化语法。

- 概率语法给一个句子或者单词的符号串指派了一个概率，从而捕捉到了比第6章中的N元语法更细致的句法信息。
- 概率上下文无关语法（PCFG）是一种上下文无关语法，其中每个规则都标上了选择该规则的概率。处理每个上下文无关规则时，假定它们在条件上是独立的；因此一个句子的概率使用剖析该句子时每个规则的概率的乘积来计算。
- Cocke-Younger-Kasami算法（CYK算法）是一种自底向上的动态规划剖析算法。不论是CYK算法还是 Earley 算法，经过增强之后都可以计算它们在剖析一个句子时的剖析概率。
- PCFG的概率可以通过一个已剖析好的语料库的计数得到，或者通过直接剖析一个语料库得到。当剖析的句子有歧义时，可以使用向内 - 向外算法来处理。
- 概率词汇化的CFG使用每个规则的词汇中心语来增强。这样，规则的概率就要以词汇中心语和邻近的中心语作为它的条件。
- 剖析器可以使用三个办法来评价：标记的召回率、标记的准确率、交叉括号。
- 花园幽径句和其他联机句子处理试验证明，人的剖析是按照概率进行的，并且使用诸如次范畴化信息这样的概率语法知识。

12.7 文献和历史说明

概率上下文无关语法的很多形式特性是首先由Booth(1969)和Salomaa(1969)揭示出来的。Baker(1979)提出了向内 - 向外算法来无指导地训练PCFG概率，使用了一个具有CYK风格的剖析算法来计算向内概率。Jelinek and Lafferty(1991)扩充了CYK算法，用这种算法来计算前缀的概率。Stolcke(1995)改进了这些算法，使 Earley 算法能够用于 PCFG。

在20世纪90年代初期，很多研究人员开始探索给PCFG增加词汇依存关系的问题，以便使PCFG的概率对于周围的句法结构具有更大的敏感性。这方面的很多论文发表在 1990 年 6 月召开的 DARPA 语音和自然语言讨论会上。Hindle and Rooth(1990)发表的一篇文章应用词汇依存关系来解决介词短语附着问题。在对这篇文章提问的会议上，Ken Church建议把这样的方法全面应用到整个剖析中(Marcus, 1990)。关于使用概率依存信息来增强概率 CFG 剖析的早期工作可参阅 Magerman and Marcus(1991)，Black et al.(1992)，Jones and Eisner(1992)，Bod(1993)和 Jelinek et al.(1994)。此外，Collins(1996)，Charniak(1997)和 Collins et al.(1999)也讨论了上述问题。

除了 PCFG 之外，使用概率方法的形式化算法还有：基于第 9 章中介绍过的 TAG 语法的概率 TAG 语法(Resnik, 1992; Schabes, 1992)，概率 LR 剖析(Briscoe and Carroll, 1993)以及概率链语法(Lafferty et al., 1992)。一种称为超级标注（supertagging）的概率剖析方法把词类标注的比喻扩展到使用非常复杂标记的剖析，这样的标记实际上是基于Schabes et al.(1988)的词汇化TAG语法的词汇

化剖析树的片断(Bangalore and Joshi, 1999; Joshi and Srinivas, 1994)。例如，*purchase*（购买）这个名词作为名词组合中的第一个名词（这里，它可能处于由一个名词性成分支配的小树的左侧）的标记与作为名词组合中的第二个名词（这里，它可能处于小树的右侧）的标记是不同的。关于基于特征的语法的概率处理，可参阅 Goodman(1997)和 Abney(1997)。另外还有方法把第 9 章中介绍的有限状态剖析模型和 N 元语法结合起来，首先进行局部剖析，然后对于基本短语，例如 $P(PP \mid NP)$，计算N元语法(Moore et al., 1995; Zechner and Waibel, 1998)。很多概率剖析器都是基于依存语法的；作为例子，可以参阅 Chelba et al.(1997)，Chelba and Jelinek(1998)和 Berger and Printz(1998)；这些剖析器也可以作为语言模型来进行语音识别。

与概率依存语法有关的是关于动词次范畴化框架以及这些框架的概率的学习。学习非概率的动词次范畴化框架的算法可参阅Brent(1993)基于提示的方法（cue-based approach）以及Manning(1993)关于有限状态自动机的方法。Briscoe and Carroll(1997)抽取了更复杂的次范畴化框架（这个框架使用了160个可能的次范畴化标记），并使用了一个概率LR剖析器和后处理方法，学习了次范畴化框架的频度。Roland and Jurafsky(1998)指出，计算单词的词目（lemma）的次范畴化概率比计算符合正词法的简单单词的次范畴化概率更重要。

自从Hindle和Rooth的研究以来，很多学者都在研究介词短语的附着问题，主要的工作有：基于转换的学习（TBL）方法(Brill and Resnik, 1994)，最大熵方法(Ratnaparkhi et al., 1994)，基于记忆的学习方法(Zavrel and Daelemans, 1997)，对数线性模型方法(Franz, 1997)，使用中心语之间的语义距离的决策树方法（根据 WordNet 进行计算）(Stetina and Nagao, 1997)以及使用诸如递进自举（boosting）这样的机器学习技术(Abney et al., 1999)。

Manning and Schütze(1999)是关于统计自然语言处理的一本很好的教材，讲述了概率剖析问题。Collins的博士论文Collins(1999)对于这个领域的工作进行了概述，可读性很强，同时也介绍了他设计的剖析器。

第13章　语言的复杂性

This is the dog, that worried the cat, that killed the rat, that ate the malt, that lay in the house that Jack built.

Mother Goose, *The House that Jack Built* ①

This is the malt that the rat that the cat that the dog worried killed ate.

Victor H. Yngve (1960) ②

很多音乐喜剧和滑稽小歌剧的幽默故事都是根据那些寓言般的复杂而曲折的情节编写而成的。Casilda是威尼斯水城贡都拉的Gilbert和Sullivan地区Plaza-Toro公爵的女儿，她爱上了她父亲的仆人Luiz。遗憾的是，Casilda发现，根据一份委托书，当她还是6个月大的婴儿时，她已经被许配给Barataria国王陛下的当时也还是婴孩的儿子（即继承人），这个国王的财富难以计数。可是后来这个国王的儿子遭到大法官的诱拐，由一个威尼斯的备受尊敬的贡都拉船夫抚养。这个贡都拉船夫有一个儿子，其年龄与国王儿子相同。他从来都记不清这两个儿子谁是谁。这样，当Casilda认识到她已经与贡都拉船夫的两个儿子中的一个结了婚，但又不知道究竟是其中的哪一个时，便坠入了一个极为难堪的处境中。大法官为了安慰可怜的Casilda，只好对她说："这种复杂情况是经常发生的"。

幸运的是，这样的复杂情况在自然语言中并不经常发生。不过，它们或许也是经常发生的。事实上，在自然语言中有些句子是非常复杂的，以至于理解起来非常困难，上面举出的Yngve的那个句子就是这样，下面的句子也是非常复杂的：

The Republicans who the senator who she voted for chastised were trying to cut all benefits for veterans.
（她投票支持的议员严厉谴责过的共和党人试图取消退伍军人的福利）

对于这些复杂性很高的句子进行研究，并且更广泛地了解在自然语言中发生的这种复杂性的程度，是自然语言处理的一个重要领域。这种复杂性起着重要的作用。例如，当我们需要使用某种特定的形式化方法时，必须知道这种形式化方法的复杂性。自然语言中的形式化方法，诸如有限状态自动机、马尔可夫模型、转录机、音位重写规则、上下文无关语法等，都可以用它们的能力（power）来描述，这种能力也就等价于它们所描写的现象的复杂性。本章将介绍Chomsky层级（Chomsky hierarchy），这是一种理论工具，使得我们可以比较这些形式化方法的表达能力或复杂性。我们手里有了这样的工具，就有可能对于自然语言句法的正确形式能力进行总结。具体地说，我们除了研究英语之外，还要研究德语的一种著名的瑞士方言，这种方言具有一种称为"交叉连续依存关系"（cross-serial dependency）的非常有趣的句法性质。这种性质可以用来证明，上下文无关语法没有足够的能力来模拟自然语言的形态和句法。

---

① 英语原文是个多层叠套的句子，翻译成蹩脚的汉语译文后已经失去了原来的结构。译文供参考：
这是为咬死偷吃放在Jack建造的房子里的麦芽的老鼠的那只猫感到焦躁不安的那条狗。

② 这是另一种结构的叠套句子，翻译成的蹩脚汉语译文后也失去了原来的结构。译文供参考：
这是为咬死偷吃麦芽的老鼠的那只猫感到焦躁不安的那条狗。

除了使用复杂性作为理解自然语言和形式模型之间的关系的度量之外，复杂性的领域还要研究究竟是什么使得一个单独的结构或句子变得难以理解。例如，上面举出的嵌套句（nested sentence）或中心嵌入句（centerembedded sentence），处理起来就很困难。了解究竟是什么使得某些句子处理起来特别困难，这是理解人的剖析的一个重要部分。

13.1　Chomsky 层级

自动机、上下文无关语法以及音位重写规则之间究竟有什么联系呢？它们之间的共同之处在于都描写了一种形式语言（formal language），而形式语言是在有限字母表上的符号串集合。但是，我们用每个形式化方法写的语法的类别具有不同的生成能力（generative power）。如果一个语法能够定义一种语言，而另一种语法不能，我们就说这种语法比另一种语法具有更强的生成能力或更大的复杂性。例如，我们将说明，上下文无关语法能够描述的形式语言是有限状态自动机不能描述的。

我们可以建立语法的层级，其中具有较强语法描述能力的语言的集合蕴涵具有较弱语法描述能力的语言。可能的语法层级有很多，在计算语言学中最常用的是 Chomsky 层级(Chomsky, 1959a)。Chomsky 层级包含四种语法，如图 13.1 所示。

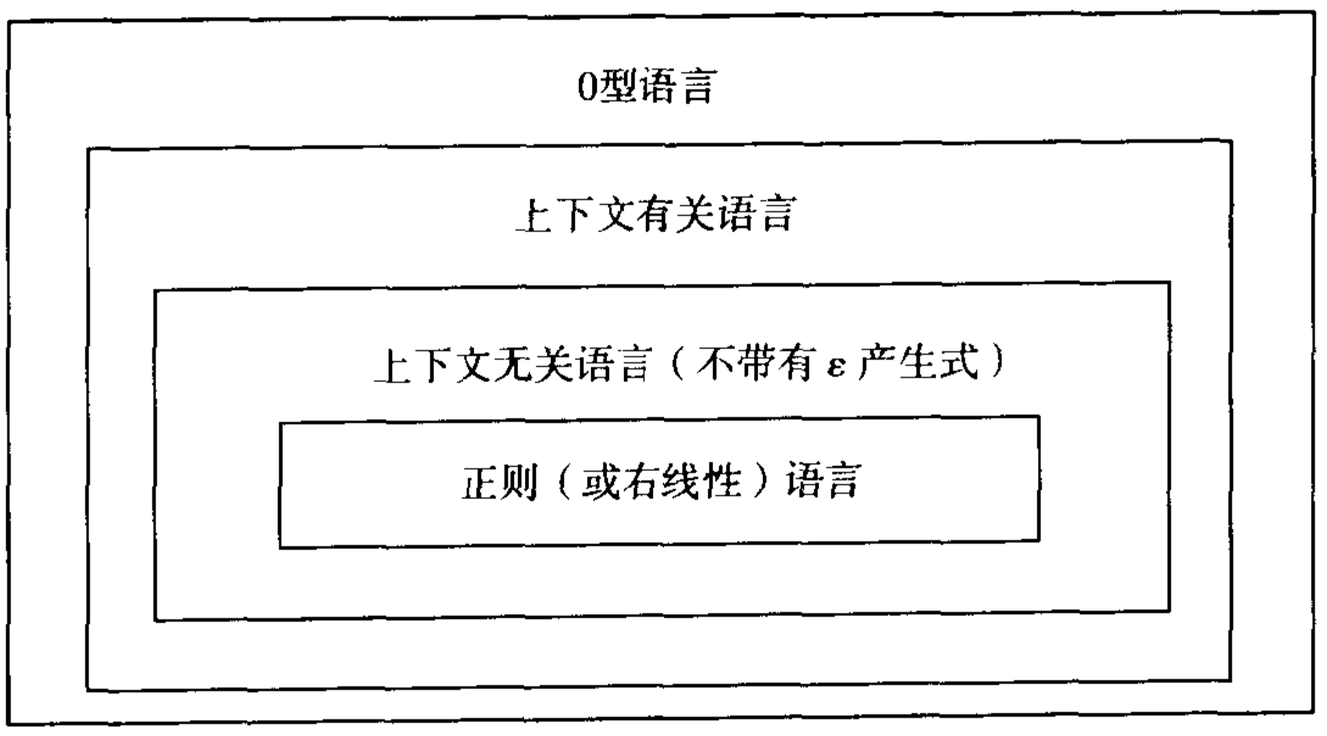


图 13.1　Chomsky 层级的语言文氏图（Venn diagram）

在直觉上不太明显的是：伴随着加在可重写语法规则上的约束的增加，语言的生成能力从最强降到最弱。下面的图 13.2 说明了 Chomsky 层级中的语法的四种类型，这些语法是通过规则形式的约束来定义的。在例子中，A 是一个简单的非终极符号，α, β 和 γ 是由终极符号和非终极符号构成的任意符号串。除了特别提出不允许之外，它们可以为空，x 是任意的终极符号串。

类型	通用名称	规则基干	语言学例子
0	Turing等价	$\alpha \rightarrow \beta$, s.t. $\alpha \neq \varepsilon$	扩充转移网络
1	上下文有关	$\alpha A \beta \rightarrow \alpha \gamma \beta$, s.t. $\gamma \neq \varepsilon$	树邻接语法
2	上下文无关	$A \rightarrow \gamma$	短语结构语法
3	正则	$A \rightarrow xB$ 或 $A \rightarrow x$	有限状态自动机

图 13.2　Chomsky 层级

0型语法或无限制语法除了要求规则的左部不能是空符号串 ε 之外，对于它们的规则的形式没有限制。任何非零的符号串都可以重写为任何其他符号串（或者 ε）。0 型语法刻画了递归可枚举语言（recursively enumerable language），也就是说，0 型语法生成的符号串可以由 Turing 机（Turing machine）列出（或枚举）。

上下文有关语法的规则可以把上下文 $\alpha A\beta$ 中的非终极符号 A 重写为任意非空符号串。规则可以写为 $\alpha A\beta \rightarrow \alpha\gamma\beta$ 的形式，或者 $A \rightarrow \gamma / \alpha___\beta$ 的形式。后一种形式在 Chomsky-Halle 的音位规则表达式中曾经介绍过(Chomsky and Halle, 1968)，颤音化（flapping）的规则表示如下：

$$/t/ \rightarrow [dx] / \acute{V} ___ V$$

这些规则的形式看起来都像上下文有关的。在第4章中说过，没有递归的音位规则系统的能力实际等价于正则语法。上下文有关的语言模型是树邻接语法(Joshi, 1985)。

理解上下文有关语法中的规则的另一个办法是：把这种规则想像成以“非递减”的方式把符号串 δ 重写为符号串 ϕ，使得 ϕ 中的符号至少与 δ 中的符号一样多。

在第9章中，我们曾经讨论过上下文无关语法。上下文无关规则可以把任何一个单独的非终极符号重写为由终极符号和非终极符号构成的符号串。这个单独的非终极符号也可以重写为 ε，尽管在第9章中没有使用过这种形式的规则。

正则语法与正则表达式等价。也就是说，一个给定的正则语言可以用第2章中介绍过的正则表达式来刻画，也可以用正则语法来刻画。正则语法可以是右线性的（right-linear），也可以是左线性的（left-linear）。右线性语法规则的右边只有一个单独的非终极符号，左边最多只有一个非终极符号。如果在右边只有一个非终极符号，它必定是符号串中的最后一个符号。左线性语法的右边是可逆的（右边必须至少以一个单独的非终极符号开始）。所有的正则语言既有左线性语法，也有右线性语法。在下面的讨论中，我们只考虑右线性语法。

例如，我们来研究下面的正则（右线性）语法：

$$S \rightarrow aA$$
$$S \rightarrow bB$$
$$A \rightarrow aS$$
$$B \rightarrow bbS$$
$$S \rightarrow \varepsilon$$

这是一个正则语法，因为每个规则的左边是一个单独的非终极符号，每个规则的右边至多只有一个（最右边的）非终极符号。下面是这种语言中一个推导的样本：

$$S \Rightarrow aA \Rightarrow aaS \Rightarrow aabB \Rightarrow aabbbS \Rightarrow aabbbaA$$
$$\Rightarrow aabbbaaS \Rightarrow aabbbaa$$

我们可以看出，每次展开 S 时，它或者产生 aaS，或者产生 $bbbS$，因此读者可以相信，这个语言对应于正则表达式 $(aa \cup bbb)^*$。

我们在这里没有证明：一个语言是正则的，当且仅当它是由一个正则语法生成的。这样的证明是首先由 Chomsky and Miller(1958)给出的，可以在 Hopcroft and Ullman(1979)和 Lewis and Papadimitriou(1981)中找到。我们从直觉上可以感到，由于非终极符号总是处于一个规则的最右边或者最左边，它们可以迭代地进行处理，而不可以递归地进行处理。

13.2 怎么判断一种语言不是正则的

怎样才能知道，对于给定的问题可以使用什么类型的规则？我们可以使用正则表达式来为英语写一个语法吗？真的需要使用上下文无关规则甚至使用上下文有关规则来写英语语法吗？已经证明，对于形式语言来说，存在着进行这样的判定的方法。也就是说，对于一个给定的形式语言，可以说它能否使用正则表达式来表达，或者能否要求上下文无关语法来表达，等等。

因此，如果想知道自然语言的某一部分（例如，英语的音位或土耳其语的形态）是不是由某种语法来表示，需要找到一种形式语言来模拟相关的现象，并且说明哪种语法适合于这种形式语言。

为什么要关心英语的句法能否由正则语言来表示呢？一个主要的原因是我们希望知道，什么类型的规则可以用来作为英语的计算语法。如果英语是正则的，就要写正则表达式，并且使用有效的自动机来处理这些规则；如果英语是上下文无关的，就要写上下文无关规则，并且使用Earley算法来剖析句子，等等。

我们要关心这个问题的另一个原因是，它可以告诉我们自然语言的不同方面的某些形式特性；如果我们知道语言在哪些地方“保持”着它的复杂性，一种语言的音位系统是不是比句法系统更简单，某些类型的形态系统是不是比另一些类型的形态系统更简单，这将是非常好的事情。例如，如果我们能够说明，英语的音系学可以使用有限状态自动机来描述，而不是由传统的上下文有关规则来描述，这就意味着英语的音系学具有非常简单的形式特性。当然，这些情况Johnson(1972)都做过说明，而且可以帮助我们了解第3章和第4章中关于有限状态方法的最新研究成果。

13.2.1　抽吸引理

证明一种语言是正则语言的最普通的方法是为这种语言实际地建立起正则表达式。我们可以根据正则语言对于并运算、毗连运算、Kleene星号运算、补运算、交运算等都是封闭的这种特性来进行这样的证明。在第2章中曾经给出过并运算、毗连运算、Kleene星号运算的一些例子。因此，如果能够为一种语言中的两个不同部分独立地建立起正则表达式，就可以使用并运算的运算子为整个语言建立正则表达式，并且证明这种语言是正则语言。

有时，我们试图证明一种语言不是正则语言。证明这种情况的最有用的工具是抽吸引理（pumping lemma）。这个引理的后面有两种直觉，对于抽吸引理的描述来自Lewis and Papadimitriou (1981)以及Hopcroft and Ullman(1979)。第一个直觉是，如果一种语言能够被有限状态自动机模拟，那么我们必定能够根据这种记忆约束量来判定任何符号串是不是在该语言中。这个记忆约束量对于不同的符号串不会增长得很大（因为对于给定的自动机，它的状态数目是固定的），因此这个记忆量不一定与输入的长度成比例。这意味着，诸如a^nb^n这样的语言不可能是正则语言，因为这时我们需要某种办法来记住n的数目，以便保证符号串中a的数量与b的数量相等。第二个直觉依赖于这样的事实：如果一个正则语言具有任意长的符号串（比自动机中的状态数还长），那么在该语言的自动机中必定会存在某种回路（loop）。我们可以使用这样的事实说明，如果一种语言没有这样的回路，它就不是正则语言。

让我们来研究一种语言L和与它相应的确定有限自动机M，M具有N个状态。考虑输入符号串的长度也是N。这个机器从状态q_0开始，在读了一个符号之后进入状态q_1，在读了N个符号之后进入状态q_n。换言之，长度为N的符号串将通过$N+1$个状态（从状态q_0到状态q_N）。但是，在机器中只有N个状态。这意味着，在接收的路径上至少有两个状态必须是相同的（把它们称为q_i和q_j）。换句话说，在从开始状态到最后状态的接收路径上，必定存在回路。图13.3说明了这种情况。设x是机器从开始状态q_0到回路起点的状态q_i读的符号串，y是机器通过回路时读的符号串，z是从回路终点q_j到最后的接收状态q_N读的符号串。

机器接收由这三个符号x, y和z构成的毗连。但是，如果机器接收了xyz，那么它一定也接收xz!这是因为机器在处理xz时可以跳过回路，y就像被抽水机抽吸了一样。另外，机器也可以在回路

上打任意次数的圈儿，这样它也就可以接收 *xyyz*，*xyyyz* 和 *xyyyyz* 等；这时，*y* 又被放出来了。事实上，当 $n \geq 0$ 时，它可以接收形式为 xy^nz 的任何符号串。

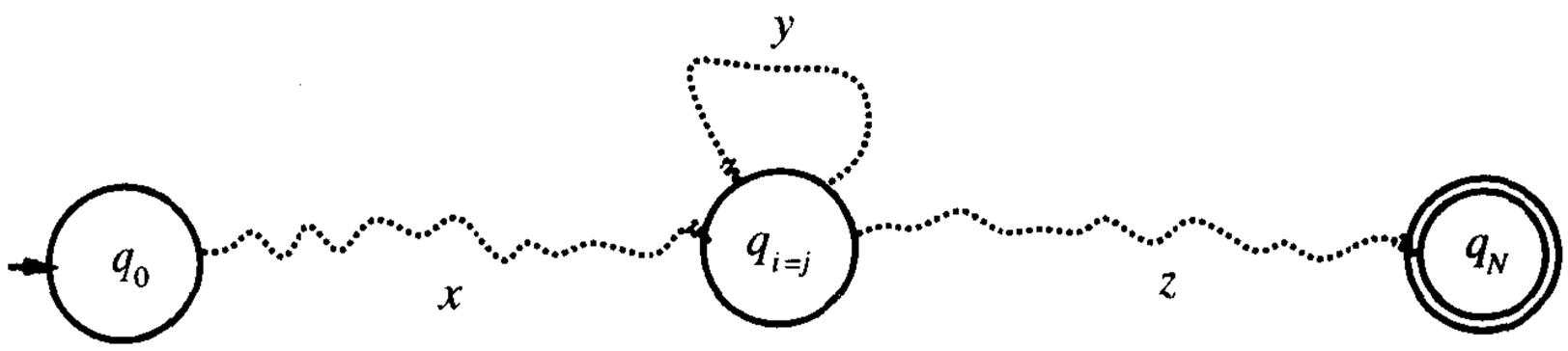


图 13.3 具有 *N* 个状态并且接收含有 *N* 个符号的符号串 *xyz* 的机器

这里给出的抽吸引理是有限状态语言中的一种简化版本。比较强的版本也可以应用于有限状态语言，但是它的抽吸引理的特定更明显。

抽吸引理： 设 *L* 是一个有限的正则语言，那么必定存在着符号串 *x*，*y* 和 *z*，使得对于 $n \geq 0$，$y \neq \varepsilon$ 并且 $xy^nz \in L$。

抽吸引理告诉我们，如果一种语言是正则语言，就存在着一个符号串 *y*，这个 *y* 可以被适当地抽吸。然而，这并不意味着，如果一种语言能够对于某个符号串进行抽吸，这种语言就是正则语言。非正则的语言也会有符号串被抽吸，因此这个抽吸引理不能用来证明一种语言是正则语言。但是，抽吸引理可以用来证明某种语言不是正则语言，这时只需要证明，在某种语言中不能用适当的办法对符号串进行抽吸。

现在，让我们来证明，语言 a^nb^n（也就是由 *n* 个 *a* 后面跟着同样数目的 *b* 构成的语言）不是正则语言。这时必须证明，我们取的任何符号串 *s* 都不可能被分成 *x*，*y* 和 *z* 三个部分，使得 *y* 能够被抽吸。随意给一个由 a^nb^n 构成的符号串 *s*，可以用三种办法来分割 *s*，并且证明无论用哪种办法，都不可能找到某个 *y* 能够被抽吸。

1. *y* 只由若干个 *a* 构成。这意味着 *x* 也全都是由 *a* 组成的，*z* 包含了全部 *b*，而 *z* 的前面可能有若干个 *a*。但是，如果 *y* 全都由 *a* 组成，就意味着 xy^nz 中的 *a* 比 *xyz* 中的多。不过，这就意味着 *s* 中的 *a* 的数目比 *b* 的数目大，因而它不能成为 a^nb^n 的成员！
2. *y* 只由若干个 *b* 构成。这种情况与上面的相似。如果 *y* 全都由 *b* 组成，就意味着 xy^nz 中的 *b* 比 *xyz* 中的多，因此，*s* 中的 *b* 的数目比 *a* 的数目多。
3. *y* 由若干个 *a* 和若干个 *b* 构成。这意味着 *x* 只包含 *a*，*y* 只包含 *b*。这时，xy^nz 必定有一些 *b* 在 *a* 之前，因此，它不可能成为 a^nb^n 的成员！

由此可见，在 a^nb^n 中没有符号串能够被分割为 *x*, *y* 和 *z*，使得 *y* 能够被抽吸。所以，a^nb^n 不是正则语言。

a^nb^n 不是正则语言，但是 a^nb^n 是上下文无关语言。实际上，上下文无关语法只需要两条规则就可以给 a^nb^n 建立模型，这两条规则是：

$$S \rightarrow a\ S\ b$$

$$S \rightarrow \varepsilon$$

图 13.4 是使用这个语法推导句子 *aabb* 的剖析树。

上下文无关语言也有一个抽吸引理，这个引理可用来鉴别一种语言是不是上下文无关的，详细的讨论可参阅 Hopcroft and Ullman(1979)和 Partee et al.(1990)。

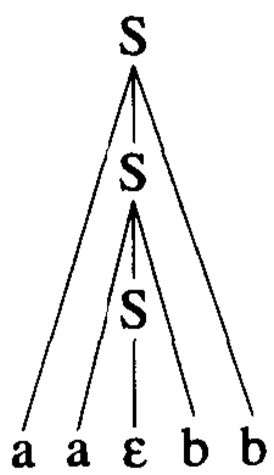


图 13.4　*aabb* 的上下文无关剖析树

13.2.2　英语和其他自然语言是正则语言吗

抽吸引理给我们提供了一个理论机制来理解关于“英语(或者作为形式语言的英语单词符号串的集合) 不是一种正则语言” 的著名论断。

Chomsky(1956, 1957)首先做出了这样的论断。他首次研究了语言$\{xx^R, x \in a, b^*\}$。x^R表示 “存在着逆向的x”，因此在这种语言的每个句子中，包含由若干个a和若干个b构成的符号串，后面跟着它的逆向符号串，或者这个符号串的 “镜像” (mirror image)。这种语言不是正则语言，Partee et al.(1990)利用将这种语言和正则语言aa^*bbaa^*求交的方法，证明了这种语言不是正则语言。求交后形成的语言是$a^nb^2a^n$；读者可以用抽吸引理来证明这个语言不是正则语言。

Chomsky 还证明了，英语语法中一个特殊的子集合与这种镜像结构语言是同态的 (isomorphic)。他研究了如下的英语句法结构，其中$S_1, S_2, \cdots, S_n$是英语中的陈述句：

- If S1, then S_2
- Either S_3, or S_4
- The man who said S_5 is arriving today

Chomsky明确地指出，这些都是英语的句子。另外，在每一种场合，每个结构的一部分和另一部分之间存在着词汇依存关系。If 后面要与 then 相对应 (而不能与 or 相对应)，Either 后面要与 or 相对应 (而不能与 because 相对应)。

这些句子在英语中还可以彼此叠套起来，我们可以造出如下的句子：

If *either* the man who said S_5 is arriving today *or* the man who said S_5 is arriving tomorrow, **then** the man who said S_6 is arriving the day after ...

正则语言对于替换 (substitution) 或同态 (homomorphism) 是封闭的。这意味着可以给上面的句子中的符号重新命名。让我们来研究如下的替换：

if	→	**a**
then	→	**a**
either	→	**b**
or	→	**b**
其他单词	→	ε

现在把这样的替换应用于上面的句子，就可以得到如下的句子：

abba

上面说明的具有镜像性质的这个句子，是不能使用有限状态方法来描述的。如果假定 *if*, *then*, *either* 和 *or* 可以无限地叠套，那么英语就是与xx^R同态的，其中$x \in a, b^*$。因此，英语不是正则语

言。显而易见，这样的结构是不可能无限地叠套的（在经过几次叠套之后，句子就很难理解了）；我们在 13.4 节中再讨论这个问题。

Partee et al.(1990)给出了英语不是正则语言的第二个证明。这个证明是基于带有中心-嵌套结构（centerembedded structure）(Yngve, 1960)的一类著名的句子；这里是这种句子的一个变体：

The cat likes tuna fish.

The cat the dog chased likes tuna fish.

The cat the dog the rat bit chased likes tuna fish.

The cat the dog the rat the elephant admired bit chased likes tuna fish.

与上面的那种either/or句子相比较，当这种句子变得更复杂时，理解起来也就更困难了。不过，现在让我们假定，英语语法允许无限数目的嵌套。这样，为了证明英语不是正则语言，就需要证明这类句子是与某种非正则语言同态的。在这些句子中，因为每个在前面的*NP*都必须与一个动词相联系，这些句子的形式可以表示为：

$$(\text{the} + \text{noun})^n (\text{transitive verb})^{n-1} \text{ likes tuna fish.}$$

要进行这种证明的基本思想是：证明这种句子的结构可以通过把英语与一个正则表达式相交而产生，然后使用抽吸引理证明，这样形成的句子不是正则表达式。

为了建立一个简单的正则表达式，以便把它与英语相交从而产生这样的句子，我们来给名词组（*A*）和动词（*B*）定义如下的正则表达式：

$A = \{the\ cat, the\ dog, the\ rat, the\ elephant, the\ kangaroo, ...\}$

$B = \{\text{chased, bit, admired, ate, befriended, ...}\}$

现在，如果取正则表达式/*A* B** likes tuna fish/，并且把它与英语（把英语看成符号串的集合）相交，结果形成如下的语言：

$$L = x^n y^{n-1}\ likes\ tuna\ fish,\ \ x \in A, y \in B$$

通过抽吸引理，我们可以证明语言*L*不是正则语言。因为英语与正则语言的交不是正则语言，所以英语不是正则语言（这是由于正则语言在交运算下是封闭的）。

上面的两个论据都是以句法为基础的。也有一些证明英语没有这种有限状态性质的论据是以形态为基础的。这些形态学的论据是一种不同类型的论据，因为它们并没有证明英语的形态不能是正则的，只是证明了英语形态的上下文无关模型更为优雅 (elegance)，更容易把某些描写进行泛化。我们以Sproat(1993)对于前缀*en-*的研究来说明这个问题。与英语的其他动词一样，加前缀构成的英语动词也可以加后缀*-able*。例如，动词*enjoy*和*enrich*可以加后缀形成*enjoyable*和*enrichable*。但是，英语名词或形容词的词干本身不能加后缀*-able*（所以有**joyable*和**richable*）。换句话说，如果已经加了前缀*en-*形成动词，就可以再加上后缀；如果没有加前缀*en-*，则不能加后缀*-able*。

其原因是非常简单的：*en-*可以产生动词，而*-able*只能加在动词的后面。但是，如果用正则语法来表达这样的事实会使人觉得很烦，而且显得很啰嗦；这种表达似乎可以有两条路，一条路通过*joy*，一条路通过*enjoy*，如图 13.5 所示。

这些形态学事实很容易用上下文无关语法来表达，作为练习留给读者。

这些反对正则语法的“优雅”论据也适用于句法现象。例如，有的学者曾经主张英语中数的一致性不能使用正则语法（甚至上下文无关语法）来描述。但是，事实上正如Pullum and Gazdar(1982)指出的，一个简单的正则语法就可以模拟数的一致关系。他们研究了下列具有长距离一致关系的句子：

Which *problem* did your professor say she thought *was* unsolvable?
Which *problems* did your professor say she thought *were* unsolvable?

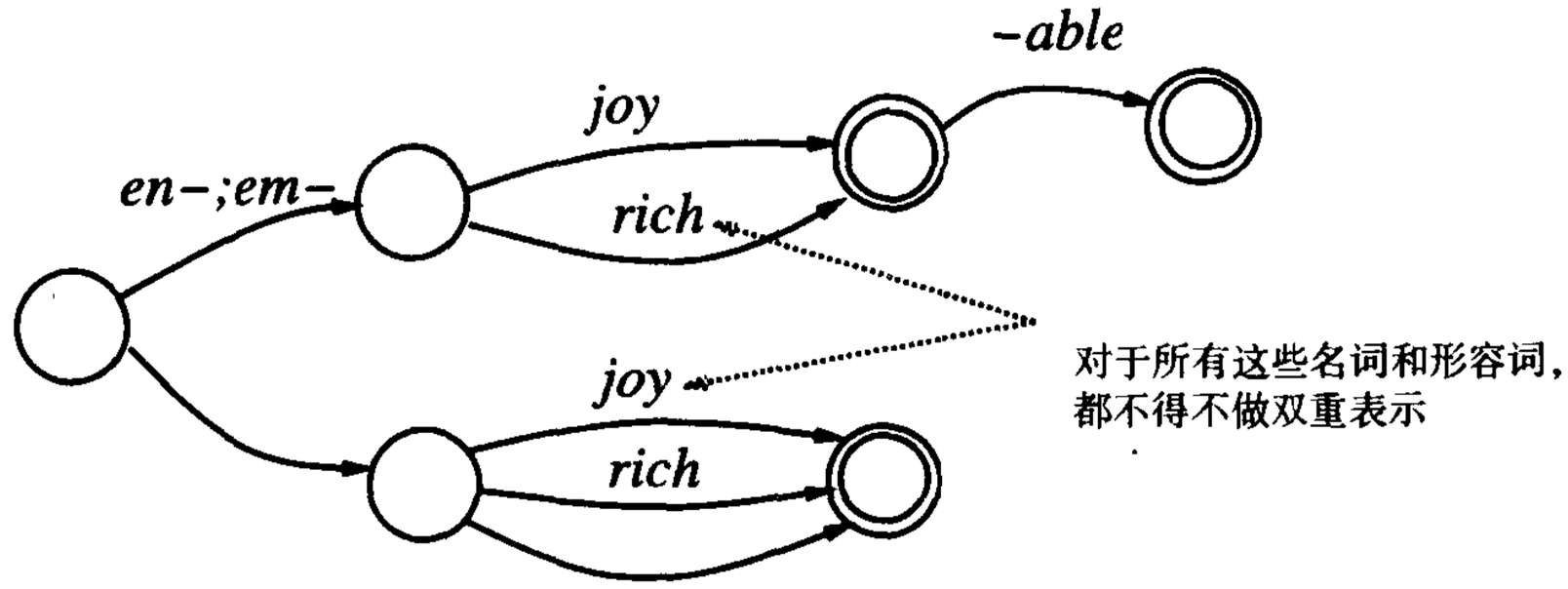


图 13.5　英语形态分析的 FSA 的一部分(Sproat, 1993)

模拟这些句子的正则（右线性）语法是：

$$S \rightarrow \text{Which problem did your professor say T}$$
$$S \rightarrow \text{Which problems did your professor say U}$$
$$T \rightarrow \text{she thought T} \mid \text{you thought T} \mid \text{was unsolvable}$$
$$U \rightarrow \text{she thought U} \mid \text{you thought U} \mid \text{were unsolvable}$$

可见，一个正则语法就可以模拟英语的一致关系。正如在第9章中看到的，这种语法的问题并不在于它的计算能力，而在于它是不是优雅；这样的语法可能会由于规则的数目过多而形成规则爆炸。不过，对于计算复杂性来说，一致关系并不能成为英语不是正则语言的论据。

13.3　自然语言是上下文无关的吗

前一节说明了英语（把它看成符号串的集合）看起来不是正则语言。接下来，读者当然会问如下的问题：英语是不是上下文无关语言。这个问题是首先由 Chomsky(1956)提出的，研究这个问题的历史也很有趣，很多已经发表的论文都试图证明英语和其他自然语言不是上下文无关语言，但是，除了两个论据之外，其他已发表的论据都被证明是不成立的。在这两个正确（或者说至少目前还没有被推翻）的论据中，一个论据来自瑞士德语的一种口语方言，另一个论据是 Bambara 语的形态。Bambara语是在马里及其邻国讲的一种称为Mande语的西北方言(Culy, 1985)。有兴趣的读者可参阅 Pullum(1991, pp. 131–146)，其中对于这样的论据正确与否都进行了非常有趣而诙谐的讨论，这是一段饶有趣味的历史。本节中只总结性地介绍一下其中的一种论据，即来自瑞士德语的论据。

正确的论据和大部分不正确的论据都使用了如下的语言事实，即下面的语言的相似特性在于这种语言不是上下文无关语言：

$$\{xx \mid x \in \{a,b\}^*\} \tag{13.1}$$

这种语言的句子是由前后两个完全等同的符号串毗连而成的。下面的相关的语言也不是上下文无关的：

$$a^n b^m c^n d^m \tag{13.2}$$

这些语言的上下文无关特性可以使用上下文无关语言的抽吸引理来加以证明。

为了证明这些语言不是上下文无关语言的子集合，我们来证明这样的自然语言具有 xx 语言的特性，这种特性称为“交叉序列依存”（cross-serial dependency）。在交叉序列依存中，语言中的单

词或更大的结构按从左向右的顺序联系起来成为图13.6中的形式。如果一种语言具有任意长的交叉序列依存，它就可以映射为 *xx* 语言。

$$x_1\ x_2\ \cdots\ x_n\ \cdots\ y_1\ y_2\ \cdots\ y_n$$

图 13.6　交叉序列依存的图示

Huybregts(1984)和 Shieber(1985a)都成功地证明，在 Zürich 地区的瑞士德语口语方言具有交叉序列约束，这使得这种语言的某些部分等价于非上下文无关语言 $a^nb^mc^nd^m$（他们的证明工作印证了前面提到的科学中的多重发现的规律）。从直觉上说，瑞士德语允许这样的句子：一个给格名词的符号串后面跟着一个宾格名词的符号串，后面又跟着一个要求给格的动词的符号串，再跟着一个要求宾格的动词的符号串。

这里介绍 Shieber(1985a)的证明。首先，Shieber 注意到，在瑞士德语中允许动词和它们的论元按照交叉序列的顺序排列。假定下面介绍的所有例子前面都有符号串 *Jan säit das*（Jan says that），它们是 *Jan säit das* 的从句：

... mer	*em Hans*	*es*	*huus*	*hälfed*	*aastriiche.*	（13.3）
... we	Hans/DAT	the	house/ACC	helped	paint.	
我们	Hans/ 给格		房子 / 宾格	帮助	漆	

"... we helped Hans paint the house."

……我们帮助 Hans 漆房子。

注意语义关系的交叉序列性质：两个名词出现在两个动词的前面，*em Hans*（Hans 的给格）是动词 *hälfed*（帮助）的论元，*es huus*（house 的宾格）是动词 *aastriiche*（漆）的论元。此外，在名词和动词之间还存在着交叉序列的格依存关系：*hälfed*（帮助）要求给格，所以 *em Hans* 是给格，*aastriiche*（漆）要求宾格，所以 *es huus* 是宾格。

Shieber 指出，这种格标记还会发生在三重嵌套交叉序列从句的场合，如下所示：

...mer	*d' chind*	*em Hans*	*es*	*huus*	*haend*	（13.4）
...we	the chidren/ACC	Hans/DAT	the	house/ACC	have	
我们	小孩们 / 宾格	Hans/ 给格		房子 / 宾格		

wele	*laa*	*hälfe*	*aastriiche.*
wanted to	let	help	paint.
想	让	帮助	漆

"... we have wanted to let the children help Hans paint the house."

"……我们想让孩子们帮助 Hans 漆房子。"

Shieber指出，在这些句子中，如果所有给格的名词词组*NP*位于所有宾格的名词词组*NP*之前，所有要求给格次范畴化的动词*V*位于所有要求宾格次范畴化的动词*V*之前，那么这样的句子就是可以接受的。例如：

Jan säit das mer (d'chind)* (em Hans)* es huus haend wele laa* hälfe* aastriiche.

现在来考虑上面的这个正则表达式，把它记为R。由于这是一个正则表达式（可以看出它只有毗连运算和 Kleene 星号运算），它必须定义一个正则语言，并且我们可以把 R 与瑞士德语求交。如果求交的结果是上下文无关的，那么瑞士德语就是上下文无关的。

然而，事实表明，瑞士德语要求带给格宾语（*hälfe*）的动词的数目必须等于给格名词词组 NP 的数目（*em Hans*）。类似地，还要求带宾格宾语的动词的数目（*aastriiche*）必须等于宾格名词词组的数目（*es huus*）。此外，在这种类型的从属从句中，可以出现任意数目的动词（取决于使用的约束情况）。这意味着，把这种正则语言与瑞士德语求交，可以得到如下的语言：

L = Jan säit das mer (d'chind)^n (em Hans)^m es huus haend wele (laa)^n (hälfe)^m aastriiche.

但是，这种语言的形式为 $wa^nb^mxc^nd^my$，这是一种非上下文无关语言。

所以，我们的结论是：瑞士德语是非上下文无关语言。

13.4　计算复杂性和人的语言处理

从前面所讲的内容可以看出，我们用来论证英语是非有限状态语言的那些句子（例如中心嵌套句子），理解起来是很困难的。如果你是一个说瑞士德语的人（或者你有一个说瑞士德语的朋友），就会发现在瑞士德语中那些很长的交叉序列句子也是很难理解的。因此，Pullum and Gazdar(1982)指出：

> 确切地说，在证明英语不是上下文无关的各种论证中所引用的那些结构类型，在人处理语言的系统中看来会引起巨大的困难……

这使我们再次使用“计算复杂性”（complexity）这个术语。在前一节中谈到了语言的计算复杂性。在这里，我们转而研究既带有计算色彩也带有更多心理色彩的问题：单个句子的计算复杂性问题。为什么有的句子理解起来很困难呢？这种情况是否能告诉我们关于计算过程的某些信息呢？

很多因素都会造成句子理解的困难：意思太复杂，句子的歧义特别严重，使用了罕用的单词，书写质量太差，等等。第 12 章介绍的花园幽径句是很复杂的，其复杂性是由于在人的剖析过程中选择的临时歧义句子不恰当而造成的。但是，还存在着另一种特殊的复杂性（通常称为“语言复杂性”或“句法复杂性”），这种复杂性与上一节介绍的形式语言的复杂性存在着有趣的关系。这些句子的复杂性不是来自罕用单词或意思难以理解，而是句法结构和人的记忆的有限性共同作用的结果。下面是一些例句，依据了 Gibson(1998)的总结，当人们在阅读这些例句时往往会感到困难（我们用 # 来表示那些会引起特殊处理困难的句子）。在下面的例句中，（ii）中的例句比（i）中的例句更复杂：

(i) The cat likes tuna fish. (13.5)

(ii) #The cat the dog the rat the elephant admired bit chased likes tuna fish.

(i) If when the baby is crying, the mother get upset, the father will help, so the grandmother can rest easily. (13.6)

(ii) #Because if when the baby is crying, the mother gets upset, the father will help, the grandmother can rest easily.

(i) The child damaged the pictures which were taken by the photographer who the professor met at the party. (13.7)

(ii) #The pictures which the photographer who the professor met at the party took were damaged by the child.

(i) The fact that the employee who the manager hired stole office supplies worried the executive. (13.8)

(ii) #The executive who the fact that the employee stole office supplies worried hired the manager.

关于这类句子的早期研究已经注意到，这些句子都有嵌套（nesting）或中心嵌套（center-embedding）(Chomsky, 1957; Yngve, 1960; Chomsky and Miller, 1963; Miller and Chomsky, 1963)。这

就是说，它们包含的结构都是一个句法范畴 *A* 嵌入到另一个范畴 *B* 中，并且被其他单词（X 和 Y）所包围：

$$[_B \text{X} [_A] \text{Y}]$$

在上面的每个例子中，(i)有零个或一个嵌套，而(ii)则有两个或更多的嵌套。例如，在上面的例句（13.5ii）中，有三个简化的关系从句存在嵌套关系：

[$_S$ The cat [$_S$· the dog [$_S$· the rat [$_S$· the elephant admired] bit] chased] likes tuna fish].

在上面的例句（13.6ii）中，*when* 从句嵌套在 if 从句中，*if* 从句嵌套在 *because* 从句中：

#[Because [if [when the baby is crying, the mother gets upset], the father will help], [the grandmother can rest easily]].

在例句（13.7ii）中，关系从句 *who the professor met at the party* 嵌套在 *the photographer* 和 *took* 之间。关系从句 *which the photographer ... took* 嵌套在 *The pictures* 和 *were damaged by the child* 之间：

#The pictures [which the photographer [who the professor met at the party] took] were damaged by the child.

我们能不能在解释这些嵌套结构的困难时，只说因为它们不合英语的语法呢？答案似乎是否定的。在例句（13.5ii）至例句（13.8ii）中的复杂句子所用的结构与在例句（13.5i）至例句（13.8i）中的比较容易的句子所用的结构是相同的。容易句子和复杂句子之间的差别似乎是根据嵌套的数目来调节的。然而，我们无法在语法中规定只允许 *N* 个嵌套而不允许 *N*+1 个嵌套。

进一步说，这些句子的复杂性看起来是一种语言处理中的现象；关于人的剖析机制的某些事实不能处理这种多重嵌套的现象。如果复杂性是一种“剖析”中的事实，而不是语法中的事实，那么，由于同样的原因，我们就可以期望在其他语言中的句子也具有这样的复杂性。这就是说，尽管其他语言有着不同的语法，但我们可以假定，人的剖析的体系结构是各种语言共享的，从一种语言到另一种语言是相同的。

这种类型的多重嵌套结构在其他语言中看起来也存在，而且处理起来同样很困难。例如，日语中允许一个简单的嵌套从句，再增加嵌套就会使句子变得难以处理(Cowper, 1976; Babyonyshev and Gibson, 1999)。

Ani-ga	*imooto-o*	*ijimeta.*	（13.9）
older-brother-NOM	younger-sister-ACC	bullied	
哥哥 -NOM	妹妹 -ACC	欺负	

“My older brother bullied my younger sister.”

“我哥哥欺负我妹妹。”

Babiisitaa-wa	*[[ani-ga*	*imooto-o*		（13.10）
babysitter-TOP	[[older-brother-NOM	younger-sister-ACC		
临时保姆 -TOP	[[哥哥 -NOM	妹妹 -ACC		
ijimeta]	*to]*	*itta.*		
bullied]	that]	said		
欺负]	]	说		

“The babysitter said that my older brother bullied my younger sister.”

“临时保姆说，我哥哥欺负我妹妹。”

#Obasan-wa	*[[Babiisitaa-ga*		*[[ani-ga*			（13.11）
aunt-TOP	[[babysitter-NOM		[[older-brother-NOM			
大娘 -TOP	[[临时保姆 -NOM		[[哥哥 -NOM			
imooto-o	*ijimeta]*	*to]*	*itta]*	*to]*	*omotteiru.*	
younger-sister-ACC	bullied]	that]	said]	that]	thinks	
妹妹 -ACC	欺负]	]	说]	]	想	

“#My aunt thinks that the babysitter said that my older brother bullied my younger sister.”

“我的大娘想：临时保姆说，我哥哥欺负我妹妹。”

对于这些复杂性效应，已经提出了一些解释，其中很多是根据记忆来解释的。这种解释所依赖的直觉是：每个嵌套都要求一些记忆资源来存储。多层嵌套的句子或者要使用很多记忆资源，或者要形成多重的记忆跟踪，从而造成了不同记忆之间的混淆。其结果就使得这样的句子处理起来非常困难。

例如，Yngve(1960)提出，人的剖析是基于一个容量有限的栈（satck）进行的。这种基于栈的剖析把不完全的短语结构规则存储到栈中；如果多重的不完全短语在这个栈中嵌套起来，那么栈就要对每个不完全的规则进行登录。Yngve认为，这种不完全的短语结构规则越多，人的剖析器就需要把它们存储到栈中，因此句子也就越复杂。根据Yngve的直觉，这种栈的有限性意味着英语事实上是正则语言，而不是上下文无关语言，其原因在于：如果上下文无关语法的栈的容量是有限的，那么，它就可以用有限自动机来模拟。

作为这种模型的一种扩充，(Miller and Chomsky, 1963)提出，自嵌入结构（self-embedded）是特别困难的。在自嵌入结构中，一个句法范畴 *A* 被嵌入到另一个 *A* 中，而周围被其他单词包围（X 和 Y）：

$$[_A \, X \, [A] \, Y]$$

这样的结构处理起来会非常困难，因为基于栈的剖析可能会把栈中规则的两个副本混淆起来。自嵌入的问题可以自然地用基于激活（activation-based）的模型来模拟，这种模型对于一个特定的规则只有一个副本。

尽管这些经典的基于剖析的解释在直觉上具有吸引力，而且这种解释与形式语言的计算复杂性问题的联系也很美妙，但这样的解释未必正确。其中的一个问题是，还有很多句法复杂性的效应是这些模型无法解释的。例如，在具有同样数目的嵌套的句子之间，它们的复杂性还有明显的差别。例如，在抽取主语的关系从句（13.12ii）和抽取宾语的关系从句（13.12i）之间的差别是众所周知的：

(i) [$_S$ The reporter [$_{S'}$ who [$_S$ the senator attacked]] admitted the error]. （13.12）

“议员攻击的那个记者承认了错误。”

(ii) [$_S$ The reporter [$_{S'}$ who [$_S$ attacked the senator]] admitted the error].

“攻击议员的那个记者承认了错误。”

抽取宾语的关系从句处理起来更困难，例如可以根据阅读这些句子所需的时间的多少来测量困难的大小(Ford, 1983)，还可以根据其他因素，读者可以参阅Wanner and Maratsos(1978), King and Just (1991)以及 Gibson(1998)。不同的研究者提出了不同的假设和不同的因素来解释这些复杂性的区别。

例如，MacWhinney 和他的同事们提出(MacWhinney, 1977, 1982, MacWhinney and Csaba Pléh, 1988)，从一个从句的参与者到另一个从句的参与者的观察点的转移（shift perspective）会引起理解

上的困难。抽取宾语的关系从句要求两个观察点的转移(从主要句子的主语到关系从句的主语,然后再反过来),而抽取主语的关系从句不要求观察点的转移(主要句子的主语与关系从句的主语相同)。这种困难的另一个潜在来源在于,抽取的宾语是第一个名词(the reporter),这个名词起着两个不同的题元作用(thematic role),在一个句子中是施事(agent),在另一个句子中是受事(patient)。这种角色指派的冲突会引起理解上的困难(Bever, 1970)。

Gibson(1998)指出了宾语抽取和主语抽取之间的另一个重要区别:在宾语抽取的句子中,当任何动词出现之前要先出现两个名词。读者必须先把这两个名词记住,但是还不知道它们在句子中究竟起什么作用。先记住若干个名词短语而又不能马上把它们与句子的整个含义结合在一起,会使读者难以理解句子。

根据这样的观察,Gibson提出了句法预示的位置理论(Syntactic Prediction Locality Theory,简称SPLT)。SPLT认为,与结构相关的句法的记忆负担(memory load)是与构成一个完整句子所需的每个单词相关的记忆负担的总和。在一个句子全部完成之前,具有多个名词短语但还未出现动词的句子部分,将要求它后面出现多个动词,使得记忆的负担相当高。记忆的负担还与有多少新的短语或话语参照需要在同一时间存储在记忆中有关系。因此,如果在预示一个单词时,还介入了新的话语参照,那么这个单词的记忆负担就会比较高。这样,尽管在一个序列中还没有结合在一起的若干个*NP*非常复杂,如果这个序列中的两个*NP*中有一个是代词,而这个代词已经在话语中有了参照,那么这个序列的复杂性就比较小。例如,下面的句子尽管有两层嵌套的关系从句,但也是可以处理的,因为最内层的*NP*(*I*)没有导入新的参照实体。

(a) A syntax book [that some Italian [that I had never heard of] wrote] was published by MIT Press.(Frank, 1992) (13.13)

(b) The pictures [that the photographer [who I met at the party] took] turned out very well.(Bever, 与 Gibson 的私人通信)

总的来说,认为句子处理的复杂性与记忆有关的一些早期研究从某个角度看来似乎是正确的;自然语言和形式语言的复杂性都是由于需要把很多未整合好的东西保存到记忆中引起的。这是一个含义深刻而令人心醉的关于语言处理的重要发现。不过,形式语言和自然语言的复杂性之间的关系,并不像Yngve等人的想法那样简单。要确切地说明究竟是什么因素在复杂性中起作用,是一个激动人心的研究领域,这方面的调查研究现在才刚刚开始。

13.5 小结

本章介绍了两种不同的关于复杂性的思想:形式语言的复杂性和句子的复杂性,如下所示:

- 语言可以用它的生成能力来刻画。如果一种语法能够定义的语言通过其他语法不能定义,就说这种语法比其他语法具有更大的生成能力或复杂性。Chomsky层级是建立在语法生成能力基础上的不同语法的层级。这些语法包括Turing等价语法、上下文有关语法、上下文无关语法和正则语法。
- 抽吸引理(pumping lemma)可以用于证明一种给定的语言不是正则语言。英语不是正则语言,一些些使英语变成非正则语言的句子类型使人们在分析这些句子类型时感到非常困难。数十年来人们试图证明英语不是正则语言,而是上下文无关语言。与此相反,瑞士德语的句法和Bambara语的形态是非上下文无关的,它们似乎需要使用上下文有关语法来描述。

- 对中心嵌套句子进行剖析时会感到很困难。很多理论都一致认为，这种困难是由人们剖析时的记忆的有限性（memory limitation）引起的。

13.6　文献和历史说明

Chomsky(1956)首先提出了这样的问题：有限状态语法或上下文无关语法是不是可以充分地描述英语的句法。他在文章中提出，英语句法包含的“一些例子是不容易用短语结构语法来解释的”，这促使他研究句法转换的问题。Pullum(1991, pp. 131–146)在历史上明确地研究了关于自然语言的非上下文无关性质（non-context-free-ness）。证明自然语言的非上下文无关性质的早期历史在Pullum and Gazdar(1982)中做了总结。抽吸引理是首先由Bar-Hillel et al.(1961)提出的，他们也给出了关于有限状态语言或上下文无关语言的封闭性和判定性的一些重要证明。关于上下文无关语言的抽吸引理的详细说明也得于Bar-Hillel et al.(1961)的研究，读者可以参阅诸如Hopcroft and Ullman(1979)等关于自动机理论的教材。

Yngve认为，如果人的剖析是有限状态的，中心嵌套句子的困难就可以得到解释；Church在硕士论文Church(1980)中注意到了Yngve的这个观点。他证明了实现这种思想的有限状态剖析器也能够解释很多其他语法现象和心理语言学现象。在认知模型的研究领域转向研究更细致的复杂性模型时，Church的工作可以看成是在语音处理和语言处理的领域向有限状态模型回归的开始，这成为了20 世纪 80 年代和 90 年代自然语言处理研究的一个特点。

还有其他一些研究复杂性的方法，我们没有更多篇幅在这里讨论这些问题。其中的一个问题是：语言处理是不是一个NP完全（NP-complete）问题。NP完全问题是一类问题的名字，是指那些估计处理起来特别困难的问题。Barton et al.(1987)证明了关于自然语言识别和剖析的复杂性的一些结果，其中指出了如下两点：

1. 在一个潜在地无限长的句子中，保持词汇和一致关系的特征歧义，引起在某些基于合一的形式化方法中（如词汇功能语法）识别句子的问题，是NP完全问题。
2. 双层形态剖析（或者甚至只是词汇和表层形式之间的映射）也是NP完全问题。

在当前的一些研究，已经开始把语言处理的复杂性和诸如Kolmogorov复杂性的信息论测度结合起来(Juola, 1999)。

第三部分　语义的计算机处理

语义学是关于语言话段（utterance）的意义的研究。本书的这一部分将重点论述捕捉这种意义的形式表示以及从话段到这种意义表示的映射算法。整个部分的关键问题是建立模型来描述一个话段的意义怎样与组成这个话段的短语、单词和语素的意义相互联系起来。关于说话人和听话人以及话段的上下文环境的问题，将在本书的第四部分中讨论。

这一部分首先探讨话段意义的表示方法，着重讨论一阶谓词演算（First Order Predicate Calculus，简称FOPC）的使用方法。接着，探讨组成性语义分析的各种理论和实际方法以及它们在诸如问答系统（question answering）和信息抽取（information extraction）等实际问题中的应用。然后，转向讨论单个单词的意义、意义在词表组织中的作用、单词意义的歧义消解算法等问题。最后，讨论关于信息检索的问题。信息检索是一个重要的应用领域，而信息检索差不多完全是在个别单词的意义基础上进行的。

第14章　意义的表示法

ISHMAEL: *Surely all this is not without meaning*,

Herman Melville, *Moby Dick*[①]

在这里介绍并将在接下来的四章中进一步探讨的语义学方法建立在这样的概念的基础上：语言话段的意义可以使用形式方法来捕捉。这种形式化方法称为意义表示（meaning representation）。相应地，用来说明这种意义表示的句法和语义学的框架称为意义表示语言（meaning representation language）。这种意义表示所起的作用与前几章介绍的音位表示、形态表示和句法表示是很类似的。

之所以需要这样的语义表示，是因为无论是没有加工过的语言输入，还是用前面研究过的任何转录机推导出来的结构，都不能帮助进行我们所希望的这种语义处理。更具体地说，我们所需要的意义表达能够在从语言输入到与语言输入意义有关的各种具体任务所需的非语言知识之间架起一座桥梁。

为了说明这样的思想，我们来研究日常生活的语言活动中要求进行语义处理的下列问题：

- 在考试时回答文章中的问题
- 在饭店里阅读菜单来点菜
- 通过阅读说明书学习使用新软件
- 根据菜谱做菜

显而易见，简单地使用前面几章讨论过的音位表示、形态表示和句法表示，并不能帮助我们解决这些问题。为了解决这些问题，需要把包含在这些问题中的语言因素与用于成功地完成这些任务所需的非语言的世界知识（world knowledge）结合起来。例如，为了解决上面的问题，需要如下的世界知识：

- 要回答文章中的问题并对它们进行分级，要求具有有关的背景知识，例如关于问题主题的背景知识、关于学生知识水平的背景知识以及关于如何按正规方式回答这些问题的背景知识等。
- 阅读菜单和决定点什么菜，对于到哪里吃饭提出建议，根据菜谱做菜以及编写一个新的菜谱，都要求对于食品有深入的知识，知道怎样做菜，知道人们喜欢吃什么以及喜欢到什么饭店等。
- 通过阅读说明书来学习一种新软件，或者对具体做法提出建议，都要求对当前的计算机，对于有关的软件及其使用都有深入的知识，并且对于用户也要有一般的知识。

在这里所研究的表示方法中，我们取语言的输入来构造意义表示，这样的意义表示要使用那些与表示日常生活中的常识性世界知识同样的材料来构成。产生这样的意义表示并且把它们指派给语言输入的过程称为语义分析（semantic analysis）。

为了把这个概念说得更具体，我们来研究图14.1，其中说明了*I have a car*这个句子使用的四种常见意义表示语言。第一行是一阶谓词演算（First Order Predicate Calculus）的一个语句，详见14.3

① ISHMAEL：所有的这一切都不是没有意义的。

节中的讨论；中间是一个语义网络（Semantic Network）的图示，详见 14.5 节中的进一步讨论；第三行包含一个概念依存（Conceptual Dependency）的图示和一个基于框架的表示。第 16 章将详细讨论概念依存理论，14.5 节将介绍基于框架的表示。

$$\exists x,y Having(x) \wedge Haver(Speaker,x) \wedge HadThing(y,x) \wedge Car(y)$$

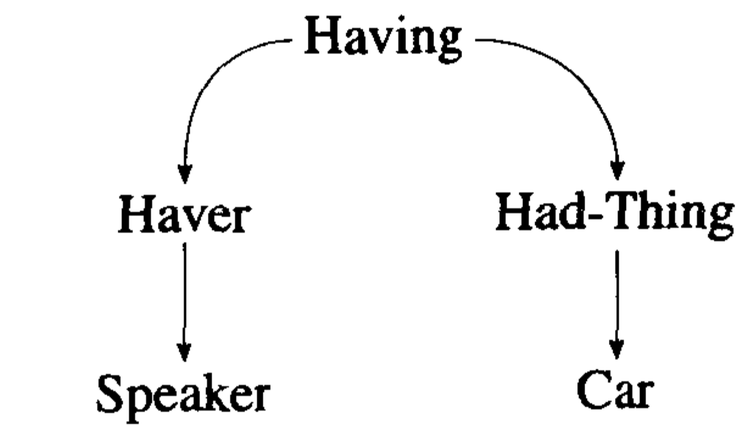


图 14.1　句子 *I have a car* 的意义表示的样本：一个符号表、两个有向图和一个记录结构

尽管这四种不同的表示方法有很多差别，但在抽象层次上它们都有一个共同的概念基础，即，意义表示是由符号集合所组成的结构构成的。如果适当地对这些符号进行安排，那么这些符号结构相应于在某个被表示的世界中的实体以及这些实体之间的关系。在这种情况下，这四种意义表示都使用了分别对应于说话人、汽车以及说明彼此之间的所属性质的一些关系。

应该注意，在所有这四种方法中的这些意义表示，至少可以从两个不同视角来看：一方面，把它们看成是特定语言输入 *I have a car* 的意义表示，另一方面，把它们看成是在某个世界中的事件状态的表示。正是这种双重的视角使得这些意义表示可以用来把语言输入和世界以及和我们关于世界的知识联系起来。

本书的这一部分与前一部分是平行的。我们将交替地讨论意义表示的本质并讨论能够产生它们的计算过程。更具体地说，本章将介绍在意义表示中需要的一些基础知识，第 15 章将介绍一些给语言输入指派意义的技术。第 16 章将探讨一些与单词的意义有关的复杂表示，第 17 章将探讨用于揭示这些意义表示的鲁棒的计算方法。

注意，由于本章的强调重点是意义表示的基本要求，因此一些非常重要的问题将在后面的章节中讨论。更具体地说，本章的重点是表示句子的字面意义（literal meaning），所以关注的是那些与单词的常规意义紧密联系的表示，而不反映它们出现的上下文环境。这种表示当涉及到成语和比喻等现象时就会显得不足，这些缺点将在下面的两章中讨论。在第 18 章和第 19 章中将讨论上下文环境在确定句子的深层意义时的作用。

本章分成三节。14.1 节介绍在意义表示语言中需要的某些实际计算要求。14.2 节讨论语言的意义结构。14.3 节介绍一阶谓词演算，这种演算在历史上是用来研究意义的主流技术。

14.1　意义表示的计算要求

首先我们来研究为什么需要意义表示，意义表示能够为我们做些什么。为了集中讨论这个问题，我们将详细研究给旅游者提供饭店建议的问题。在讨论中假定我们已经有一个计算机系统能够

接收旅游者的口语提问，并且能使用相关领域的知识库构造出相应的回答。给出的一系列的例子都用于介绍意义表示必须实现的基本要求，以及在设计这样的意义表示的过程中不可避免地会产生的各种复杂问题。在每个例子中，我们将检查提问的意义表示在满足这种要求的过程中所起的作用。

14.1.1 可能性验证

让我们首先来研究下面的简单提问：

Does Maharani serve vegetarian food? (14.1)

这个例子说明了对于意义表示的最基本的要求：必须能够使用意义表示来决定句子的意义和我们所知道的世界之间的关系。换句话说，必须能够确定意义表示的真实性。实现这种思想的最直接办法是，使一个系统能够把输入的意义表示与存储着关于世界的信息的知识库（knowledge base）中的表示相比较或匹配。

在这个例子中，假定提问的意义包含了命题 Maharani serves vegetarian food 的意义，把它作为一个组成部分。现在，我们把这个表示简单地注释为：

Serves (*Maharani*, *VegetarianFood*)

这个输入的意义表示将与关于饭店的集合的事实知识库相匹配。如果系统在知识库中找到一个表示与输入的命题相匹配，就可以返回一个肯定的回答。否则，如果关于当地饭店的知识是完全的，系统就说“不”；如果有理由证实关于饭店的知识是不完全的，系统就说“不知道”。

这个概念称为“可能性验证”（verifiability）。可能性验证是系统把意义表示所描述的情况与在知识库中所模拟的某个世界的情况进行比较的一种能力[①]。

14.1.2 无歧义表示

与我们研究过的其他领域一样，语义学的领域也遇到了歧义问题。具体地说，单个的语言输入根据发生环境的不同可以指派不同的意义表示，这在语言学中是天经地义的事情。

我们来研究来自 BERP 语料库的例子：

I wanna eat someplace that's close to ICSI. (14.2)

给出动词 *eat* 可允许的论元结构，这个句子的意思可以是说话人在某个比较近的地方吃东西；如果把说话人表示为 Godzilla，那么这个句子的意思也可以是说话人想吞吃某个比较近的地方。系统生成的对于这个提问的回答将取决于系统选择哪个作为正确的解释。

由于这样的歧义在一切语言的各种类别中都普遍存在，所以需要一些方法来决定某种解释比另外的解释更优先（或者更不优先）。各种语言现象都存在着这样的歧义，处理这些歧义的技术将在接下来的四章中讨论。

但是，我们在本章中只考虑意义表示与歧义的状况，而不考虑如何得到正确的解释。因为要对语言输入的语义内容进行推理，并且要根据它来采取行动，所以一个输入的意义的最后表示必须与任何歧义没有关系。因此，无论在原来的输入中有没有歧义，关键的问题是意义表示语言支持的表示只能有一个无歧义的解释[②]。

---

① 这是“可能性验证”的一个非常实际的特征。关于这个概念的更多理论性论述将在 14.6 节中进行简要介绍。

② 这并不妨碍在导致一个单独的无歧义表示的过程中，使用的中间语义表示还保持着某些层面的歧义。这种表示的例子将在第 15 章中讨论。

与歧义性密切相关的概念是模糊性（vagueness）。正如歧义性那样，对于特定的输入，在根据其意义表示确定究竟是什么意思时，模糊性也会造成困难。不过，模糊性不会产生多个意义表示。

作为例子，我们来研究如下的提问：

I want to eat Italian food.　（14.3）

当使用短语*Italian food*时，饭店的服务人员可以为提出合理建议提供丰富的信息，因为在顾客真正需要吃什么时，这个短语的意义并不太模糊。因此，当为了其他目的需要做更具体的表示时，这个短语中的模糊表示对于一定的目的来说可能是恰到好处的。在这种情况下，意义表示语言保持一定程度的模糊性，对于支持意义表示还是有好处的。注意，区别歧义性和模糊性并不总是那么容易。Zwicky and Sadock(1975)对于判别它们之间的区别提出了一套有用的检测方法。

14.1.3　规范形式

由于一个句子可能指派多个意义，这导致了与此相关的一种现象：不同的输入可能指派同样的意义表示。我们来研究对于例句（14.1）的下列不同表达方式：

Does Maharani have vegetarian dishes?　（14.4）

Do they have vegetarian food at Maharani?　（14.5）

Are vegetarian dishes served at Maharani?　（14.6）

Does Maharani serve vegetarian fare?　（14.7）

面对这些不同单词的不同用法以及变化各异的句法分析，期望上述句子具有本质上不同的意义表示，不是毫无道理的。但是，这将会给确定意义表示的真值的匹配方法带来我们所不期望的后果。如果在系统的知识库中对于所提的问题只有一个意义表示，那么这些不同的意义表示将不能进行正确的匹配。当然，也可以把同样事实的各种可能的不同意义表示存储在知识库中，但为了保持这个知识库的一致性，将会引起数不胜数的问题。

学者们提出了摆脱这种两难困境的出路，其事实根据是：对于这些不同的意义表示的回答，在所有的情况下都应该是相同的；至少对于饭店向顾客提出建议这个目的来说，我们可以说这些意义表示都意味着同样的事情。换言之，至少在这个领域内，我们对于每个此类提问所蕴涵的命题指派同样的意义表示，是理所当然的。采用这样的方法就可以保证我们的匹配方法对于回答“是”或“否”这样的提问仍然是适用的。

表达同样事情的输入具有同样的意义表示，这种思想称为“规范形式”（canonical form）理论。这样的方法大大简化了各种推理的工作，因为对于潜在的各种表达方式，系统只需处理一个意义表示即可。

当然，规范形式使语义分析的任务变复杂了。为了看到这一点，我们注意到上述的不同表达方式使用了完全不同的单词和句法，来说明素食品以及在什么饭店制作素食品。更具体地说，为了给所有这些提问指派同样的表示，我们的系统必须把 *vegetarian fare*，*vegetarian dishes* 和 *vegetarian food* 归入在这种上下文环境中的同一个事物，这里的 *having* 和 *serving* 的用法也是等价的，在这些提问中的句法剖析全都要与同样的意义表示相兼容。

给如此不同的输入指派相同的意义表示是太过于苛求了。幸好，在单词的意义之间以及在语法结构之间存在的某些系统性的意义关系，可以用来使这样的任务有线索可循。我们来研究这些例子中的单词*food*，*dish* 和*fare* 的意义。只要稍微想一想，或者查一查词典，就会发现这些单词都具有

许多不同用法。幸好，我们同时也可以看出，至少有一个意义是这些单词都共享的。如果有能力在不同用法中筛选出这个共享的意义，就可以把同样的意义表示指派给包括这些单词的短语。

一般来说，我们可以说这些单词都有不同的词义（word sense），并且某些词义和其他词义具有同义关系（synonymous）。在上下文中选出正确词义的过程称为词义消解（word sense disambiguation），或者与词类标注（part-of-speech tagging）相类比，称为词义标注（word sense tagging）。关于同义词、词义标注以及其他与单词意义有关的问题，将在第16章和第17章中讨论。这里只要知道这样的事实即可：尽管输入可以使用不同的单词，但是这并不妨碍给它们指派同样的意义。

不同单词的意义之间存在着系统的关系，句法分析在指派意义给句子时所起的作用也存在着相似的关系。具体地说，不同的句法分析经常会具有一些意义，如果它们不是完全相同的，至少也是彼此关联的。我们来研究下面的一对句子：

Maharani serves vegetarian dishes. （14.8）

Vegetarian dishes are served by Maharani. （14.9）

尽管在这两个句子中动词*serve*的论元位置各不相同，我们还是能在两个句子中给*Maharani*和*vegetarian dishes*指派相同的角色，因为我们具有关于主动句结构和被动句结构的相互关系的知识。具体地说，我们可以使用关于出现在这些结构中的语法主语和直接宾语的位置的知识，在这两个句子中，我们都可以给*Maharani*指派服务者的角色，给*vegetarian dishes*指派服务品的角色，尽管它们在表层中的出现位置各不相同。我们将在第15章中讨论语法在构造意义表示时的作用。

14.1.4 推论与变元

我们继续来研究意义表示的计算目的这个题目，现在来研究下面的更复杂的提问：

Can vegetarians eat at Maharani? （14.10）

如果在这里调用规范形式来强制系统给这个提问指派与前面那些例子同样的意义表示，则可能会犯错误。事实上，如果这个提问的结果得到与其他提问相同的回答，并不是因为它们意味着同样的事物，而是因为在素食者吃的东西和素食饭店提供服务的东西之间，存在着一种常识性的联系。这是一种关于世界的事实，而不是一种关于任何特定语言学规则的事实。这种情况说明，基于规范形式的方法和简单的匹配不能使我们对这个提问做出合适的回答。我们需要用系统的方法，把这个提问的意义表示和在知识库中表示的关于世界的事实彼此联系起来。

我们使用“推论”（inference）这个术语来一般地说明系统根据输入的意义表示以及存储的背景知识做出可靠结论的能力。推论必须能够使系统做出关于命题的真值的结论，而命题的真值并没有明显地表示在知识库中，而是必须从当前的命题中逻辑推导出来。

现在来研究下面的更复杂的提问：

I'd like to find a restaurant where I can get vegetarian food. （14.11）

与前面的例子不同，这个提问没有对任何特定的饭店进行推论。用户在这里说的是需要关于某个不知道的出售素食的无名饭店的实体的信息。由于这个提问没有提到任何特定的饭店，所以前面提倡的那种基于简单匹配的方法是行不通的。回答这样的提问要求进行更复杂的匹配，这种匹配需要使用变元（vaiable）。这种包含变元的表示可按如下形式来注释：

Serves (*x*, *VegetarianFood*)

只有当*x*能够被知识库中某个已知的客体来替换，使得整个命题得到匹配时，我们才说这个命题被成功地匹配了。变元替换之后就可以用来实现用户的提问了。当然，这个简单的例子只是说明

了可以使用这样的变元。语言输入可以包含不确定推论的各种类型的很多实例。关键的问题在于，任何的意义表示语言都能够处理这种类型的表示，现在只要了解这一点就足够了。

14.1.5　表达能力

最后，意义表示方法应该具有足够的表达能力（expressiveness）来处理领域广泛的各种事情。最理想的情况当然是只用一个单独的意义表示语言就可以恰当地表达任何可以感觉到的自然语言话段的意义。尽管对于任何一个表示系统来说，这样的期望也许太高了一些。14.3 节中将说明，一阶谓词演算具有足够的表达能力来处理很多意义表示中必须处理的问题。

14.2　语言的意义结构

前一节重点讨论了意义表示必须服务的目标，但没有过多地谈到称为“语言的意义结构”（meaning structure of language）的问题。我们已经知道了人类语言负载意义的各种方法。这些方法包括各种常规形式与意义联系、词序规则、时态系统、连接词和数量词、基本的谓词论元结构，等等。本节将重点讨论谓词论元结构。谓词论元结构这种机制对于意义表示语言的实质在实践上曾经有过巨大的影响。在第 15 章中，我们还要研究关于谓词论元结构的其他论题，但重点将不是意义表示的性质，而是怎样把意义表示组合起来。

14.2.1　谓词论元结构

看来人类的所有语言在它们的语义结构的核心部分都有一种谓词论元排列的形式。近似地说，这种谓词论元结构（predicate-argument structure）表示了隐藏在构成句子的单词和短语的成分的底层下的各个概念之间存在着的特定关系。这个底层结构在很大程度上能够从输入的各个部分的意义出发，构造出一个单独的组合性的意义表示。语法的最重要的任务之一是帮助组织这样的谓词论元结构。因此，我们的意义表示语言应该支持语言向我们表现出来的这种谓词论元结构。

第 9 章和第 11 章已经讨论过动词补语的初步概念。动词对于在句法结构中伴随它的数、语法范畴以及短语的位置都有特殊的约束。我们简短地回顾这种想法，研究下面的例子：

I want Italian food.　（14.12）

I want to spend less than five dollars.　（14.13）

I want it to be close by here.　（14.14）

这些例子分类之后，可以形成下面三个句法论元框架：

NP want NP

NP want Inf-VP

NP want NP Inf-VP

这些句法框架说明了将会伴随动词的论元的数量、位置和句法范畴。例如，例句（14.12）中出现动词 *want* 的框架说明了如下的事实：

- 这个谓词有两个论元
- 两个论元都是 *NP*
- 第一个论元在动词之前，起主语的作用
- 第二个论元在动词之后，起直接宾语的作用

我们在前面的章节已经说明过，这类事实对于捕捉动词的各种重要信息是非常有价值的。如果我们能分析与这些框架相关的那些容易观察到的语义信息，就能够获得关于意义表示的相当丰富的见识。我们开始时把这些框架从两方面扩充到语义领域：语义角色（semantic role）以及对于这些角色的语义限制（semantic restriction）。

语义角色的概念可以通过观察例句（14.12）到例句（14.14）中论元之间的相似性来理解。在每个场合，动词前的论元始终起着 *want* 行为的实体的作用，而动词后的论元则起着 *want* 的内容的作用。注意到这些规则并且相应地标注它们，就能够把动词的表层论元与在底层语义中的一套离散的角色联系起来。更加一般地说，我们可以说动词的次范畴化框架允许我们把表层结构中的论元与在这个输入的底层语义表示中论元所扮演的语义角色连接起来（linking）。把角色和特定的动词与动词的类别联系起来的这种研究，通常称为题元角色分析（thematic role analysis）或格角色分析（case role analysis），14.4 节和第 16 章将更详细地讨论这些问题。

语义限制的概念直接来自语义角色。再回到例句（14.12）到例句（14.14），可以看到并不是每个在动词前面的名词都能做“想的人”（wanter），只有某一类的概念或范畴才能够直截了当地充当“想的人”的作用。具体地说，动词 *want* 把作为第一个论元出现的成分限制为那些能够实际上进行 *want* 行为的人。在传统上讲，这样的概念称为“选择限制”（selectional restriction）。通过使用这种选择限制，动词就可以具体地说明对于它的论元的语义限制是什么。

在离开这个主题之前，我们还应该注意，动词并不是在语法中惟一具有谓词论元结构的东西。我们来研究来自 BERP 语料库中的例子：

an Italian restaurant under fifteen dollars （14.15）

在这个例子中，与介词 *under* 相关的意义表示可以具有如下结构：

Under (*ItalianRestaurant*, $15)

换言之，介词可以刻画为具有两个论元的谓词，其中第一个论元是对象，它与第二个论元处于某种关系中。

下面的例子表现了另一种非动词的谓词论元结构：

Make a reservation for this evening for a table for two persons at 8. （14.16）

在这个例子中，谓词论元结构建立在名词 *reservation* 的概念的基础上，而不是建立在短语中的主要动词 *make* 的基础上。这个例子的谓词论元结构有如下四个论元：

Reservation (*Hearer*, *Today*, 8*PM*, 2)

这样的讨论清楚地说明了，任何有用的意义表示语言必须能够支持语义的谓词论元结构的特征。具体地说，它必须包含对于语言所表示的语义信息的支持，主要有：

- 可变的谓词论元结构
- 论元对谓词的语义标记
- 对于过滤论元角色的语义约束说明

14.3 一阶谓词演算

一阶谓词演算（First Order Predicate Calculus，简称 FOPC）是一种灵活方便的、容易理解的、在计算上可循的方法，这种方法可以表示的知识能够满足 14.1 节和 14.2 节中对意义表示语言提出

的要求。具体地说，一阶谓词演算可以为意义表示的可能性验证、推论和表达能力等方面提供坚实的计算基础。FOPC最引人注目的特征是: 对于所表示的事物只做很少的说明。我们将会看到，FOPC所做的说明是相当容易理解的；它所表达的世界包括客体、客体的性质以及客体之间的关系。

在这一节中首先介绍FOPC的基本句法和语义，然后描述FOPC在一些与语言学相关的领域中的应用。然后，在 14.6 节中讨论 FOPC 之间的连接以及在前面的图 14.1 中介绍过的其他表示方法。

14.3.1　FOPC 基础

我们采用自底向上的方式来介绍FOPC。首先检查它的各种原子元素，然后说明把它们组合起来构成更大的意义表示。图14.2是用于描述FOPC的句法的一个完全的上下文无关语法，我们将使用这个语法并把它作为本节的路线图。

$$
\begin{aligned}
\textit{Formula} &\rightarrow \textit{AtomicFormula} \\
&\mid \textit{Formula Connective Formula} \\
&\mid \textit{Quantifier Variable}, \ldots \textit{Formula} \\
&\mid \neg \textit{Formula} \\
&\mid (\textit{Formula}) \\
\\
\textit{AtomicFormula} &\rightarrow \textit{Predicate}(\textit{Term}, \ldots) \\
\\
\textit{Term} &\rightarrow \textit{Function}(\textit{Term}, \ldots) \\
&\mid \textit{Constant} \\
&\mid \textit{Variable} \\
\\
\textit{Connective} &\rightarrow \wedge \mid \vee \mid \Rightarrow \\
\textit{Quantifier} &\rightarrow \forall \mid \exists \\
\textit{Constant} &\rightarrow A \mid \textit{VegetarianFood} \mid \textit{Maharani} \cdots \\
\textit{Variable} &\rightarrow x \mid y \mid \cdots \\
\textit{Predicate} &\rightarrow \textit{Serves} \mid \textit{Near} \mid \cdots \\
\textit{Function} &\rightarrow \textit{LocationOf} \mid \textit{CuisineOf} \mid \cdots
\end{aligned}
$$

图 14.2　用于说明一阶谓词演算表示句法的一个上下文无关语法。取自 Russell and Norvig(1995)

首先来检查“项”（term）的概念。项是 FOPC 用于表示客体的一种设置。从图 14.2 中可以看出，FOPC 提供了三种方法来表示这个基本的信息块：常量、函数和变量。每个这样的设置可以想像成一种命名方法，它们要给所研究的世界指定一个客体。

FOPC中的常量（constant）引用所描述的世界中的特定客体。按照惯例，常量通常用一个单独的大写字母来描述，如 *A* 和 *B* 等，也可以用一个单独的大写单词来描述，这常使我们想起诸如 *Maharani* 和 *Harry* 这样的专有名词。正如程序设计语言中的常量，FOPC的常量只严格地引用一个客体。但是，当存在若干个客体时，可以有多个常量引用它们。

FOPC中的函数（function）相当于在英语中经常表示为所属格（genitive）的概念，如 *location of Maharani* 或 *Maharani's location*。这样的表达式翻译成 FOPC 可以表示如下：

LocationOf (*Maharani*)

FOPC函数在句法上相当于一个单独的论元谓词。但是，重要的是应该记住，它们在外表上像谓词，事实上却只涉及一个单独客体的“项”。FOPC的函数为引用特定的客体提供了一种方便的途径。使用函数来引用客体时，不用与命名它的常量相联系。当存在像饭店这样的很多命名客体时，如果使用函数，只需一个像 location 这样的函数就可以与各种名字的饭店联系起来，非常方便。

在 FOPC 引用客体的机制中的最后一个概念是变量（ variable ）。变量一般用单个小写字母表示，使我们能够对客体做出判断，进行推论，而不必参照任何特定命名客体。变量的这种对没有名字的客体进行说明的能力有两个特色：一是能够对未知的匿名客体进行说明，二是能够对任意一个客体世界中的所有客体进行说明。在介绍了量词（ quantifier ）之后，还要回过头来再讨论变量的问题，FOCP 的量词将使变量更有用。

现在我们有了引用客体的方法，因此可以研究如何用 FOPC 来说明客体之间保持的关系。从 FOPC 的名称可以猜到，FOPC 是围绕谓词的概念组织起来的。谓词是一种符号，这种符号用于引用名称以及在给定领域内的一定数量的客体之间的关系。再回到14.1节中那个非形式化地介绍的例子，*Maharani serves vegetarian food* 可用 FOPC 的公式表示如下：

Serves (*Maharani*, *VegetarianFood*)

这个FOPC句子的意思是: *Serves*是一个具有两个位置的谓词，表示由常量*Maharani*和*VegetarianFood*标出的客体之间的关系。

谓词的另一种稍微不同的用法是对句子*Maharani is a restaurant*的下列表示，这是一种典型的表示方法：

Restaurant (*Maharani*)

这个例子中的谓词只有一个位置，它不涉及多个客体，而只是确认一个单独客体的某个性质。在这种情况下，谓词对*Maharani*的范畴所属关系进行编码。应该注意，这是一种处理范畴的普通方法，它的用处可能不太大。14.4 节将再次讨论范畴表示的问题。

由于具备了引用客体的能力、确认关于客体的事实的能力、把一些客体与另外的客体相互联系的能力，所以就能够构造出初步的组合表示。这些表示与图14.2中的原子公式的水平相对应。构造组合意义表示的这种能力是 14.2 节中描述的意义结构的核心组成部分。

这种组成复杂表示的能力不局限于使用单个谓词。更大的组合表示也可以通过使用逻辑连词（ logical connective ）的办法把不同的意义表示结合到一起。从图 14.2 中可以看出，逻辑连词使用三个运算符中的一个把逻辑公式连接起来，这样就使我们有了构造更大的意义表示的能力。例如，我们来研究如下的来自 BERP 语料库的例子以及这个句子的一个可能的意义表示：

I only have five dollars and I don't have a lot of time. (14.17)

$$Have(Speaker, FiveDollars) \wedge \neg Have(Speaker, LotOfTime)$$

这个例子的语义表示是通过使用运算符∧和¬，直接从个别从句的语义构造而成的。注意，由于图14.2中的语法具有递归性，可以使用这些逻辑连词构造出无限数目的逻辑公式。因此，使用这样的句法，就有能力使用有限的手段构造出无限数目的意义表示。

14.3.2 FOPC 的语义

在FOPC知识库中的各种客体、性质以及关系，借助于它们与这个知识库所模拟的外界世界中

的客体、性质和关系而获得意义。因此，FOPC 的句子可以根据它们所编码的命题是否与外界世界相符而被指派“真”（True）或“假”（False）值。

我们来研究下面的例子：

Ay Caramba is near ICSI.　（14.18）

捕捉在 FOPC 中这个例子的意义包括辨认与句子中的各种语法成分相对应的“项”和“谓词”，并构造逻辑公式，用以表达那些在这个句子的单词和句法中所蕴涵的关系。对于这个例子来说，通过这些工作可以得到如下的结果：

$$Near(LocationOf(AyCaramba), LocationOf(ICSI))$$

这个逻辑公式的意义可以根据*LocationOf* (*AyCaramba*)和*LocationOf* (ICSI)两项之间的关系、谓词 *Near* 以及在它们所模拟的世界中相应的客体和关系等而获得。具体地说，这个句子可以根据在现实世界中 Ay Caramba 是不是真正与 ICSI 离得近而被指派 True 或 False 值。当然，由于我们的计算机很少直接访问外部世界，所以只好依靠某些其他手段来决定这种公式的真值。

为了达到目的，我们将采用所谓“数据库语义学”来确定逻辑公式的真值。从操作性的角度看，对于原子公式，如果它们从字面上在知识库中表现出来，或者可以从知识库中的其他公式推论出来，就认为这个原子公式为真。对于包含逻辑连词的公式，可以把公式中的成分的意义与它们包含的逻辑连词的意义结合起来，从而解释整个公式的意义。图 14.3 是对图 14.2 中的每个逻辑运算符的具体解释。

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$
False	*False*	*True*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*	*True*
True	*False*	*False*	*False*	*True*	*False*
True	*True*	*False*	*True*	*True*	*True*

图 14.3　表示各种逻辑连词的意义的真值表

运算符∧（and，和）与¬（not，非）的语义是很直接的，并且至少与它们在英语中相应的意义有联系。但是应该指出，∨（or，或）这个运算符在某种情况下并不与英语中的“或”这个析取（disjunctive）意义相对应，⇒（imply，蕴涵）这个运算符与英语中的“蕴涵”或“因果”的常识概念只有很松弛的联系。14.4 节将会更详细地讨论到，在大多数情况下，直接根据真值表的值来判断，与顾名思义地根据直觉来判断要可靠得多。

14.3.3　变量和逻辑量词

现在我们已经有了必要的意义表示机制来进一步讨论前面说过的变量。前面曾经指出，在 FOPC 中变量有两种用法。一种用法是引用特定的匿名客体，一种用法是一般性地引用在一个集合中的全部客体。这两种用法都可以通过使用称为“量词”（quantifier）的运算符来实现。作为 FOPC 基础的这两个量词运算符，一个是存在量词（existential quantifier），记为∃，读为“there exists”（存在），一个是全称量词（universal quantifier），记为∀，读为“for all”（对于一切的）。

需要使用存在量词的变量在英语中通常表现为一个不确定的名词短语。我们来研究下面的例子：

a restaurant that serves Mexican food near ICSI.　（14.19）

这个名词短语的参照是具有特定性质的一个特定范畴的某个匿名客体。下面是这个短语的一个合理的意义表示：

$$\exists x Restaurant(x) \wedge Serves(x, MexicanFood) \wedge Near((LocationOf(x), LocationOf(ICSI)))$$

在这个句子的开头的存在量词告诉我们如何在这个句子的上下文中解释变量x。不很形式地说，应该至少有一个客体，如果用它来替换变量x，结果能够使形成的句子为真。例如，如果*AyCaramba*是在ICSI附近的墨西哥饭店，那么用*AyCaramba*替换x，可以得到如下的逻辑公式：

$$Restaurant(AyCaramba) \wedge Serves(AyCaramba, MexicanFood) \wedge Near((LocationOf(AyCaramba), LocationOf(ICSI)))$$

根据运算符∧的语义，如果其中的三个成分的原子公式都为真，那么这个句子就为真。如果它们都在系统的知识库中存在，或者它们可以从知识库中的其他事实推论出来，那么这三个原子公式就为真。

全称量词的用法也可以通过用已知的客体来替换变量的办法得到解释。全称量词的替换语义完全是从字面上来解释“for all”（对于一切）这种表达；运算符∀的用法是：用知识库中的任何客体来替换全称量词变量，结果都得到真的公式，那么这个逻辑公式就为真。而运算符∃与此形成鲜明的对照，为了使句子为真，它只需要一个单独的可靠替换。

我们来研究下面的例子：

All vegetarian restaurants serve vegetarian food. （14.20）

这个句子的一个合理表示如下：

$$\forall x VegetarianRestaurant(x) \Rightarrow Serves(x, VegetarianFood)$$

如果用已知客体来替换x，使得每个句子为真，那么整个句子就为真。我们可以把所有可能替换的集合分成两个集合：一个是包含素食饭店的客体的集合，一个是包含其他饭店的集合。首先来研究替换的客体在事实上全都是素食饭店的情况；这样的替换可得到如下的句子：

$$VegetarianRestaurant(Maharani) \Rightarrow Serves(Maharani, VegetarianFood)$$

如果假定我们已经知道表示结果的从句

$$Serves(Maharani, VegetarianFood)$$

为真，那么整个句子必定为真。由于前提（antecedent）和结果（consequent）都为真，根据图14.3中前两列都为真，那么这个蕴涵句子本身的值为真。显而易见，这个结果对于用x来代表素食饭店的“项”的所有替换都是一样的。

要注意的是，为了让这个句子为真，所有可能的替换都必须为真。如果我们从非素食饭店客体的集合中选出一个来进行替换，将会发生什么情况呢？我们研究把*Ay Caramba*作为非素食饭店来替换x的情况：

$$VegetarianRestaurant(AyCaramba) \Rightarrow Serves(AyCaramba, VegetarianFood)$$

因为这个蕴涵的前提为假，根据图14.3可以确认，这个句子总是为真，同样也满足∀的约束。

注意，也可能存在这样的情况，*Ay Caramba* 也出售素食，但它不一定是一个素食饭店。同时还要注意，尽管在我们选择的例子中，对于用这样的推理来替换 x 的客体并没有范畴的限制。换句话说，对于 x 是否是饭店或者与饭店有关的概念，是不存在限制的。我们来研究下面的替换：

$$VegetarianRestaurant(Carburetor) \Rightarrow Serves(Carburetor, VegetarianFood)$$

这时，前提仍然为假，在这种无关替换的情况下规则仍然为真。

总之，在逻辑公式中的变量必须用存在量词（∃）或全称量词（∀）来限定。为了满足存在量词限定的变量，必须至少有一个替换形成一个为真的句子。对于全称量词限定的变量，句子必须在所有可能的替换时都为真。

14.3.4　推论

在 14.1 节中对于意义表示语言的一个最重要的要求是：意义表示语言必须支持推论（inference）。推论能够给知识库增加可靠的新命题，或者能够确定那些没有明确包含在知识库中的命题的真值。本节简短地讨论取式推理（modus ponens），这是FOPC提供的最重要的推论方法。取式推理的应用将在第 18 章中讨论。

取式推理的推论形式是我们很熟悉的，它相当于非形式化的 *if-then* 推理。我们可以随意地把取式推理定义如下，其中 α 和 β 是 FOPC 的公式：

$$\frac{\alpha \quad \alpha \Rightarrow \beta}{\beta}$$

一般来说，在这样的图示中，横线下面的公式是从横线上面的公式通过某种形式的推论推导出来的。取式推理简单地说明，如果蕴涵规则左手边在知识库中出现，就可以推论出这个规则的右手边。在下面的讨论中，我们把蕴涵的左手边作为前提（antecedent），把蕴涵的右手边作为结论（consequent）。

作为取式推理的典型使用的一个例子，我们来研究下面的例子。这个例子使用了上一节的规则。

$$\frac{VegetarianRestaurant(Rudys) \quad \forall x\, VegetarianRestaurant(x) \Rightarrow Serves(x, VegetarianFood)}{Serves(Rudys, VegetarianFood)} \qquad (14.21)$$

这里，公式 *VegetarianRestaurant* (*Rudys*)与规则的前提相匹配，这样就可以使用取式推理，得出 *Serves* (*Rudys*, *VegetarianFood*)这样的结论。

取式推理的典型实际应用有两种方式：向前链接（forward chaining）和向后链接（backward chaining）。在向前链接系统中，取式推理按照刚才所描述的方法来使用。当一个单独的事实加到知识库中的时候，取式推理用这种事实来激发所有可应用的蕴涵规则。在这种类型的安排中，每当一个新的事实被加到知识库中时，就可以找到并应用所有可应用的蕴涵规则，每个结果都把新的事实加到知识库中，依次使用知识库中的这些新命题去激发那些可应用于它们的蕴涵规则。这个过程继续进行到没有新的事实可以被推导出来为止。

向前链接方法的优点是，当需要的时候，有关的事实必须在知识库中表现出来，因为在向前链接中所有的推论都必须事先进行。这样可以充分地减少回答下一个问题时所需的时间，因为这时只需要进行简单的查询。这种方法的缺点是：所引用或存储的事实可能是以后永远用不上的。产生式系统（production system）大量地使用认知模型的研究成果，通过增加控制知识的方法来决定所要激发的规则，从而提升了向前链接系统。

在向后链接中，取式推理按相反的方向进行调用提问，以证明特定的命题。第一步根据提问是否存储在知识库中来判定提问公式是否为真。如果提问不在知识库中，那么下一步就搜索在知识库中有没有可应用的蕴涵规则。如果一条规则的结果部分与提问公式相匹配，那么这条规则就是可应用的规则。如果存在任何这样的规则，并且规则的前提为真，那么提问就被证明了。如果把前提作为一个新的提问，那么递归地进行向后链接也就不足为奇了。Prolog程序语言就是实现这种策略的一个向后链接系统。

现在我们来看一看这是怎样实现的。假定给定例句（14.21）中直线上方的事实，让我们来证实命题 *Serves* (*Rudys*, *VegetarianFood*)为真。由于这个命题在知识库中不存在，因此需要开始搜索一个可应用的规则，使得我们能够得到上面给定的规则。用常量替换变量 x 之后，下一个任务就是证明规则的前提 *VegetarianRestaurant* (*Rudys*)，这当然是我们给定的事实之一。

注意，重要的是应该区分从提问到已知事实的向后链接的推理和从已知结果到未知前提的向后推理。具体地说，所谓“向后推理”是指如果一个规则的结果为真，就假定其前提也为真。例如，假定我们知道了 *Serves* (*Rudys*, *VegetarianFood*)为真，由于这个事实与规则中的结果相匹配，就可以使用向后推理得出 *VegetarianRestaurant* (*Rudys*)的结论。

向后链接是一种可靠的推理方法，而向后推理则是不可靠的，尽管向后推理是一种经常使用的似是而非的推理形式。从结果推导出前提的这种似是而非的推理称为“诱导法”或“溯因推理”（abduction）。在第 18 章中将会看到，当人们在分析扩展的话语时，在很多推论的计算中，诱导法还是经常被使用的。

向前链接和向后链接两者都是可靠的，但它们又都是不完全的。这意味着还存在着一些可靠的推论，不可能通过单独使用向前链接或向后链接的方法的系统来发现。幸运的是，有一种称为“归结法”（resolution）的交替界面技术是可靠而且完全的，但这种基于归结法的推论系统的计算代价比向前链接和向后链接系统高得多。因此，在实际应用中大多数系统还是采用某种链接推理的形式，把系统的负担放到知识库的开发方面，使我们能够进行必要的推理。

14.4 某些与语言学相关的概念

我们的整个生命都用在研究如何表示人类知识的各个方面的问题上。这样的努力范围极为广泛，从表示个别的诸如时间这样的非常专门的领域，到给关于周围世界的常识进行编码，都是我们研究的领域(Lenat and Guha, 1991)。但是，这里研究的重点比较适中，并没有那么广泛。本节简要地回顾了与语言的计算机处理有关的一些重要问题。具体地说，本节将介绍范畴、事件、时间和信念的意义表示。

14.4.1 范畴

在14.2节中可以注意到，具有谓词语义的单词，其论元经常以选择限制形式表现出优先性。这些选择限制的典型表示采用的是基于语义的范畴，其中一个范畴的所有成员共享一套相关的特征。

表示范畴的最普通的方法是为每个范畴造出一个一元谓词。这样的谓词可以对每个有关的范畴进行说明。例如，在关于饭店的讨论中就使用了一元谓词 VegetarianRestaurant：

VegetarianRestaurant (*Maharani*)

对于每个已知的素食饭店，在知识库中都有一个相似逻辑公式。

遗憾的是，在这个方法中范畴表示的是关系，而不是实实在在的客体。因此，这只能对于构成这个关系的各个成分有所说明，而很难对于范畴本身有所说明。例如，如果想把一个给定范畴的最普通的成员表示如下：

MostPopular (*Maharani*, *VegetarianRestaurant*)

这就不是一个合格的 FOPC 公式，因为在 FOPC 中的谓词必须是“项”，而不能是其他谓词。

解决这个问题的一个办法是使用一种称为“具体化”（reification）的技术，把我们想表述的所有概念都表示为实实在在的客体。在这种情况下，我们就可以把*VegetarianRestaurant*这个范畴表示为诸如*Maharani*这样的实在客体。这样，所属性概念这样的范畴就可以通过所属性关系表示如下：

ISA (*Maharani*, *VegetarianRestaurant*)

这个记为*ISA* (is a)的关系在客体和以客体为成员的范畴之间是成立的。这样的技术也可以通过使用其他相似关系的办法加以扩充，使它能表达范畴的层次。例如：

AKO (*VegetarianRestaurant*, *Restaurant*)

这里，关系 *AKO* (a kind of)在范畴之间成立，并且说明范畴的包含关系。当然，为了真正给出这些谓词的意义，应该把范畴定义为集合，并把这些谓词放到更大的集合中。

第 16 章将讨论这些关系在词汇关系数据库、选择限制表示以及单词意义的歧义消解中的实际应用。

14.4.2　事件

迄今为止，使用的事件（event）表示包括一个单独的谓词以及与给定的例子相联系的角色所需的多个论元。例如，在 14.2 节中讨论过的 *making a reservation* 事件的表示包括一个单独的谓词以及人在预定时所需的论元，如饭店、日期、时间、参加人数等，如下所示：

Reservation (*Hearer*, *Maharani*, *Today*, *8PM*, 2)

在动词的场合，这种方法简单地假定，表示动词意义的谓词的论元数目与该动词在它的次范畴化框架中所表现出来的论元数目是相同的。

遗憾的是，这种方法有四个问题使得它在实际应用中显得捉襟见肘：

- 如何对于一个给定的事件确定它的正确角色数目
- 如何表示关于与一个事件相联系的角色的事实
- 如何保证能够直接从一个事件的表示引导出所有正确推论
- 如何保证从一个事件的表示引导出所有推论没有不正确的

考虑一系列的事件表示，我们将探索这些问题以及其他相关问题。这个讨论将集中研究动词 eat 的如下例句：

I ate.　　(14.22)

I ate a turkey sandwich.　　(14.23)

I ate a turkey sandwich at my desk.　　(14.24)

I ate at my desk.　　(14.25)

I ate lunch.　　(14.26)

I ate a turkey sandwich for lunch. (14.27)

I ate a turkey sandwich for lunch at my desk. (14.28)

显而易见，在这些例句中，诸如动词*eat*这样的谓词的论元数目是可变的，这就给我们提出了一个非常棘手的问题。我们原来曾经这样想像：由于所有这些例子都表示同一类事件，因此，在FOPC中的谓词应该有固定的相关成分，所以它们应该有固定数目的论元。

根据这些例子，我们提出的一个可能的解决办法是从句法上来处理。在第11章中给出的解决办法是为动词所允许的每种论元的格式建立一个次范畴化框架。把这种办法类推到语义方面，就是为eating建立不同的谓词，用来处理动词*eat*的各种可能的行为方式。用这样的方法可以把例句（14.22）到例句（14.28）表示如下：

$Eating_1(Speaker)$
$Eating_2(Speaker, TurkeySandwich)$
$Eating_3(Speaker, TurkeySandwich, Desk)$
$Eating_4(Speaker, Desk)$
$Eating_5(Speaker, Lunch)$
$Eating_6(Speaker, TurkeySandwich, Lunch)$
$Eating_7(Speaker, TurkeySandwich, Lunch, Desk)$

在为每个次范畴化框架建立不同的谓词时，这种方法巧妙地回避了谓词*Eating*究竟有多少个论元的问题。遗憾的是，这种方法的代价太高了。除了谓词的名字能够给出某些暗示之外，尽管在这些事件之间在逻辑上存在着明显的关系，但是这种方法并不能提供任何关于事件之间的这种关系。具体地说，如果例句（14.28）为真，则其他例子也为真。类似地，如果例句（14.27）为真，则例句（14.22）、例句（14.23）和例句（14.26）也为真。这样的逻辑联系不能根据这些谓词单独得出。此外，我们还希望建立一个常识知识库，把概念*Eating*与*Hunger*和*Food*这些相关概念也包含进来。

解决这些问题的一个办法是使用所谓的“意义假设”（meaning postulate）。我们来研究下面关于意义假设的例子：

$$\forall w,x,y,z\ Eating_7(w,x,y,z) \Rightarrow Eating_6(w,x,y)$$

这个意义假设把谓词中的两个语义联系在一起了。建立其他意义假设可以用来处理不同的Eating的其他逻辑关系，并且把它们与相关的观念联系起来。

尽管这个方法在小的领域中行得通，但是还明显地存在规模设定性（scalability）问题。更敏感的办法是，从例句（14.22）到例句（14.28）全都涉及同样的谓词，只是某些论元在表层形式中消失了。使用这种方法时，很多论元都被包含在谓词的定义中，就像它们在输入中出现时那样。例如，诸如$Eating_7$这样的谓词结构给我们的谓词是含有四个论元的，它们是：吃的人、吃的东西、吃的哪一顿饭以及吃的地点。下面的公式表现了例子的语义：

$\exists w,x,y\ Eating(Speaker,w,x,y)$
$\exists w,x\ Eating(Speaker,TurkeySandwich,w,x)$
$\exists w\ Eating(Speaker,TurkeySandwich,w,Desk)$
$\exists w,x\ Eating(Speaker,w,x,Desk)$
$\exists w,x\ Eating(Speaker,w,Lunch,x)$
$\exists w\ Eating(Speaker,TurkeySandwich,Lunch,w)$
$Eating(Speaker,TurkeySandwich,Lunch,Desk)$

这个方法直接表示出这些公式之间的逻辑联系。具体地说，所有带有论元项的句子在逻辑上都包含了公式的真值，而这些公式是以存在量词的约束变量作为论元的。

遗憾的是，这种方法至少有两个明显的不足：第一，这种方法的负担太重；第二，这种方法使我们不能把事件个性化。为了说明这种方法为什么负担过重，我们来研究例句（14.26）到例句（14.28）中关于*for lunch*这个补语的处理方式，这种方法把*for lunch*作为第三个论元，即“吃的哪一顿饭”，加到谓词*Eating*中。这样的表示方法使得我们对于任何*Eating*事件都必须和“吃的哪一顿饭”联系起来，也就是说，凡是*Eating*事件，都必须说明这是早饭、午饭还是晚饭。更加具体地说，在上面的例子中，关于吃饭（Eating）的论元的存在量词变量必须在形式上都和“吃的哪一顿饭”联系起来。这种做法显然是愚蠢的，因为人们在吃东西时不一定都要说明这是哪一顿饭，人们也可以在早饭、中饭和晚饭的时间之外进食。

为了证明这种方法不适合处理个性化的事件，我们来研究下面的公式：

$$\exists w,x\ Eating(Speaker,w,x,Desk)$$
$$\exists w,x\ Eating(Speaker,w,Lunch,x)$$
$$\exists w,x\ Eating(Speaker,w,Lunch,Desk)$$

如果我们知道前面两个公式描述的是同一事件，就可以把它们结合起来造出第三个表示公式。遗憾的是，使用当前的表示公式，我们不能说出这样做是否有可能。*I ate at my desk* 和 *I ate lunch* 这两个独立的事实不允许我们得出 *I ate lunch at my desk* 的结论。显而易见，这时我们还缺少如何引用 *I ate lunch at my desk* 这个事件的办法。

我们可以使用范畴来解决这个问题，这时应用具体化（reification）的办法来提升事件，使事件成为能够量词化的客体，并且能够通过定义好的关系与其他客体联系起来(Davidson, 1967; Parsons, 1990)。使用这样的方法，我们来研究例句（14.23）的表示。

$$\exists w\ ISA(w,Eating)$$
$$\wedge Eater(w,Speaker)\wedge Eaten(w,TurkeySandwich)$$

这样的表示的意思是：存在着一个吃饭的事件，其中 *Speaker* 是吃饭这个事件的行为者，*TurkeySandwich* 是被吃的东西。使用相似的方法，我们可以得出例句（14.22）和例句（14.27）的意义表示：

$$\exists w\ ISA(w,Eating)\wedge Eater(w,Speaker)$$
$$\exists w\ ISA(w,Eating)$$
$$\wedge Eater(w,Speaker)\wedge Eaten(w,TurkeySandwich)$$
$$\wedge MealEaten(w,Lunch)$$

这种事件具体化（reified-event）的方法有如下特点：

- 对于一个给定的表层谓词，不需要说明量词的确定数目，在输入中出现多少角色和填充项都可以结合到表层谓词中。
- 只要在输入中提到角色，不需要再对角色进行意义假设。
- 在有密切联系的例子之间，只要使用逻辑连接就可以满足把它们连接起来的要求，不再需要意义假设。

14.4.3　时间表示

在前面对于事件的讨论中，当所表示的事件发生时没有讨论关于时间表示的问题。如何在一个可用的形式中表示时间信息，属于时间逻辑（temporal logic）的研究领域。这里将介绍时间逻辑的

最基本概念，简单地讨论人类语言传达时间信息的手段，此外还将介绍时态逻辑（tense logic），这种逻辑是动词时态传达时间信息的一种手段。

关于时间的最简单理论认为，时间是一直向前流动的，事件与时间线（timeline）上的一个点或者一个片断相联系。根据这样的概念，可以把不同的事件放在这个时间线上，从而形成事件的顺序。更具体地说，如果时间流把第一个事件引导到第二个事件，我们就说第一个事件先于（precede）第二个事件。在大多数理论中，与这些概念有关的还有时间的当前时刻概念。把这些概念与时间顺序的概念结合起来，就产生了我们所熟知的关于现在、过去和将来的概念。

现在已经有很多表示这种时间信息的方法，这是不足为奇的。其中一种相当简单的时间表示法是在我们刚才说的具体化事件的 FOPC 框架内进行的。我们来研究下面的例子：

I arrived in New York. (14.29)

I am arriving in New York. (14.30)

I will arrive in New York. (14.31)

这些句子讲的都是同一种类型的事件，它们的区别仅仅在于动词的时态不同。按照现在的事件表示方法，这三个句子都共享如下表示，这个表示中没有时间信息：

$$\exists w\ ISA(w, Arriving) \wedge Arriver(w, Speaker) \wedge Destination(w, NewYork)$$

由动词时态提供的关于时间的信息，可以通过给事件变量w的谓词附加信息的方法来表达。具体地说，我们可以增加时间变量来表示相应于事件的时间段、事件的终点以及由动词时态说明的关于这个终点到当前时间的时间谓词。使用这样的方法可以得到关于 *arriving* 的下列意义表示：

$$\exists i, e, w, t\ ISA(w, Arriving) \wedge Arriver(w, Speaker) \wedge Destination(w, NewYork) \wedge IntervalOf(w, i) \wedge EndPoint(i, e) \wedge Precedes(e, Now)$$

$$\exists i, e, w, t\ ISA(w, Arriving) \wedge Arriver(w, Speaker) \wedge Destination(w, NewYork) \wedge IntervalOf(w, i) \wedge MemberOf(i, Now)$$

$$\exists i, e, w, t\ ISA(w, Arriving) \wedge Arriver(w, Speaker) \wedge Destination(w, NewYork) \wedge IntervalOf(w, i) \wedge EndPoint(i, e) \wedge Precedes(Now, e)$$

这个意义表示引入了一个表示与这个事件相关的时间段的变量以及一个表示这个时间段终点的变量。二元谓词 *Precedes* 表示第一个时间点论元前于第二个时间点论元；常量 *Now* 表示当前时间。对于过去的事件，时间段的终点必须前于当前时间。类似地，对于将来的事件，当前时间必须前于事件的终点。对于发生在现在的事件，当前时间包含在事件的时间段内。

遗憾的是，简单动词的时态和时间点之间的关系并不是直截了当的。我们来研究下面的例子：

Ok, we fly from San Francisco to Boston at 10. (14.32)

Flight 1390 will be at the gate an hour now. (14.33)

在第一个例子中，动词 *fly* 的现在时态用于说明一个将来的事件，在第二个例子中，动词的将来时态用于说明一个过去的事件。

如果考虑到其他一些动词时态，情况就更复杂了。我们来研究下面的例子：

Flight 1902 arrived late.　（14.34）

Flight 1902 had arrived late.　（14.35）

尽管这两个句子都是讲的过去的事件，如果用相同的方式来表示似乎不对。第二个例子似乎在背景上隐藏着另一个没有名字的事件（例如，当 1902 航班晚点到达时，某件其他事情已经发生过了）。为了处理这些语言现象，Reichenbach(1947)提出参照点（reference point）的概念。在我们简单的时间处理方法中，时间流中的当前时刻等于说话的时间，并且把它作为事件发生时的参照点（在之前，在当时，在之后）。在Reichenbach的方法中，参照点的概念是与发话时间（utterance time）和事件时间（event time）分开的。下面的例子可以说明这种方法的基本内容：

When Mary's flight departed, I ate lunch.　（14.36）

When Mary's flight departed, I had eaten lunch.　（14.37）

在这两个例子中，“吃饭”这个事件发生在过去，也就是说，发生在发话之前。在第一个例子中的动词时态说明，当吃饭事件开始时飞机就起飞了；而在第二个例子中，吃饭事件发生在飞机起飞之前。这样，在 Reichenbach 的术语中，“起飞”这个事件是参照点。考虑了“吃饭”事件和“起飞”事件有关的附加约束，就可以确认这些事实。在第一个例子中，参照点和吃饭事件发生在同一时间；而在第二个例子中，吃饭事件发生在参照点之前。图 14.4 用英语的简单时态来说明 Reichenbach 的方法。请读者用 FOPC 来表示例句（14.36）和例句（14.37）。

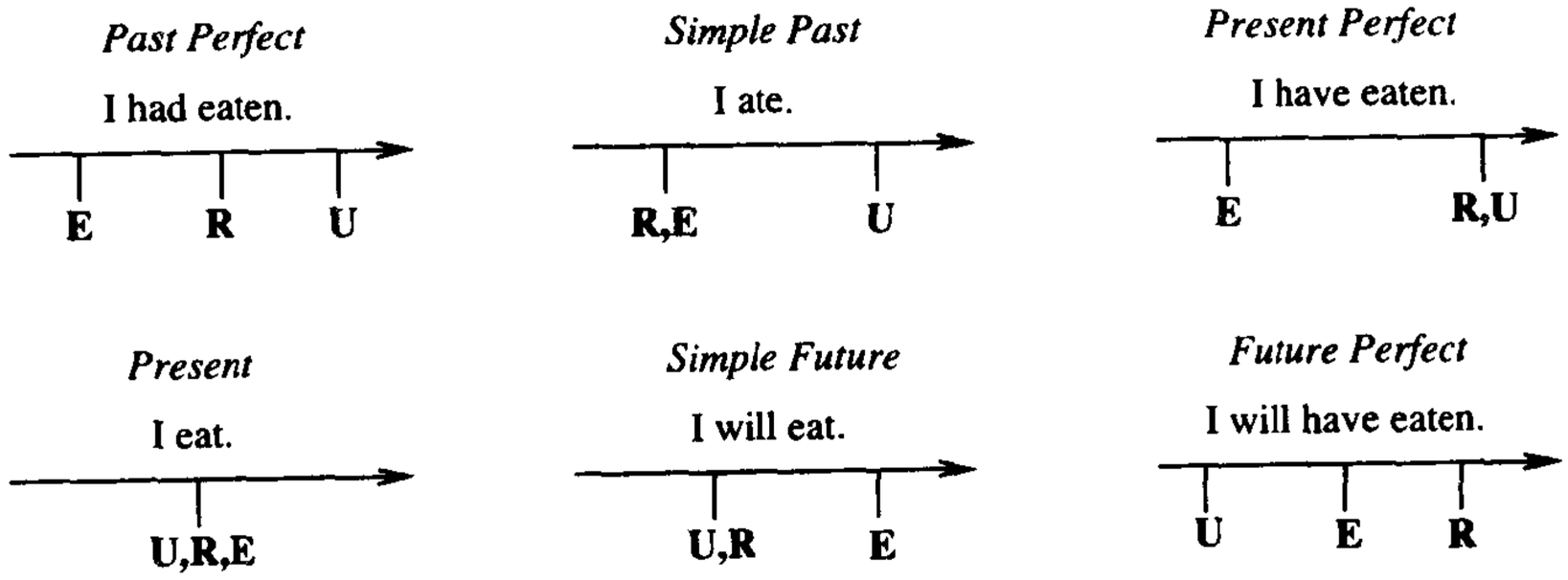


图 14.4　把Reichenbach的方法应用于英语时态。在这个图中，时间流是从左到右的，**E**表示事件的时间，**R**表示参照时间，**U**表示发话时间

这样的讨论只局限在过去、现在和将来这些比较广泛的概念，以及怎样把它们用动词时态表现出来。当然还有很多其他更直接和更专门的方式来表现时间信息，包括使用各种时间表达方式，下面是 ATIS 中的一些例子：

I'd like to go at 6：45, in the morning.　（14.38）

Somewhere around noon, please.　（14.39）

Later in the afternoon, near 6PM.　（14.40）

我们在下面的章节中可以看到，在信息抽取和问答系统等实际应用中，这种时间表达的语法是非常重要的。

最后，我们还要注意，这些例子还反映了系统化的概念组织结构。具体地说，英语中的时间表达常常反映在空间方面，正如在这些例子中*at*，*in*，*somewhere*和*near*的各种用法(Lakoff and Johnson,

1980; Jackendoff, 1983)。在这些例子中，时间和空间存在着比喻性的概念组织结构，其中一个领域可以系统地用另一个领域来表示，这些问题将在第 16 章中详细讨论。

14.4.4 体

上一节讨论了参照发话时间来表示事件时间的方法。这里将讨论“体”（aspect）的概念。“体”涉及到相关话题的一个聚类，包括一个事件是否结束，一个事件是否进行，一个事件是在一个时间点上还是在一个时间段上表达概念，由于一个事件是否会有世界上的某个特定事件跟随而来。根据这些以及其他相关概念，在传统方式上，事件的表示可以分为四个体：

静态体（Stative）：I know my departure gate.
行动体（Activity）：John is flying.
完成体（Accomplishment）：Sally booked her flight.
达成体（Achievement）：She found her gate.

Aristode在早期讨论过这样的分类，但这里的分类来自 Vendler(1967)。在下面的讨论中将简要描述这四种体，同时也介绍 Dowty(1979)提出的某些诊断技术，以便辨别例子应该属于哪一类。

静态体表示（Stative expression）表达了在一个给定时间点的事件，事件的参与者具有特定的性质或处于一个状态中。因此，可以把静态体表示想像成在一个单独的时间点上捕捉了世界的某个方面。我们来研究下面来自 ATIS 的例子：

I like Flight 840 arriving at 10：06. (14.41)

I need the cheapest fare. (14.42)

I have a round trip ticket for $662. (14.43)

I want to go first class. (14.44)

在这些例子中，事件的参与者作为句子的主语，在一个特定的时间点上对于某个事情有所经验，无论这个经验者前面是否处于相同的状态，或者在将来是否想做什么，这些在句子中都没有说明。

为了辨别这种静态体表示，提出了一些诊断测试方法。例如，如果静态体动词使用进行时形式，会明显地感到句子很古怪。

*I am needing the cheapest fare on this day. (14.45)

*I am wanting to go first class. (14.46)

要注意，在这里以及以后的例子中，我们使用“*”号来标志那些非良构（ill-formedness）的概念，既包括句法因素，也包括语义因素。

当使用命令式时，静态体表示也会使人感到古怪：

*Need the cheapest fare! (14.47)

最后，静态体表示不容易使用 *deliberately*（故意地）和 *carefully*（小心地）等动词来修饰。

*I deliberately like Flight 840 arriving at 10：06. (14.48)

*I carefully like Flight 840 arriving at 10：06. (14.49)

在行动体表示（activity expression）描述的事件中，参与者所参与事件的行为没有特定的结束点。与静态体不同，行动体的行动是发生在时间的某一个跨段上的，因此它与一个单独的时间是没有联系的。我们来考虑下面的例子：

She drove a Mazda.　(14.50)

I live in Brooklyn.　(14.51)

这两个例子中的主语从事于或从事过某种行动，而这种行动是用在时间的某个期间之内的动词来说明的。

与静态体不同，动作体表示可以很好地使用进行时形式和命令式形式。

She is living in Brooklyn.　(14.52)

Drive a Mazda!　(14.53)

但是，与静态体表示一样，行动体表示如果使用带 *in* 的时间短语来进行时间方面的修饰，会使句子变得古怪。

*I live in Brooklyn in a month.　(14.54)

*She drove a Mazda in an hour.　(14.55)

但是，这些句子可以成功地使用带 *for* 的时间状语来修饰。如下面的例子所示：

I live in Brooklyn for a month.　(14.56)

She drove a Mazda for an hour.　(14.57)

与行动体表示不同，完成体表示（Accomplishment Expression）描写的事件有一个自然的结束点，并且结果会形成一个特定的状态。我们来考虑下面的例子：

He booked me a reservation.　(14.58)

United flew me to New York.　(14.59)

在这些例子中，事件可以看成是发生在时间的某个期间内，当所希望的状态完成时，事件就结束了。

有一些诊断方法可以用来区分完成体事件和行动体事件。我们来研究下面的例子，使用动词 *stop* 进行测试：

I stopped living in Brooklyn.　(14.60)

She stopped booking my flight.　(14.61)

第一个例子表示一个行动，我们可以很确定地得出结论说：尽管 *I lived in Brooklyn* 这个命题是一个行动，但这个行动已经结束了。然而，对于第二个例子，我们并不能得出结论说 *She booked my flight* 是一个成立的命题，因为行动在所希望的状态完成之前已经结束了。因此，尽管停止一个行动可以推演出这个行动曾经发生过，但停止一个完成的事件就说明这个事件没有成功。

行动体和完成体也可以根据它们能否用各种时间状语来修饰而加以区别。考虑下面的例子：

*I lived in Brooklyn in a year.　(14.62)

She booked a flight in a minute.　(14.63)

一般来说，完成体可以用带 *in* 的时间表达式来修饰，而简单的行动体不行。

最后一种体是达成体表示（achievement expression）。达成体和完成体都以一个状态作为结束，这是它们的相似之处。考虑下面的例子：

She found her gate.　(14.64)

I reached New York. (14.65)

与完成体不同，达成体事件可以想像成是立即发生的，它不等同于任何导致某种状态的特定行动。更具体地说，在这些例子中的事件可能前面会有 *searching* 或 *traveling* 这样的事件，当前事件是这样的事件的扩展；直接相应于 *found* 和 *reach* 的事件被设想为一个时间点，而不是一个时间段。

这种事件具有的时间点的特性含蓄地表现了它们被时间状语修饰的情况。我们来具体考虑如下的例子：

I lived in New York for a year. (14.66)

*I reached New York for a few minutes. (14.67)

与行动体表示和完成体表示不同，达成体表示不能用带 *for* 的状语来修饰。

如前面所述，达成体表示还可以使用单词 *stop* 与完成体表示区分开来。考虑下面的例句：

I stopped booking my flight. (14.68)

*I stopped reaching New York. (14.69)

前面已经看到，完成体表示如果使用 *stop*，会导致不能达到预想状态的结果。但是要注意，这样所得到的表示仍然是良构的（well-formed）。另一方面，如果达成体表示使用 *stop*，所得到的表示则是不可接受的。

还要注意，因为完成体表示和达成体表示的结果都能形成一个状态，有时可以把它们结合起来用一类单独的体来描述。这种结合而成的类的成员称为终结体可能事件（telic eventuality）。

在继续讲下面的内容之前，关于分类的策略还要提出两点。

第一点，事件表示易于从一个类移动到另一个类。我们来看下面的例子：

I flew. (14.70)

I flew to New York. (14.71)

第一个例子是简单的行动体，它没有自然的终止点，也不能用带 *in* 的时间表达式来进行时间方面的修饰。另一方面，第二个例子显然是一个完成体事件，因为它有一个终止点，结果得到一个特定的状态，可以用完成体能用的各种方式来进行时间方面的修饰。显而易见，对于一个事件的分类不仅仅是由动词支配的，而且也受到在上下文中整个表达语义的支配。

第二点，像这样的分类常常是很有用的，它们并不说明原因，并不解释为什么用自然语言来表达的这种状态要归入某个特定的类别之中。在第16章中还要讨论这个问题，并根据Dowty(1979)对这些分类的研究，给表示方法勾画出一个轮廓。

14.4.5 信念表示

有相当数量的单词和词语具有所谓“创造世界的能力”（worldcreating ability）。之所以说“创造世界的能力”，是因为它们的意义表示包含的逻辑公式并不一定在现实世界中真实存在，而是某个假设世界中的东西。具有这种“创造世界的能力”的单词有 *believe*，*want*，*imagine* 和 *know* 等。这些具有创造世界的能力的单词通常都采用各种类似于句子的成分作为它们的论元。

我们来研究如下的例子：

I believe that Mary ate British food. (14.72)

应用面向事件的方法，可以说这个句子中有两个事件：一个是*believing*事件，这个事件是关于说话者的某个特定信念，另一个是*eating*事件，这个事件说明信念的东西是什么。不管时间信息，应用把事件具体化的方法，可以得到如下的意义表示：

$$\exists u, v\ ISA(u, Believing) \wedge ISA(v, Eating) \\ \wedge Believer(u, Speaker) \wedge BelievedProp(u, v) \\ \wedge Eater(v, Mary) \wedge Eaten(v, BritishFood)$$

这样的表示看起来似乎比较直接，所有参与的角色都表示出来了，而且两个事件也合情合理地结合在一起了。但是，在这种连接的表示中，所有的连接部分都必须为真。在这样的场合，其结果是一个命题，即*an eating of British food by Mary*。具体地说，删除连接符号把这个公式分解成不同的公式，可以造出如下的公式：

$$\exists v\ ISA(v, Eating) \\ \wedge Eater(v, Mary) \wedge Eaten(v, BritishFood)$$

这种表示比我们想说的还清楚。说话者相信这个命题的事实并不能使这个命题成为真命题；这个命题只是在说话者的信念所表示的世界中才是真的。我们所需要的是具有相似结构的表示，而其中要给事件*Eating*一个特殊的地位。

注意，恢复到本章前面使用过的简单谓词表示是无济于事的。在使用这种谓词表示来表达句子时，最常见的错误是采用如下的表示：

$$Believing(Speaker, Eating(Mary, BritishFood))$$

这种表示的问题在于它不是有效的FOPC。谓词*Believing*的第二个论元应该是一个FOPC项，而不是一个公式。这种句法错误反映了一个深刻的语义学问题。在FOPC中谓词应该在所模拟领域的客体之间成立，而不能在所模拟领域的客体的关系之间成立。因此，FOPC缺乏有意义的手段来表示完整的命题之间的关系，而这种关系恰恰是单词*believe*，*want*，*imagine*和*know*等所要表示的。

处理这种情况的标准办法是使用算符（operator）来增强FOPC，使我们能够做出全逻辑公式的命题。让我们来研究如何使用这种方法来处理例句（14.72）。我们可以引入一个称为*Believes*的算符，取两个FOPC公式作为它的论元。一个公式指派一个believer，另一个公式指派所相信的命题。使用这个算符可以得到如下的意义表示：

$$Believes(Speaker, \exists v ISA(v, Eating) \\ \wedge Eater(v, Mary) \wedge Eaten(v, BritishFood))$$

使用这样的方法，单词*Believes*对这个意义表示的贡献完全不是一个FOPC命题，而是一个可应用于所相信命题的算符。因此，正如第15章将要讨论的，这些“创造世界的动词”在语义分析中所起的作用与一般的像*eat*这样的普通动词很不相同。

正如我们所看到的，在任意给定的时刻，要确定谁相信谁究竟要做什么事情是非常困难的。在第18章中将会看到，在一个交互系统中，当对话过程出现改变时，要抓住用户的信念是一个非常重要的任务。

在逻辑公式中使用的诸如*Believes*这样的算符，称为模态算符（modal operator）。相应地，用模态算符来增强的逻辑，称为模态逻辑（modal logic）。除了模拟信念之外，模态逻辑在常识的表示方面还有很多用处，其中最引人注目的是对于时间词和假设词的表示。

模态算符和模态逻辑提出了很多复杂的理论和实践问题，我们现在不能详细讨论这些问题。其中最重要的问题如下：

- 当出现特定模态算符时如何进行推理
- 这些特定的算符可以应用于哪些种类的逻辑公式
- 模态算符如何与逻辑量词和逻辑连词交互
- 这些算符对于公式中的各个项的相等关系有什么影响

这些问题中的最后一个是对话系统中模拟智能代理的知识和信念的结果，值得在这里做些讨论。在标准的FOPC系统中，相等的逻辑项彼此之间可以自由替换，而不会对它们所出现的句子的真值发生影响。我们来研究下面的例子：

Snow has delayed Flight 1045. (14.73)

John's sister's flight serves dinner. (14.74)

如果这两个航班是相同的，那么用*Flight 1045*来替换*John's sister's flight*对于两个句子的真值都不会产生任何影响。

现在把这些例句改写为如下的句子：

John knows that snow has delayed Flight 1045. (14.75)

John knows that his sister's flight serves dinner. (14.76)

这里就不能进行前面那样的替换了。John可以很好地知道1045航班给延误了，但他不知道他姐姐的航班被延误了，因为他可能不知道他姐姐的航班号。换言之，尽管我们假定这两个句子是真的，并且John的姐姐确实在1045航班上，我们也不能对下面的句子的真值做出任何判断：

John knows that snow has delayed his sister's flight. (14.77)

包含像*know*这样的模态算符的情况称为参照不透明（referentially opaque）。在参照不透明的情况下，相同项的替换可能成功，也可能不成功。在一般情况下，这样的替换是可行的，这种情况称为参照透明（referentially transparent）。

14.4.6 缺陷

我们在14.3节中说过，在自然语言话段的意义表示中存在一些普通的错误，这些错误来自现实的语言与FOPC中的成分之间存在混淆不清或者相等判别的情况。我们来研究下面的例子，从表面上看来它似乎是一个标准的蕴涵规则：

If you're interested in baseball, the Rockies are playing tonight. (14.78)

这个句子似乎可以直接转写为如下的FOPC公式：

$$HaveInterestIn(Hearer, Baseball) \Rightarrow Playing(Rockies, Tonight)$$

这样的意义表示是有漏洞的，原因很多。最明显的一个原因来自FOPC蕴涵的语义。如果听话者对于棒球运动没有兴趣，这个公式就没有意义。具体地说，当前提分句是假的时，我们不能对结果分句做出任何结论。但是，这样的结论显然是可笑的，无论听话者是否对棒球运动感兴趣，Rockies的比赛肯定是要进行的。

现在我们来研究下面的例子：

One more beer and I'll fall off this stool. (14.79)

这个句子可以简单地转写为用连接词连接起来的两个分句：一个表示“喝啤酒”的事件，一个表示“醉倒”的事件。在这种场合，在表层中使用的单词*and*使得这个句子中的蕴涵关系变得晦

暗不清了。这两个例子给我们的教训是：英语中诸如 *and*，*or* 和 *if* 这样的单词与 FOPC 中使用的同样名字的成分之间的关系不太明确。

另外还有一个重要的问题。不要忘记，FOPC公式中使用的名字是完全缺乏明确含义的。我们来研究下面的常量：

InexpensiveVegetarianIndianFoodOnTuesdays

尽管这个常量有明显的形态，但是作为 FOPC 的这个项本身并不比常量 X99 具有更多意义。参看 McDermott(1976)关于这种命名方式固有的危险性的论述。

14.5　有关的表示方法

多年来，为了捕捉在自然语言处理系统中语言话段的意义，提出了很多表示方法。除了逻辑方法之外，还有两种广泛使用的方法，一种是语义网络（ semantic network ）方法，一种是框架（ frame ）方法，框架方法又称为槽填充（ slot-filler ）表示法。KL-ONE(Brachman and Schmolze, 1985)和 KRL (Bobrow and Winograd, 1977)等系统在自然语言处理系统中的知识表示方面做出了影响深远的努力。

在语义网络中，客体用图的结点来表示，客体之间的关系用有名字的连接边来表示。在基于框架的系统中，用类似于第 11 章中讨论过的特征结构来表示客体，因此它当然也可以很自然地用一个图来表示。在这样的表示方法中，特征称为槽（ slot ），而这些槽的值，或者可以用填充者（ filler ）来表示，或者可以用原子值来表示，或者可以用另一个嵌套的框架来表示。下面的图示说明，如何用基于框架的方法来表示例句（ 14.72 ）：

I believe Mary ate British food.

$$\begin{bmatrix} \text{BELIEVING} & & \\ \text{BELIEVER} & \text{SPEAKER} & \\ \text{BELIEVED} & \begin{bmatrix} \text{EATING} & \\ \text{EATER} & \text{MARY} \\ \text{EATEN} & \text{BRITISHFOOD} \end{bmatrix} & \end{bmatrix}$$

这种意义表示方法目前已被广泛地接受，它可以比较容易地被转写为等价的 FOPC 命题。

14.6　意义的其他表示方法

把语言输入转写成能够恰当捕捉意义的概念的离散符号形式表示，这样的研究引起了很多的争论，这不足为奇。下面将简短扼要地介绍这些争论的主要内容。

14.6.1　作为行动的意义

当我们考虑命令句的语义时，自然会想到关于“作为行动的意义”（ meaning as action ）的观点。根据这样的观点，话段被看成是行动，而这些话段的意义则存在于听话者听这些话段的结果时所激发的过程（ procedure ）中。在设计历史上很重要的SHRDLU系统时，就遵循了这样的观点。SHRDLU 的设计人 Terry Winograd(1972b)把这种观点总结如下：

> 这个模型的一个基本观点是：所有的语言使用都可以想像成在听话人那里激发过程的一种手段。我们可以把话段想像成一个程序，这个程序间接地在听话人的认知系统内引起一系列操作。

最近提出的过程性语义模型是Bailey et al.(1997)，Narayanan(1997a, 1997b)和Chang(1998)的执行图式（executing schema）模型或者x图式（x-schema）模型。这个模型的直觉是，事件语义的各个部分，包括前面描述过的“体”的因素，都是基于感官发动机过程（sensory-motor process）的图式描述的，诸如初始化、重复、完成、用力和努力等。这个模型通过一个称为Petri网(Murata, 1989)的概率自动机来表示事件在“体”方面的语义。在这个模型中的Petri网具有诸如准备、处理、结束、延缓、结果等状态。

例如，*Jack is walking to the store* 这个句子可以激发 *walking* 事件的一个处理（process）状态，*Jack walked to the store* 这样的完成事件可以激发一个结果（result）状态。*Jack walked to the store every week*这样的重复性行动可以在一个关于重复性行动的模型中激发，而处理和结果是这个模型中的结点。使用感官发动机基元作为语义描述基础这种思想的另一个根源是Rieger(1996)关于在学习空间介词语义的计算模型中视觉基元作用的研究工作。

14.6.2 作为真值的意义

在语言学、自然语言处理、人工智能和认知科学中的形式意义表示的作用，与它们在很多哲学派别中的作用很不相同。在前面的各种方法中，规则的名字从语言输入变成了适当的、无歧义的、在操作上有用的表示[①]。

但是，对于哲学家来说，把一个句子从原来的自然形式转写成另外的人工形式，并不能使我们离它的意义更接近(Lewis, 1972)。尽管形式表示可以使实际的语义学研究工作变得容易一些，但是这些形式表示本身并没有多大意思。根据这样的观点，重要的工作是功能和过程，因为它们决定了从这些形式表示到被模拟的世界之间的映射关系。这些方法中最有意思的是确定句子或其形式表示的真值条件（truth condition）的那些功能。

14.7 小结

本章介绍意义表示的方法。下面是本章的主要内容：

- 在计算语言学中意义表示的主要方法是建立形式意义表示（formal meaning representation），以便捕捉与语言输入的内容有关的意义。这些意义表示的目的是在语言和关于世界的普通知识之间建立一座桥梁。
- 说明这种意义表示的句法和语义构架称为意义表示语言（meaning representation language）。这种语言有各种各样的变体，它们广泛用于自然语言处理和人工智能中。
- 这样的意义表示需要有能力支持语义处理的计算要求，包括需要确定命题的真值，支持无歧义的表示，表达变量，支持推理以及具有充分的表现力。
- 人类语言具有各种特征来传达意义。其中最重要的特征是表达谓词论元结构（predicate-argument structure）的能力。
- 一阶谓词演算是一种容易理解的、在计算上可循的意义表示语言，它能够提供意义表示语言需要的很多东西。
- 包括范畴、事件和时间意义的重要类别都能用FOPC来表达。相应于诸如信念和愿望等概念的命题要把FOPC扩充到能够包容模态算符（modal operator）。
- 语义网络和框架能够在FOPC的构架内表达。

① 当然，对于什么才是有用的认识，会由于领域的不同而有很大的不同。

14.8　文献和历史说明

自然语言处理中陈述意义表示的早期计算机应用是在问答系统(Green et al., 1961; Raphael, 1968; Lindsey, 1963)的环境下进行的。这些系统都使用针对性很强的意义表示来表达回答问题时所需的各种事实。然后把问题转写为能够与知识库中的事实相匹配的一种形式。Simmons(1965)对于早期学者们在这个方面的努力进行了回顾和总结。

Woods(1967)研究了在问答系统中类似于FOPC的表示方法，用这种方法来代替当时在这个领域的那些针对性很强的表示方法。后来Woods(1973)在具有里程碑意义的Lunar系统的研究中进一步发展并扩充了这种方法。有趣的是，Lunar系统中的表示方法既使用了真值条件，也使用了语义。Winograd(1972b)在SHRDLU系统中也使用了基于Micro-Planner语言的类似表示方法。

在同一时期，对于语言的认知模型和记忆有兴趣的研究人员在各种形式的联想网络表示方法的研究方面做了很多工作。虽然一般人们都认为语义网络是Quillian(1968)提出来的，但是Masterman(1957)最早把语义网络之类的知识表示方法应用在计算方面。在这个时期，在语义网络的框架内进行了大量研究工作(Norman and Rumelhart, 1975; Schank, 1972; Wilks, 1975c, 1975b; Kintsch, 1974)。Fillmore关于格角色的概念(Fillmore, 1968)开始被很多研究人员引入其意义表示研究中。Simmons(1973)最早把格角色作为自然语言处理的表示的一个部分。

Woods(1975)和Brachman(1979)对于语义网络的实际含义的精细研究导致了更精致的类似于语义网络的语言的研究，其中包括KRL(Bobrow and Winograd, 1977)和KL-ONE(Brachman and Schmolze, 1985)。随着这些研究的进一步发展和更精确的定义，人们清楚地发现，这些表示方法只不过是FOPC再加上一些特殊推理过程的具有不同约束的变体。Brachman and Levesque(1985)是这方面的研究论文的合集，很有用处。Russell and Norvig(1995)对意义表示的一些最新研究进行了很好的描述。

在生成语法时期，把语义结构指派给自然语言句子的语言学研究是从Katz and Fodor(1963)的工作开始的。这种简单的基于特征的表示方法的局限性以及这种方法对当时很多语言学问题在逻辑上的适应性很快引导人们使用各种谓词－论元结构作为优先的语义表示方法(Lakoff, 1972; McCawley, 1968)。后来，Montague(1973)把真值条件模型理论的框架引入语言学理论中。Montague的工作把形式句法理论和各种形式语义框架更紧密地结合起来。对于Montague语义学以及它在语言学理论中的作用可参阅Dowty et al.(1981)和Partee(1976)。

把事件表示为具体化的客体的研究工作应该归功于Davidson(1967)。这里介绍的把事件的参与者具体化的方法应该归功于Parsons(1990)。模态算符的使用以及知识表示和信念表示应该归功于Hintikka(1969)。Moore(1977)是第一个把这种方法应用于计算的学者。Fauconnier(1985)从认知科学的角度研究了有关信念和信念空间的各种问题，涉及面很广。当前在时态推理方面的大多数计算方法都基于Allen关于时间间隔的概念(Allen, 1984)。Meulen(1995)提出了一种关于时态和体的现代处理方法。Davis(1990)描述了如何用FOPC来表示在常识领域的各种广泛的知识，包括量词、空间、时间和信念。

当前关于逻辑与语言的全面处理的问题，可以参阅van Benthem and ter Meulen(1997)。经典的语义学读本可以参阅Lyons(1977)。McCawley(1993)是一本不可缺少的教材，它涉及关于语言和逻辑的非常广泛的论题。Chierchia and McConnell-Ginet(1991)从语言学的角度介绍了有关语义学的各种知识。Heim and Kratzer(1998)从当前生成理论的角度论述了一些内容更新的问题。

第15章　语义分析

"Then you should say what you mean," the March Hare went on.
"I do," Alice hastily replied; "at least–at least I mean what I say–that's the same thing, you know."
"Not the same thing a bit!" said the Hatter. "You might just as well say that 'I see what I eat' is the same thing as 'I eat what I see'!"

Lewis Carroll, *Alice in Wonderland*[①]

本章介绍的是语义分析（semantic analysis）这一问题的各种计算方法。语义分析是一个将前一章讨论过的意义表示进行组合，并指派给输入语句的过程。在本章和后续的几章中将会看到，为了产生丰富和准确的意义表示，必须引入广泛的知识源和推理技术。常用的知识源包括：词本身的意义、语法结构所带的意义、话语的结构知识、发生话语的上下文知识以及与话题相关的常识等。

我们首先介绍句法驱动的语义分析（syntax-driven semantic analysis），它的视野很有限。它仅以词典和语法中的静态知识为输入语句指派意义表示。在该方法中，当提及一个输入意义或意义表示时，我们就立刻意识到这是一个贫瘠的表示，即它是上下文独立和推理无关的。这类意义表示符合前一章所介绍的字面意义的概念。

我们沿着这些线索讲述的原因有两个：一方面确实存在受限的应用领域，通过这样的表示足以产生有用的结果；另一方面这些贫瘠的表示可以作为后续处理的输入，进而产生更丰富、更有用的意义表示。比如，第18章和第19章将介绍这些意义表示怎样用于扩展话语处理，而第21章将介绍它们怎样用于机器翻译。

接着，15.5节将介绍更适用于实际问题的其他两个语义分析方法。第一个方法称为语义语法（semantic grammar），它已被广泛用于interactive对话系统的构建。在该方法中，我们通过所讨论领域的语义实体和关系来建立语法元素。我们将看到该方法所用的实际算法与15.1节所描述的那些算法很类似。区别仅在于所用的语法。

最后，15.5节所描述的方法针对的是从大规模文本中抽取少量相关信息的任务。我们将会看到，这类信息抽取任务并不需要其他方法所要求的完整句法分析，而是通过一系列受限的有限状态（通常情况下）自动机的层叠式组合来生成一个鲁棒的语义分析器。

15.1　句法驱动的语义分析

本节详细介绍的方法是基于组合性原则（principle of compositionality）的[②]。

---

① 这是Lewis Carroll的《爱丽丝梦游仙境》中的一段对话，译文如下，供读者参考：
"那你怎么想就怎么说吧，"三月兔继续说。
"我正是这样的，"爱丽丝急忙回答，"至少……至少凡是我说的就是我想的，你知道，这是一回事。"
"根本不是一回事，"帽匠说，"那么，你说'凡是我吃的东西我都能看见'和'凡是我看见的东西我都能吃'，也算是一样的了？"

② 这通常称为Frege组合性原则。这种称谓似乎没有道理，因为在Frege的著作中从来没有明确地阐明这一原则。事实上，他在许多著作中支持非组合观点。在Janssen(1977)中对这一话题有更详细的讨论。

这种方法的内在关键思想是：一个句子的意义可以由它的几个部分的意义组合而成。当然，从表面理解时这个原则似乎不大有用处。我们都知道句子是由词构成的，而词是语言中意义的最基本载体。因此，这个原则所告诉我们的全部似乎是：应该由组成句子的词的意义来组成句子所代表的意义。

幸运的是，Mad Hatter给出了让这个原则有用的提示：一个句子的意义并不仅仅依赖于句中的词汇，它还依赖于句中词汇的顺序、词汇所形成的群组和词汇之间的关系。当然，该提示可以简单地理解为：句子的意义部分取决于句法结构。因此，在句法驱动的语义分析中，意义表示的组成是由在第9章、第11章和第12章中讨论的语法所提供的句法成分和关系来引导的。

首先，我们假定输入句的句法分析结果将形成语义分析器的输入。图15.1是对这个明显的管道流方法（该假定的直接反映）的图形表示。输入句首先通过剖析器获得它的句法分析结果，接着这个句法分析结果被传给语义分析器来产生意义表示。值得注意的是，尽管在本图中是用剖析树作为输入，而实际上其他的句法表示，比如特征结构或词汇依存图等都可以作为输入使用。本节的后续部分将假定以树形结果作为输入。

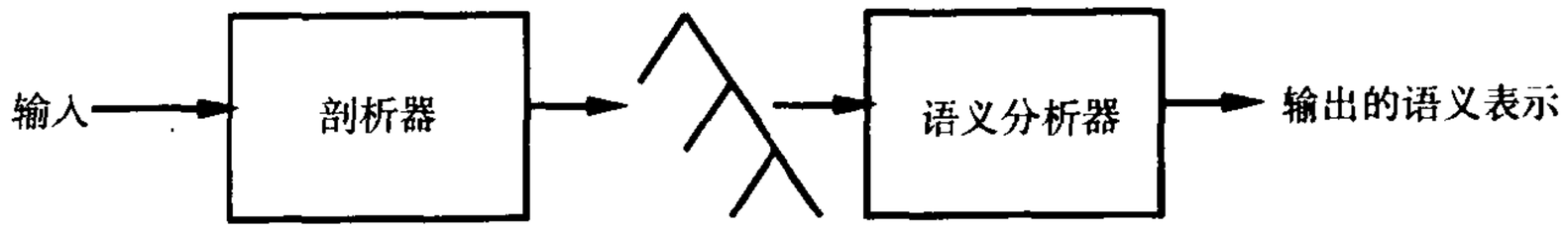


图15.1 用于语义分析的简单的管道流方法

在继续讲述之前，我们首先应该澄清对该方法中歧义的角色所做的重要假定。在现在介绍的句法驱动方法中，由于句法和词汇中都存在歧义，因此将产生成倍的歧义意义表示。解决这些歧义并不是我们所定义的语义分析器的工作，而是通过使用领域知识和上下文知识在候选项中进行选择的后续理解处理。当然，我们可以通过使用鲁棒的词性标注器、介词短语附着机制以及词义排歧原理（将在第17章中介绍），以减少歧义表示的数量。

让我们以下句为例来介绍如何进行这样的语义分析：

AyCaramba serves meat. （15.1）

在图15.2中给出的是该例句的带有恰当意义表示的简化剖析树（缺少特征附着）。从虚线可以看出，以该剖析树为输入的语义分析器首先通过动词*serves*对应的子树来获取意义表示。接着，分析器获取句中两个名词所对应的意义表示，然后以从动词获取的意义表示为模板，用名词短语的意义表示来填充动词表示中合适的变元，这样就生成了句子整体的意义表示。

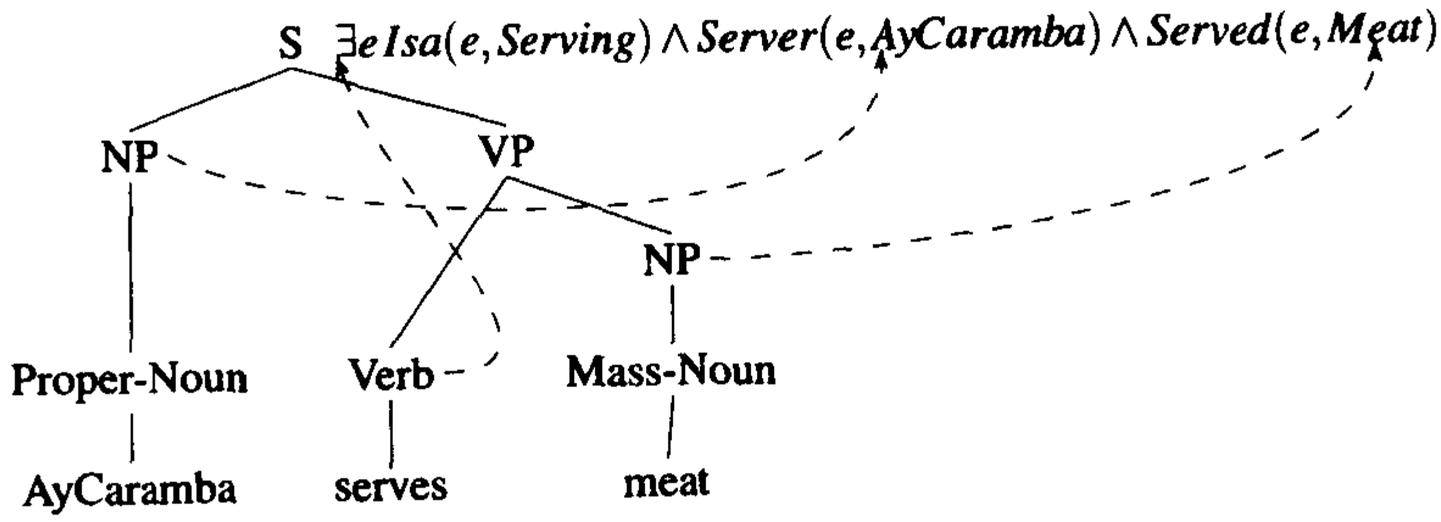


图15.2 语句 AyCaramba serves meat 的剖析树

遗憾的是，这个简单的故事存在着相当明显的问题。正如我们所描述的，用于理解图15.2中剖析树的函数至少需要知道：最终意义表示是基于动词的哪个模板，动词出现在树的什么位置，动

词所对应的论元是什么，在动词意义表示中每个论元应填充的又是什么角色，等等。换言之，为生成需要的意义表示，我们需要关于这个特例以及它的剖析树的丰富知识。假如对于任何合理的语法都存在无限多的剖析树，那么对于每个可能的剖析树都仅基于一个语义函数来进行语义分析，这样的任何方法都是不可行的。

幸运的是，我们以前也曾遇到过类似的问题。语言并不是通过枚举所允许的字符串或剖析树来定义的，而是通过定义有限的工具来定义的。利用这些有限的工具，就能生成所需的输出的集合。因此，在面向句法的方法中，语义知识的正确职责首先与用于产生剖析树的有限的工具集（语法规则和词典条目）相关。这种方法称为规则到规则假设（rule-to-rule hypothesis）(Bach, 1976)。

设计基于该方法的分析器涉及成分及其所带意义的概念。本节之后的内容可以看成是对下面两个问题的回答：

- 句法成分带有意义到底意味着什么？
- 为了能够组成较大的意义，这些（句法成分所带有的）意义到底必须具备哪些特征？

15.1.1 给上下文无关语法规则扩充语义

沿用第11章所用的方法，我们将开始给上下文无关语法规则扩充语义附着（semantic attachment）。这些附着可以看成是指令（instruction），由它们确定了如何通过句法结构的各个成分的意义来计算其整体的意义。从理论上讲，扩充的规则应符合下面的形式：

$$A \rightarrow \alpha_1 \cdots \alpha_n \qquad \{f(\alpha_j.sem, \cdots, \alpha_k.sem)\}$$

基本上下文无关规则的语义附着出现于{...}中，位于规则的句法成分的右边。该符号表示指派给结构 A 的意义表示（将以 $A.sem$ 表示），可以通过对组成 A 的成分的语义附着的某些子集运行函数 f 来进行计算。

语义附着作为一个简单函数应用的特征是相当抽象的。为了使这个概念更具体，对于例句（15.1）和例句（15.2）中的系列例句，先忽略计算意义表示必需的语义附着。我们将从本例中更具体的实体，也就是名词短语出发，逐步迈向更复杂的有关整句意义表示的表达式。在本例中，*FOPC* 常量 *AyCaramba* 和 *Meat* 表示具体的实体。首先，需要将这些常量与引导它们的树的成分联系起来。为实现这个目标，第一步要将这些常量与表示这些词的词法规则配对。而正是这些规则将常量引入句子。

$$ProperNoun \rightarrow AyCaramba \qquad \{AyCaramba\}$$

$$MassNoun \rightarrow meat \qquad \{Meat\}$$

这两个规则表明通过由常量 *AyCaramba* 和 *Meat* 组成的规则，可以生成与子树相关的意义。

但是，正如图15.2中的箭头所示，这些FOPC常量对最终意义表示的贡献并不是直接通过这些规则所对应的子树来传递的，而是通过在树中较高层的 *NP* 对最终的意义表示起作用。遵循组合性原则，通过规定上层 *NP* 可以从它们的子女处获得意义，就可以实现这个间接贡献。在这两个实例中，我们假定子女的意义表示只是简单地向上复制给父母。

$$NP \rightarrow ProperNoun \qquad \{ProperNoun.sem\}$$

$$NP \rightarrow MassNoun \qquad \{MassNoun.sem\}$$

这些规则表明名词短语的意义表示与它们各自的组件（由ProperNoun.sem和MassNoun.sem表示）的意义表示相同。通常，对于非分支的语法规则，子女的语义表示将直接复制给父母而保持不变。

在继续讲述之前，我们应该指出这里的讨论至少有一处是非常容易混淆的。虽然例子中第一个 *NP* 规则的语义附着是 *ProperNoun.sem*，然而该规则所产生的树的语义值却是 *AyCaramba*。正确区

分一个规则的语义附着和由该规则所生成的树的语义值是非常重要的。前者是一系列如何构建意义表示的指令，而后者组成遵循这些指令的结果。

回到例句，在说明了表示中的常量后，我们继续探讨该话段所蕴涵的事件，在本例中也就是*serves*事件。如图15.2所示，一个普通的服务（*serving*）事件涉及一个服务者（*server*）和所服务的东西（*something served*），这可以用下面的逻辑公式来捕捉：

$$\exists e,x,y\ Isa(e,Serving) \wedge Server(e,x) \wedge Served(e,y)$$

作为该动词初始的语义附着，我们可以简单地就以上面的逻辑公式作为服务（serves）的语义附着，如下所示：

$$Verb \rightarrow serves$$
$$\{\exists e,x,y\ Isa(e,Serving) \wedge Server(e,x) \wedge Served(e,y)\}$$

沿着剖析树向上，下一个要考虑的成分是*VP*，而*VP*管辖着*serves*和*meat*。与*NP*不同，我们不能简单地将这些子女的语义表示复制给父母*VP*，而是需要将*NP*的意义并入（incorporate）*Verb*的意义，并在*VP.sem*中指派这种表示。在本例中，就是用作为*Serves*事件的*Served*角色的第二个论元的逻辑项*Meat*来替换变量*y*。这样就形成了下面的意义表示，这可以解释为*someone serves meat*。

$$\exists e,x\ Isa(e,Serving) \wedge Server(e,x) \wedge Served(e,Meat)$$

为产生这种表示，*VP*的语义附着必须能够提供一种手段，通过该手段可以将*V.sem*的量化的变量*y*替换为由*NP.sem*规定的逻辑常量*Meat*。从这个例子抽象地看，*VP*语义附着必须具备两方面的能力：一方面必须清楚地知道动词论元的语义将替换的是动词语义附着中的哪个变量；另一方面必须具备实现这种替换的能力。

遗憾的是，到目前为止我们还不具备直接实现上述机理的技术储备。我们附着给*V.sem*的FOPC公式没有提供任何有关这三个量化变量如何替换以及替换先后的手段，而且在目前的FOPC框架下即使我们知道，也没有处理这种替换的简单方法。

幸运的是，一种被称为lambda的符号(Church, 1940)，正好提供了我们需要的这种形式化参数的功能。它是对FOPC的符号扩展。这种符号扩充了FOPC句法，使它能够引入下面的表达式：

$$\lambda x P(x)$$

该表达式由三部分组成，首先是希腊符号λ，接着是一个或多个变量，最后是使用这些变量的FOPC表达式。

λ表达式的有用之处在于它们用于逻辑项时可以生成新的FOPC表达式这一能力，在这些新的FOPC表达式中，形参变量由指定的项来绑定。这种处理称为λ化简（λ-reduction）是λ变量由指定的FOPC项来进行简单的字面替换，并去掉λ的过程。下面的表达式说明的是将一个λ表达式用于常量*A*，接着对该表达式进行λ化简的结果：

$$\lambda x P(x)(A)$$
$$P(A)$$

λ符号提供了前述的在动词语义中需要的两种能力：形参表使我们可以获得变量集，而λ化简实现了我们渴望的用项来替换变量的处理。

这项技术的重要且实用的变形就是将一个λ表达式作为另一个λ表达式的一部分，如下所示：

$$\lambda x \lambda y\ Near(x,y)$$

上面的表达式非常抽象，可以解释为某些东西与另一些东西接近（near）的状态。下面的表达式说明的是一个单独的λ应用，以及对这种嵌入式λ表达式后续的化简：

$$\lambda x \lambda y\, Near(x,y)(ICSI)$$

$$\lambda y\, Near(ICSI,y)$$

这里很重要的一点是化简后的表达式仍是一个λ表达式。第一次化简绑定了变量x，去掉了外面的λ，显现出内部的表达式。正如我们所期望的，这个化简后的λ表达式还可以接着化简生成如下完全指定的逻辑公式：

$$\lambda y\, Near(ICSI,y)(AyCaramba)$$

$$Near(ICSI,AyCaramba)$$

这项技术被称为currying(Schönkfinkel, 1924)[①]。借助currying技术可以将多个论元的谓词转换为一系列单个论元的谓词。我们即将看到，这项技术在一个谓词的论元不全是谓词剖析树的子女时尤为重要。

有了λ符号和λ化简处理，就有了继续进行*VP*成分语义附着计算的工具。注意，我们需要的是一个用*VP*的*NP*成分所提供的意义表示来替换代表*Served*角色的变量的方法。这可以通过两个步骤来实现：首先将动词的语义附着转换成一个λ表达式，然后在得到该*VP*的语义附着后，对*NP*语义应用该表达式。第一步可以通过指定x（对应于*Served*角色的变量）为一个λ表达式的λ变量来实现，由该λ表达式提供*serve*的语义附着。

$$Verb \rightarrow serves$$
$$\{\lambda x \exists e,y\, Isa(e,Serving) \wedge Server(e,y) \wedge Served(e,x)\}$$

这个语义附着使得变量λ可以通过将该表达式应用到一个逻辑项而得到绑定。因此，对本例中及物动词*VP*规则的语义附着确定了λ应用，这里λ表达式是由*Verb.sem*指定的，论元是由*NP.sem*指定的。

$$VP \rightarrow Verb\ NP \qquad \{Verb.sem(NP.sem)\}$$

这个λ应用的目的是用*NP.sem*包含的值来替换或绑定这个λ表达式的单个形参x。通过λ化简去掉λ，并显现出用常量*Meat*替换变量x后的内部表达式。这个表达式，即动词短语*serves meat*的意义，也就是*VP.sem*的值。

$$\exists e,y\, Isa(e,Serving) \wedge Server(e,y) \wedge Served(e,Meat)$$

为完成整个例子，我们必须对*S*规则生成语义附着。像*VP*规则一样，这个规则必须将一个*NP*论元并入事件表示的适当角色，目前这个事件表示位于*VP.sem*。因此，必须组成另一个λ应用，由*VP.sem*的值指定λ表达式，而位于句子开头的*NP.sem*指定要并入的最后的论元。

$$S \rightarrow NP\ VP \qquad \{VP.sem(NP.sem)\}$$

遗憾的是，目前上面所列的*VP.sem*的值并没有指定我们需要的λ表达式。在*VP*规则上进行λ应用的结果是一个含有两个存在量化变量的通用FOPC表达式。*Verb*附着应该包含一个嵌入的λ表达式，这个表达式使得在语法的*S*层来绑定*Server*的角色成为可能。因此，我们将*Verb*附着修改为下面的表示：

$$Verb \rightarrow serves$$
$$\{\lambda x \lambda y\, \exists e\, Isa(e,Serving) \wedge Server(e,y) \wedge Served(e,x)\}$$

① currying是一个标准术语，但是Heinn and Krakzer(1998)中提出了用术语Schönkfinkelization来替代currying的有趣的观点，因为Curry后来是基于Schönkfinkel的工作的。

*Verb*附着的主要部分是由一个λ表达式嵌入一个λ表达式来组成的。外部的表达式提供了首次λ化简中可以替换的变量，而内部的表达式可用于对应于*Server*角色的最后变量的绑定。在动词语义附着中多层λ表达式的变量顺序，清楚地表明在句法中动词论元具有所期望的位置这一事实。

在图 15.3 中给出了本例在每个结点上标注相对应的语义值的剖析树。

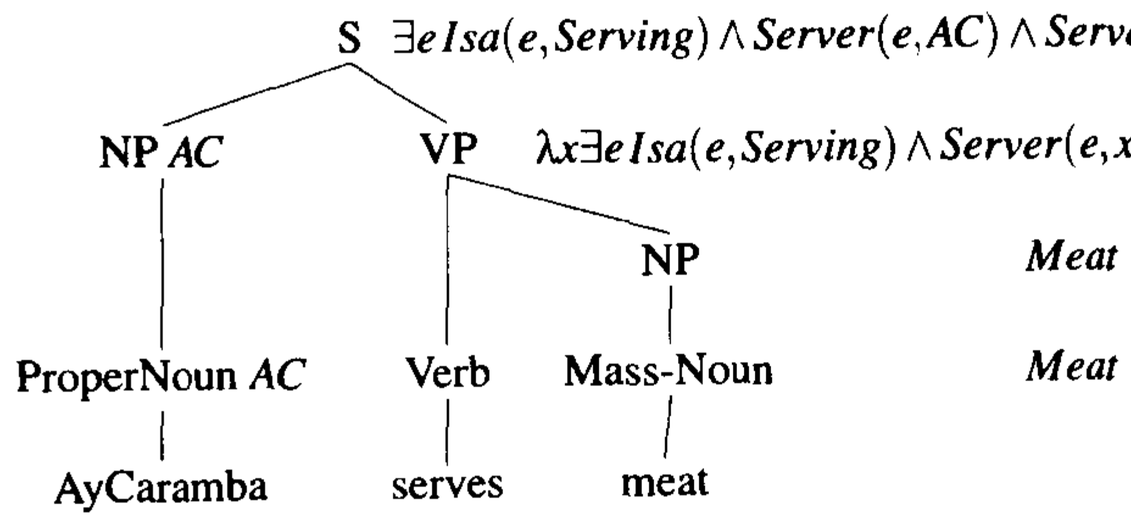


图 15.3　*Aycaramba serves meat* 的语义附着剖析树

借助这个例子，我们介绍了语义分析的句法驱动方法中的许多基本技术。在 15.2 节将介绍针对一些主要的英语语法范畴的更完整的语义附着的详细清单。然而，在讲述这些清单之前，我们将首先分析几个其他的例子。通过这些例子将引入几个实现该方法所需的基本架构，并将举例说明对一个语法建立语义附着的一般方法。

让我们考虑下面所示的例句（15.1）的变形：

A restaurant serves meat.　　（15.2）

因为上例与例（15.1）中的动词短语相同，因此我们将注意力集中于主语名词短语的语义推导和后续它与动词短语在S规则下的结合。作为出发点，我们假设下面的公式是对本例中主语意义的似乎合理的表示：

$$\exists x\, Isa(x, Restaurant)$$

将这个新的表达式与已形成的动词短语的表达式相结合，就生成了下面的意义表示：

$$\exists e, x\, Isa(e, Serving) \wedge Server(e, x) \wedge Served(e, Meat) \wedge Isa(x, Restaurant)$$

在这个公式中，restaurant（由变量 x 表示）被指定为 *Server* 的角色，并作为 *Server* 谓词的第二个论元出现。

遗憾的是，指定用于产生 *S* 规则语义附着的λ应用并不能生成这个结果。将λ简化按字面意义理解为一种文字替换，导致了在下面的表达式中，名词短语的整体意义表示嵌入在 *Server* 谓词中：

$$\exists e\, Isa(e, Serving) \wedge Server(e, \exists x Isa(x, Restaurant)) \wedge Served(e, Meat)$$

尽管这个表达式在直觉上是合理的，但它不是一个有效的 FOPC 公式。像本例中这样的表示 rstaurant 的表达式不能作为谓词的论元出现，因为只有 FOPC 项才能作为论元。事实上，因为定义λ表达式的方法只能用于 FOPC 项,因此将附着于 *VP* 的λ表达式应用于主语的语义从一开始就是不合适的。

我们可以用一种类似于λ表达式在用于解决动词短语和*S*语义附着问题时所采用的方式来解决这个问题，也就是通过对现有 FOPC 句法增加一个新的符号，来推动我们需要的意义表示的合成。在此，我们将引入复杂项（complex-term）的概念，复杂项允许像 $\exists x\, Isa(x, Restaurant)$这样的表达

式出现在那些只有普通FOPC项才能出现的地方。从形式上讲，一个复杂项是由三部分结构组成的表达式：

<Quantifier variable body> （<量词 变量 体>）

对当前的例子应用这个符号，可以得到下面的表示：

$$\exists e\ Isa(e, Serving) \wedge Server(e, <\exists x Isa(x, Restaurant)>) \wedge Served(e, Meat)$$

与λ表达式相同，这种符号变化只有在能够提供一种直接方式将它转换为普通FOPC句法时才有用。这一点可以根据下面的方案，利用复杂项来重写任意谓词得以实现。

$$P(<Quantifier\ variable\ body>)$$
$$\Rightarrow$$
$$Quantifier\ variable\ body\ Connective\ P(variable)$$

换言之，复杂项：

1. 是从它出现的谓词中抽取的；
2. 是由问题中代表客体的变量来替换的；
3. 具有通过使用适当的连接词对新表达式预设的变量、量词和体。

下面的表达式说明的是当前例子中这个复杂项的化简：

$$Server(e, <\exists x Isa(x, Restaurant)>)$$
$$\Rightarrow$$
$$\exists x Isa(x, Restaurant) \wedge Server(e, x)$$

用于附着抽取的公式与前面新表达式的连接词的选择取决于所用的量词的类型：∧用于存在量词∃，而⇒用于全称量词∀。

能获取组成复杂项的三个部分也是十分有用的。因此，我们将扩展用于指向成分的语义的句法，让该句法允许指向组成成分的每个部分。举例说明，假如*A.sem*是一个复杂项，那么*A.sem.quantifier*，*A.sem.variable*和*A.sem.body*则分别获取复杂项的量词、变量和体。

回到例句（15.2），现在我们来介绍短语*a restaurant*意义表示的生成。给定该名词短语简单的句法结构，*NP*语义附着问题就变得十分直截了当。

$$NP \rightarrow Det\ Nominal \qquad \{<Det.sem\ x\ Nominal.sem(x)>\}$$

上面的附着生成了一个复杂项，该复杂项由三部分组成：从*Det*获取的量词、任意变量、将*Nominal*与该变量联系起来的λ表达式的λ应用。通过λ应用可以确保正确的变量出现在*Nominal*指定的谓词中。

限定词的附着简单地指定了所用的量词：

$$Det \rightarrow a \qquad \{\exists\}$$

名词性范畴的任务就是生成需要在名词短语中使用的*Isa*公式和λ表达式。

$$Nominal \rightarrow Noun \qquad \{\lambda x Isa(x, Noun.sem)\}$$

最后，名词附着简单地规定正在讨论的范畴的名字。

$$Noun \rightarrow restaurant \qquad \{Restaurant\}$$

通过这个例子，我们逐步引入了五个具体化的机制，并通过这些机制对本节开始讲述的语义附着的抽象函数的特征进行实例化：

1. 将普通 FOPC 表达式与词汇项相关联；
2. 将类似 λ 表达式的函数与词汇项相关联；
3. 语义值从子女向父母的复制；
4. 将类似 λ 表达式的函数的应用用于句法成分的一个或多个子女的语义；
5. 通过采用复杂项来允许量化的表达式被临时视为项。

FOPC句法和英语句法之间的差异是我们引入λ表达式和复杂项的动机。这些外部的逻辑手段拉近了FOPC句法和它所处理的语言的句法之间的距离，使语义分析处理变得很方便。使用这类手段的意义表示通常被称为准逻辑形式（quasi-logical form）或中间表示（intermediate representation）。注意，这两个术语之间有细微的不同。“准逻辑形式”术语常应用于那些可以通过一些简单的句法变换被容易地转换为逻辑表示的表示，而“中间表示”术语一般是指，为了试图产生更深意义表示，作为进一步分析处理的输入的意义表示。

根据本章的目的，我们这里的意义表示是准逻辑形式，因为它们可以很容易地转换为FOPC。但从稍广阔的视角来看，它们也是中间表示，因为我们确实还需要进一步的诠释来使它们更接近合理的意义表示。

这一节所引入的规则几乎都不符合在组合框架下引导语义附着的设计的原则。一般来说，正是词汇规则对我们的意义表示提供了内容级的谓词和项。语法规则的语义附着将这些谓词和项正确地联系起来，但总体来说确实没有将谓词和项引入生成的意义表示。

15.1.2 量词辖域和复杂项的转译

遗憾的是，上面所给出的将包含复杂项的表达式转译成FOPC表达式的方案并不是惟一的。考虑下面的例子，以及它的原始的未限定作用域的意义表示：

Every restaurant has a menu. (15.3)

$$\exists e Isa(e, Having) \wedge Haver(e, < \forall x\, Isa(x, Restaurant) >) \wedge Had(e, < \exists y\, Isa(y, Menu) >)$$

如果为使*Haver*角色的量词具有外部的辖域而重写*Haver*和*Had*角色中的复杂项，则可以得到下面的意义表达，它对应于人们对该句的常识性的理解。

$$\forall x Restaurant(x) \Rightarrow \exists e, y\, Having(e) \wedge Haver(e, x) \wedge Isa(y, Menu) \wedge Had(e, y)$$

反之，如果项的重写是由相反的方向进行的，则生成下面的FOPC表示结果，它表示所有的饭店（restaurant）共用一个菜单（menu）。

$$\exists y\, Isa(y, Menu) \wedge \forall x\, Isa(x, Restaurant) \Rightarrow \exists e Having(e) \wedge Haver(e, x) \wedge Had(e, y)$$

这个例子说明量词辖域歧义的问题——一个简单的带有两个复杂项的逻辑公式导致了两种截然不同的FOPC表示。在最糟糕的情况下，有*N*个量词的句子将有*O*(*N*!)种不同的量词辖域。

在实际应用中，大部分系统都使用一个启发式优先规则的特定集从所有可能排列中生成最优先的形式。在没有优先规则的情况下，使用从左到右量词的排序。这与量词的表面顺序一致。然后，

使用领域的特定知识来接受一个量词公式或拒绝该公式，并请求其他公式。Alshawi(1992)介绍了一种全面的方法来生成可靠的量词辖域。

15.2 给英语片断附加语义分析

本节介绍给一些小的英语片断附加语义分析，大多数英语片断基于核心语言工具（Core Language Engine）(Alshawi, 1992)。为了保持表述上的简洁性，在本章的其余部分，如果不是必需的，将省略与这些规则相关的特征结构。但是要注意，为了确保在正确的场合使用正确的规则，这些特征是必需的，并且最重要的一点是这些特征对于确保基于次范畴特征结构的正确动词实体的使用是必需的。

15.2.1 句子

在语义讨论中，大量的篇幅都用于说明句子的处理。本节将扩展覆盖面，把在第9章引入的其他句子类型也包括进来，比如祈使句，yes-no 问句和 wh 问句。首先以下面的例句作为开始：

Flight 487 serves lunch. (15.4)

Serve lunch. (15.5)

Does Flight 207 serve lunch? (15.6)

Which flights serve lunch? (15.7)

这些例句的意义表示都包含在飞机上提供午餐这个命题，但是在句子所使用的背景下这些命题所服务的角色是不同的。更具体地说，第一个例句是给听众传递一个事实信息，第二句是请求一个动作，而后两句是询问信息。为了捕捉这些不同，正如我们在第14章中引入belief算子一样，我们将引入一组可用于FOPC句子的算子。具体地说，我们将针对说明句、祈使句、yes-no问句和wh-问句的FOPC表示，分别引入*DCL*算子、*IMP*算子、*YNQ*算子和*WHQ*算子。

正确使用这些算子来生成意义表示，需要对每个可能的句子类型有一组正确的语义附着。对于说明句，可以将已使用的基本句法规则简单地转换为公式：

$$S \rightarrow NP\ VP \qquad \{DCL(VP.sem(NP.sem))\}$$

通常以*DCL*开头的意义表示可以理解为给现有知识库加入一个对事实的陈述。

祈使句以动词短语开头并缺少一个明显的主语。因为这个默认主语，对主要动词短语的意义表示将由一个λ表达式和一个未绑定的代表这个默认主语的λ变量组成。为了处理这种情况，我们可以通过对一个虚拟的变量使用最终的λ化简来为λ表达式简单地补充一个主语。将*IMP*算子用于这个表示，则可以得到如下的语义附着：

$$S \rightarrow VP \qquad \{IMP(VP.sem(DummyYou))\}$$

将这个规则用于例句（15.5），则有下面的表示：

$$IMP(\exists eServing(e) \wedge Server(e, DummyYou) \wedge Served(e, Lunch)$$

祈使句可以看成是一种言语行为（speech act），这一点将在第19章中讨论。

正如第9章所述，yes-no问句是由一个位于句首的助动词，加上一个主语名词短语和一个动词短语组成的。下面的语义附着简单地忽略了助动词，除了*YNQ*算子，其他都与说明句的意义表示相同：

$$S \rightarrow Aux\ NP\ VP \qquad \{YNQ(VP.sem(NP.sem))\}$$

将上面的规则用于例句（15.6），则可以生成下面的表示：

$$YNQ(\exists e Serving(e) \wedge Server(e, Flt207) \wedge Served(e, Lunch))$$

yes-no问句可以理解为：在给定知识库中目前所含知识的前提下，判断问句意义的命题部分是真还是假。采用第 14 章所描述的这类语义，通过判断该命题是否在这个知识库中或是否可以从这个知识库推断出，就可以回答 yes-no 问题。

与yes-no问题不同，*WH* 主语问题询问的是句中主语的特定信息而不是整体句子。下面的语义附着生成了由三部分组成的表示：*WHQ* 算子，对应于句子主语的变量和命题体。

$$S \rightarrow WhWord\ NP\ VP \qquad \{WHQ(NP.sem.var, VP.sem(NP.sem))\}$$

下面是将这个规则用于例句（15.7）得到的表示：

$$WHQ(x, \exists e, x\ Isa(e, Serving) \wedge Server(e, x) \wedge Served(e, Lunch) \wedge Isa(x, Flight))$$

可以通过对主语变量返回一组指派（assignment）来回答这类问题，这些指派使得结果命题成立（相对于当前知识库）。

最后，考虑下面的 *WH* 非主语问题：

How can I go from Minneapolis to Long Beach?　　（15.8）

在这类例子中，问题不是关于句子的主语而是一些其他论元，或命题整体的一些方面。在这种情况下，意义表示需要提供一种能表明问题所问的是什么的指示。通过规定助词的语义作为 WHQ 算子的一个论元，就可以让下面的语义附着提供这样的指示信息。

$$S \rightarrow WhWord\ Aux\ NP\ VP \qquad \{WHQ(WhWord.sem\ VP.sem(NP.sem))\}$$

对例句（15.8）应用这个规则就可以得到下面的表示：

$$WHQ(How, \exists e\ Isa(e, Going) \wedge Goer(e, User) \wedge Origin(e, Minn) \wedge Destination(e, LongBeach))$$

我们将在 15.5 节和第 19 章中讲到，正确回答这类问题需要相当数量的领域内的特定推理。例如，正确回答例句（15.8）的方法是查询在特定出发地和目的地城市之间的航班。然而，注意在实际问题中从未提到过的航班或飞行。因此，该问题-回答不得不应用该领域的知识以产生这样的结果：关于去这些地方的问题实际上是关于去这些地方的航班的问题。

最后，我们应该清楚地指出这种特殊的附着仅用于较简单的 *WH* 问题，这些问题不存在省略论元或没有嵌入从句。正如第 11 章所讨论过的，一旦问题中出现长距离依赖，就需要其他机制来确定所问的是什么。Woods(1977)和 Alshawi(1992)提供了对处理 *WH* 非主语问题的一般机理的详尽讨论。15.5 节将介绍一种在实际系统中常用的更特定的方法。

15.2.2　名词短语

我们已经看到，名词短语的意义表示可以是普通的FOPC项，也可以是复杂项。在下面的几节中将详细介绍为产生意义表示而对最常见的几种英语名词短语进行的附着语义分析。遗憾的是，英语名词短语的句法几乎没有为它们的意义表示提供帮助。通常能做的最好情形不过是提供一个相对中间的意义表示，而这个中间意义表示又可以被用于进一步的理解处理。

复合名词

复合名词，又称名词－名词序列，由名词的简单序列组成，如下面的例子所示：

Flight schedule (15.9)

Summer flight schedule (15.10)

正如第9章所指出的，这类结构的句法结构可以通过正则表达式*Noun**或下面的上下文无关文法来捕捉：

$$Nominal \rightarrow Noun$$

$$Nominal \rightarrow Noun\ Nominal$$

在这些结构中，序列中最后的名词是短语的中心语，表示一个主体，从语义上通过一些未指明的方式与序列前面的其他名词相关联。一般情况下，通过这个结构可以表示各种各样的常性识关系。辨别这些关系的性质完全超出了本章介绍的这种表面语义分析的范围。下面的规则的语义附着建立了一个模糊的表示，只是指出在中心名词和修饰名词之间存在一种语义关系，并通过递增方式指出中心名词和每个在它的左边的名词的关系：

$$Nominal \rightarrow Noun\ Nominal$$
$$\{\lambda x\ Nominal.sem(x) \wedge NN(Noun.sem, x)\}$$

关系*NN*用于指定一个复合名词的修饰元素与中心名词之间的关系。使用下面的例子，可以生成下面的意义表示：

$$\lambda x Isa(x, Schedule) \wedge NN(x, Flight)$$

$$\lambda x Isa(x, Schedule) \wedge NN(x, Flight) \wedge NN(x, Summer)$$

注意，这个公式成功地实例化了表示*Schedule*的项，而避免生成代表*Flight*或*Summer*的项。

所属名词短语

回想第9章所述，在所属名词短语中使用复杂限定词将名词短语与所属标记组合起来，例如*Atlanta's airport*和*Maharani's menu*。这些词之间的关系很容易被表示为一种抽象的所有关系。但是，稍微思考一下就会发现一个城市和它的机场之间的关系与一个饭店与它的菜单之间的关系几乎没有相同点。因此，与复合名词一样，最好的方式是只给出多种成分抽象的语义关系。

$$NP \rightarrow ComplexDet\ Nominal$$
$$\{< \exists x Nominal.sem(x) \wedge GN(x, ComplexDet.sem) >\}$$

$$ComplexDet \rightarrow NP\ 's \qquad \{NP.sem\}$$

对*Atlanta's airport*应用这些规则导致了下面的复杂项：

$$< \exists x Isa(x, Airport) \wedge GN(x, Atlanta) >$$

必须由后续的语义理解来确定由关系*GN*所表示的关系实际上是一个地点。

形容词短语

英语中的形容词可分为两大类：名前形容词和谓语性形容词。这种分类可由下面EBRP中的例子来说明：

I don't mind a cheap restaurant. (15.11)

This restaurant is cheap. (15.12)

对于名前形容词，一个明显但经常出错的语义附着方案如下面的规则所示：

$$Nominal \rightarrow Adj\ Nominal$$
$$\{\lambda x\ Nominal.sem(x) \wedge Isa(x, Adj.sem)\}$$
$$Adj \rightarrow cheap \qquad \{Cheap\}$$

这个解决方案将形容词确定的谓词应用于表示名词的变量。对于cheap restaurant这样的例子，则生成下面的相当合理的表示：

$$\lambda x\ Isa(x, Restaurant) \wedge Isa(x, Cheap)$$

这就是被称为交集语义（intersective semantic）的一个例子。这个短语的意义可以看成是由名词确定的范畴和由形容词确定的范畴的交集。在这个例子中，这相当于是便宜东西的范畴和饭店的范畴的交集。

遗憾的是，这个解决方案经常是错的。例如，想想下面短语*small elephant*，*former friend*和*fake gun*的意义表示：

$$\lambda x\ Isa(x, Elephant) \wedge Isa(x, Small)$$
$$\lambda x\ Isa(x, Friend) \wedge Isa(x, Former)$$
$$\lambda x\ Isa(x, Gun) \wedge Isa(x, Fake)$$

这些表示每个都有其特殊性。第一个例子说明这里特指的大象（elephant）隶属于小物体（small thing）这一普通范畴，而这很可能是错的。第二个例子的奇怪之处有两点：一方面它宣称所提及的人是一个朋友（friend），但这是错的；另一方面它使用了一个以前的事物（former thing）这样的完全不合理的范畴。类似地，第三个例子宣称所提及的物体是枪（gun）；而事实上，fake表示它不是。

与复合名词一样，在当前组合框架下对这些问题没有一个巧妙的解决方案，因此最好的方法是一开始仅仅指出这是一种特殊的修饰关系，并假定在额外的相关知识帮助下的进一步处理能够将这种模糊关系替换成正确的表示(Alshawi, 1992)。

$$Nominal \rightarrow Adj\ Nominal$$
$$\{\lambda x\ Nominal.sem(x) \wedge AM(x, Adj.sem)\}$$

将这个规则用于*cheap restaurant*就得到了下面的公式：

$$\exists x\ Isa(x, Restaurant) \wedge AM(x, Cheap)$$

注意，即使是这种削减难度的方案，对于*fake*和*former*这样的例子也不能产生逻辑上正确的表示。在这两种情形下，语义表示强调的都是：所提及的客体事实上是它们所描述的范畴的成员。通常，这个问题的解决方法不得不基于所提及的形容词和名词的特殊语义。例如，*former*的语义不得不引入一些时间推理，而*fake*需要有关范畴和概念的性质的推理能力。

15.2.3 动词短语

通常，动词短语的语义计算的方案都是基于函数应用的概念的。在大多数情况下，附着于动词的λ表达式可以简单地应用于动词论元的语义附着。但是在许多情况下也不得不少许偏离这个通用模式。

不定式动词短语

许多英语动词以各种形式的动词短语作为论元。这使得常规的动词短语的语义方案变得复杂，因为这些动词短语论元与中心语动词的其他论元之间的相互作用不太清楚。

思考下面的例子：

I told Harry to go to Maharani. (15.13)

这个例子的意义表示应该是类似下面的公式：

$$\exists e, f, x\ Isa(e, Telling) \wedge Isa(f, Going) \wedge Teller(e, Speaker) \wedge Tellee(e, Harry) \wedge ToldThing(e, f) \wedge Goer(f, Harry) \wedge Destination(f, x)$$

在这个意义表示中，有两个值得注意的有趣的事情：第一它是由两个事件组成的，第二是其中的一个参与者，*Harry*，在两个事件中都兼有角色。生成该复杂表示的难点在于动词短语，该动词短语支配的动词*tell*需要类似下面的语义附着：

$$\lambda x, y\ \lambda z\ \exists e\ Isa(e, Telling) \wedge Teller(e, z) \wedge Tellee(e, x) \wedge ToldThing(e, y)$$

从语义角度，我们可以将*Tell*的这个次类框架理解为由三个语义角色组成：告诉这个动作的执行者（a person doing the telling），告诉这个动作的接收者（a recipient of the telling）和正在传送的命题（the proposition being conveyed）。

本例的困难部分在于正确表达主动词短语的意义表示。如图15.4所示，*Harry*兼有双重角色：*Telling*事件中的*Tellee*和*Going*事件中的*Goer*。但是，在不定式动词短语中生成*Going*事件时无法获得有关*Harry*的信息。

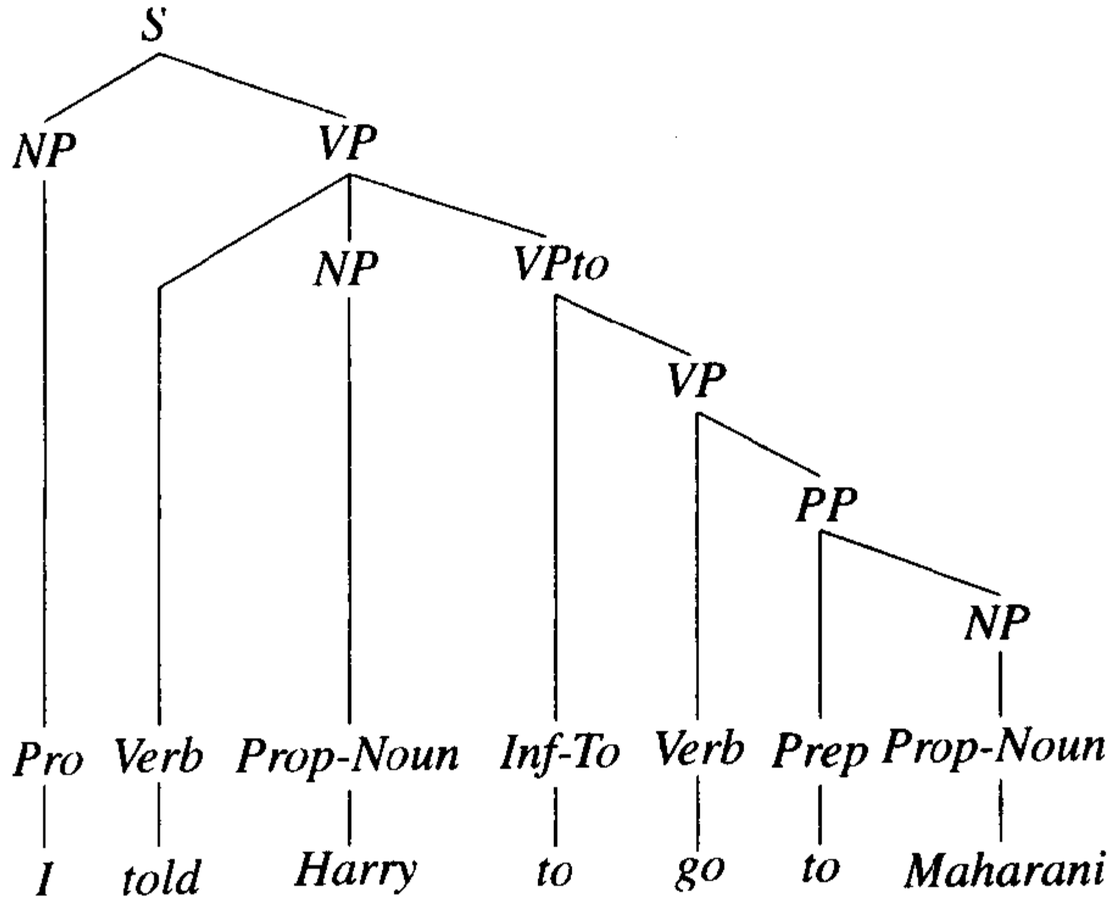


图15.4 *I told Harry to go to Maharani*的剖析树

尽管对这个问题有好几种可能的解决方案，但是通常最好采用统一的方法。因此，我们首先将动词的语义简单应用于动词的其他论元的语义，如下所示：

$$VP \rightarrow Verb\ NP\ VPto \qquad \{Verb.sem(NP.sem, VPto.sem)\}$$

因为不定式动词短语结构中的“*to*”对它的意义并没有贡献，所以我们只是将儿子动词短语（child verb phrase）的意义向上复制给不定式动词短语。注意，我们正依赖于未知的特征结构来确保只有正确的动词短语才能用于这个结构。

$$VPto \rightarrow to\ VP \qquad \{VP.sem\}$$

在这个解决方案中，动词的语义附着有两个任务：一个是将*NP.sem*，也就是*Goer*引入*VPto.sem*，另一个是引入*Going*事件作为*Telling*的*ToldThing*。下面的语义附着实现了这两个任务：

$$\begin{aligned}
&\textit{Verb} \rightarrow \textit{tell}\\
&\quad \{\lambda x,y\\
&\qquad \lambda z\\
&\qquad\quad \exists e,y.\textit{variable Isa}(e,\textit{Telling})\\
&\qquad\qquad \wedge \textit{Teller}(e,z) \wedge \textit{Tellee}(e,x)\\
&\qquad\qquad \wedge \textit{ToldThing}(e,y.\textit{variable}) \wedge y(x)
\end{aligned}$$

在该方法中，λ变量*x*是*telling*中的*Tellee*角色，也是该不定式的语义的论元，该不定式的语义由变量*y*的λ表达式表示。表达式*y*(*x*)表示一个λ化简，它的作用是把*Harry*作为*Goer*插入*Going*事件。符号*y.variable*类似于在复杂项变量中使用的符号，这样就能够在不定式意义表示中访问代表*Going*事件的事件变量。

注意，在这个方法中λ化简的定义是固定的，也是宽松的。允许λ表达式作为其他λ表达式的论元，而技术上只有FOPC项才能用于这个角色。这一技术的方便性类似于复杂项的使用，这使得在生成表示时允许临时将复杂表达式当成项来处理。

15.2.4 介词短语

从一个相当抽象的层面看，介词短语具有两个独特的功能：一个是维持它们的中心语和附着成分之间的二元关系，另一个是预示具有论元结构的成分的论元。而这两个功能对应于基于语义附着角度的两种截然不同的介词短语。下面来研究语法中介词短语的三个位置以及在这三个位置上介词短语所服务的角色：名词短语的修饰语、动词短语的修饰语和动词短语的论元。

修饰名词的介词短语

修饰性介词短语表示的是：所修饰的概念（介词短语的外部）和介词短语的中心语之间的二元关系。考虑下面的例子及其意义表示：

$$\text{A restaurant on Pearl} \tag{15.14}$$

$$\exists x\ \textit{Isa}(x,\textit{Restaurant}) \wedge \textit{On}(x,\textit{Pearl})$$

约束这个例子的相关语法规则如下：

$$\textit{NP} \rightarrow \textit{Det Nominal}$$

$$\textit{Nominal} \rightarrow \textit{Nominal PP}$$

$$\textit{PP} \rightarrow \textit{P NP}$$

以自底向上的方式处理，这种关系性介词的语义附着应该提供一个有两个位置的谓词，并且它的论元是两个λ表达式，如下式所示：

$$\textit{P} \rightarrow \textit{on} \qquad \{\lambda y \lambda x\ \textit{On}(x,y)\}$$

通过这样的安排，谓词的第一个论元由介词短语的中心语确定，而第二个论元由介词短语最终所依附的成分确定。下面的语义附着提供第一部分：

$$\textit{PP} \rightarrow \textit{P NP} \qquad \{\textit{P.sem}(\textit{NP.sem})\}$$

该λ应用产生一个新的λ表达式，剩余的论元是内部的λ变量。

剩余的论元可以利用下面的名词性结构引入：

$$Nominal \rightarrow Nominal\ PP \qquad \{\lambda z Nominal.sem(z) \wedge PP.sem(z)\}$$

修饰动词短语的介词短语

修饰动词短语的介词短语的一般语义表示方法类似于修饰名词短语的介词短语。不同点在于动词短语规则中修饰的细节部分，而对于介词和介词短语规则的语义附着则没有任何变化。我们来研究由下面的动词短语规则约束的短语 *ate dinner in a hurry*：

$$VP \rightarrow VP\ PP$$

在这个结构中，动词短语成分 *ate dinner* 的意义表示是一个λ表达式，其中λ变量代表的是至今仍然未知的主体。

$$\lambda x \exists e\ Isa(e, Eating) \wedge Eater(e, x) \wedge Eaten(e, Dinner)$$

介词短语的表示也是一个λ表达式，其中λ变量是 *PP* 语义中的第二个论元。

$$\lambda x\ In(x, < \exists h\ Hurry(h) >)$$

为了正确表示被修饰的动词短语，需要通过将 *Eating* 事件变量填入 *In* 表达式的第一个论元槽，将这两个表示联系起来。另外，这个被修饰的表示必须保留一个未绑定 *Eater* 变量的λ表达式作为新的λ变量。下面的附着表达式满足所有这些要求：

$$VP \rightarrow VP\ PP \qquad \{\lambda y VP.sem(y) \wedge PP.sem(VP.sem.variable)\}$$

该语义附着的两个方面仍需要进一步推敲。第一个方面是将成分动词短语的λ表达式应用于变量 *y*。将较低层λ表达式的变量绑定到一个新的变量，使得我们能够将这个较低层的变量提升到一个较高的新生成λ表达式的层。该技术的结果是：一个新的λ表达式采用的变量所具有的角色实际上与较低层表达式采用的原始变量所具有的角色相同。因此，在分析处理中表达式的论元实际获取之前，允许λ表达式被修饰。

在这个附着中第二个新的方面是 *VP.sem.variable* 符号。这个符号用于访问事件变量，代表动词短语（在本例中 *e*）的潜在意义。这与前述的用于访问复杂项的不同部分的符号是类似的。

将这个附着应用于当前的例子，就生成了下面的适合与后续的主语名词短语合并的意义表示：

$$\lambda y \exists e\ Isa(e, Eating) \wedge Eater(e, y) \wedge Eaten(e, Dinner) \wedge In(e, < \exists h Hurry(h) >)$$

作为动词论元的介词短语

这类介词短语用于预示在一些较大的事件结构中一个论元所担当的角色。实际上介词本身并不修饰名词短语的意义。考虑下面带有预示角色功能的介词短语的例子：

I need to go from Boston to Dallas. (15.15)

在这个例子中，*go* 的论元是由介词短语来担当的。但是，这些短语的意义表示应该完全由它们的中心语名词的原本表示来组成。为了处理这种情况，我们用与处理非分支语法规则相同的方法来处理论元介词短语，即名词短语的语义附着不加改变地复制给较大的短语。

$$PP \rightarrow P\ NP \qquad \{NP.sem\}$$

然后，动词短语可以将这个意义表示指派到恰当的事件角色。在第 16 章中，我们将更深入地讨论这些担当论元的介词短语如何对应于潜在的事件角色。

15.3　把语义分析结合到 Earley 剖析中

在 15.1 节中，我们介绍了语义分析器的简单管道流的结构，在这种结构中，将一个完全的句法剖析结果作为输入传给语义分析器。这样做的动机源于组合方法在处理之前就需要句法剖析这一事实。然而，语义分析与句法处理同时进行也是可能的。可能的原因在于，在我们的组合框架中只要给出所有成分部件就可以生成一个成分的意义表示。本节要介绍这样一种方法，它可以把语义分析结合到第 10 章讨论过的 Earley 剖析中。

把语义分析结合到 Earley 剖析中是比较容易理解的。而且与第 11 章把合一运算结合到算法中的处理方法是相同的。对原来的算法需要做三项修改：

1. 将一个新的域引入语法规则，以包含它们的语义附着。
2. 将一个新的域引入线图中的状态，以容纳成分的意义表示。
3. 改变 ENQUEUE 函数，以便于当一个完全状态进入线图时，在状态的语义场中能够计算和存储语义。

在图 15.5 中给出了为生成意义表示而对 ENQUEUE 函数所做的修改。当一个完全状态通过 ENQUEUE 并且这个状态能成功地合并它的合一约束时，ENQUEUE 调用 APPLY-SEMANTICS 来为这个状态计算并存储意义表示。注意，在语义分析之前进行特征合一处理的重要性。这样做确保语义分析处理只是基于有效的剖析树，并且确保能给出语义分析需要的特征。

```
procedure ENQUEUE(state, chart-entry)
  if INCOMPLETE?(state) then
     if state is not already in chart-entry then
        PUSH(state, chart-entry)
  else if UNIFY-STATE(state) succeeds then
     if APPLY-SEMANTICS(state) succeeds then
       if state is not already in chart-entry then
        PUSH(state, chart-entry)

procedure APPLY-SEMANTICS(state)
  meaning-rep ← APPLY(state.semantic-attachment, state)
  if meaning-rep does not equal failure then
    state.meaning-rep ← meaning-rep
```

图 15.5　修改ENQUEUE函数以便于处理语义。如果状态是完全的并且合一成功，则 ENQUEUE 调用 APPLY-SEMANTICS 来计算并存储完全状态的意义表示

这种结合方法相对于管道流方法的根本优点在于这一事实：APPLY-SEMANTICS 能以与合一失败类似的方式失败。如果发现一个语义的非良构形式，在生成意义表示时，则这个相关的状态就能够被阻塞，以防止进入线图。借助这个方法，在句法处理时就实现了语义考虑。第 16 章将详细介绍识别这种非良构形式的几种方法。

遗憾的是，这同时也是把语义结合到剖析中所带来的根本缺点中的一个，也就是大量的精力将耗费在孤立（orphan）成分的语义分析上，而这最终对一个成功的剖析并没有多少贡献。早期实现语义所带来的益处是否大于引入外部语义处理所带来的耗费，这个问题只能根据每个具体的实例来回答（对于不同的实例，答案很可能不同）。

15.4 惯用语和组成性

Ce corps qui s'appelait et qui s'appelle encore le saint empire romain n'était en aucune manière ni saint, ni romain, ni empire.

This body, which called itself and still calls itself the Holy Roman Empire, was neither Holy, nor Roman , nor an Empire.

Voltaire, 1756[①]

正如组合性原则看起来的易碎性，当采用真实语言检验时，组合性原则就很容易陷入困境。在许多情况下，一个成分的意义表示并不基于组成它的词的意义，至少不是直接组合的。研究下面的WSJ的例子：

Coupons are just the tip of the iceberg. (15.16)

The SEC's allegations are only the tip of the iceberg. (15.17)

Coronary bypass surgery, hip replacement and intensive-care units are but the tip of the iceberg. (15.18)

显然这些例子中的每个短语 *the tip of the iceberg* 都与 *tip* 或 *iceberg* 无关，而是大致对应于开始（the beginning）之类的意思。处理这种成语结构的最直接方法是引入为处理这些成语而特别设计的新语法规则。这些成语规则将词典项与语法成分相混合，并引入不是源于组成成语的任何部分的语义内容。在下面的规则中我们给出了该方法的示例：

$$NP \rightarrow \textit{the tip of the iceberg}$$
$$\{\textit{Beginning}\}$$

该规则右边的小写字母项用于精确地表示输入的词。尽管常量 *Beginning* 严格来说不应该看成是这个成语的意义表示，但是它确实反映出这个成语的意义不以组成它的单词的意义为基础。注意，当遇到这个短语时，Earley风格的剖析器采用这个规则将产生两个短语，其中一个代表这个成语，另一个代表组合的意义。

就像其余的语法一样，为了得到正确的规则，我们需要进行一些尝试。研究下面WSJ语料库中的 *iceberg* 的例子：

And that's but the tip of Mrs. Ford's iceberg. (15.19)

These comments describe only the tip of a 1000-page iceberg. (15.20)

The 10 employees represent the merest tip of the iceberg. (15.21)

很明显，上面给出的规则的概括性还不足以处理这些例子。这些例子表明对这个成语存在一个不完全的句法结构，这个句法结构允许使用限定词中的一些变体，也允许对 *iceberg* 和 *tip* 加以形容词性修饰语。一个更有希望的规则应该类似于下面的公式：

$$NP \rightarrow \textit{TipNP of IcebergNP}$$
$$\{\textit{Beginning}\}$$

其中，范畴 *TipNP* 和 *IcebergNP* 被给予一个内部的类似名词的结构，也就是允许一些形容词修饰语和一些限定词的变体，但仍然严格保持这些名词短语的中心语是词典项 *tip* 和 *iceberg*。注意，这个

① 译文如下，供读者参考：
这个团体，以前称自己并且现在仍然称自己为神圣罗马帝国，既不是神圣的，也不是罗马的，也不是一个帝国。

句法解决方案忽略了棘手的情况，也就是说，修饰语*mere*和*1000-page*似乎表明事实上tip和*iceberg*两者都可能在成语的意义中起到某种组合角色的作用。当我们在第16章开始介绍隐喻时，将再回到这个话题。

概括地讲，处理成语至少需要对普通的组合框架进行如下改变：

- 允许词典项与传统语法成分的混合。
- 为了能够处理成语多样性的正确范围，允许生成额外的特定成语的成分。
- 准许在语义附着引入与规则中的任何成分无关的逻辑项和谓词。

显然，这里的讨论只是冰山的一角（the tip of an enormous iceberg）。成语比我们通常所认识的更常见，也更加多样化。这对于许多应用，包括第21章将要讲述的机器翻译，带来了严重的困难。

15.5 鲁棒的语义分析

我们早就注意到，当句法驱动的语义分析在实践中应用时，总是难免采取一些折中处理，以方便系统设计和保持运算的有效性。在下面的章节中会介绍将句法驱动的方法用于实践的两种基本途径。

15.5.1 语义语法

当我们在15.1节首次介绍Frege的组合性原则时，就曾指出该原则中的部分（part）是指句法语法所提供的成分。遗憾的是，由这种语法所提供的句法结构常常不是特别适合组合语义分析的任务。这一点也不足为怪，因为在设计语法时的考虑重点在于捕捉精致的句法概括性和避免过度生成，而不是对语义的敏感性。这种由传统语法所提供的结构与组合语义分析的需求之间的不匹配，尤其体现在以下三个方面：

1. 关键的语义元素常常散布于整个剖析树，使组合需要的意义表示变得很复杂。
2. 剖析树常常包含许多以句法为目的成分，而它们在语义处理中并不担当任何实质角色。
3. 对于许多句法成分的概括性导致的语义附着，其生成的意义表示几乎毫无意义。

作为前两个方面的示例，在图15.6中给出了如下BERP例子的剖析树：

I want to go to eat some Italian food today. (15.22)

这个剖析树的分支结构使得意义表示的关键成分广泛散布于整个剖析树中。同时，在剖析树中大部分结点对该句的意义几乎没有贡献。该结构需要三个λ表达式和一个复杂项，以便于将带有内容的元素汇集到剖析树的顶端。

第三个问题源于在组合性的“规则到规则”（rule-to-rule）方法中需要一致的语义附着。这个需求常产生一些成分，它们对于句法目的来说处于概括性恰当的层面，但对于语义目的来说，这个层面就太高了。充分体现这一点的例子是复合名词和形容词短语，其中，语义附着过于概括，以至于接近无意义。例如，研究当前例子中短语*Italian food*的规则：

$$Nominal \rightarrow Adj\ Nominal$$
$$\{\lambda x\ Nominal.sem(x) \wedge AM(x,\ Adj.sem)\}$$

应用这个语义附着，产生下面的意义表示：

$$\exists x\ Isa(x, Food) \wedge AM(x, Italian)$$

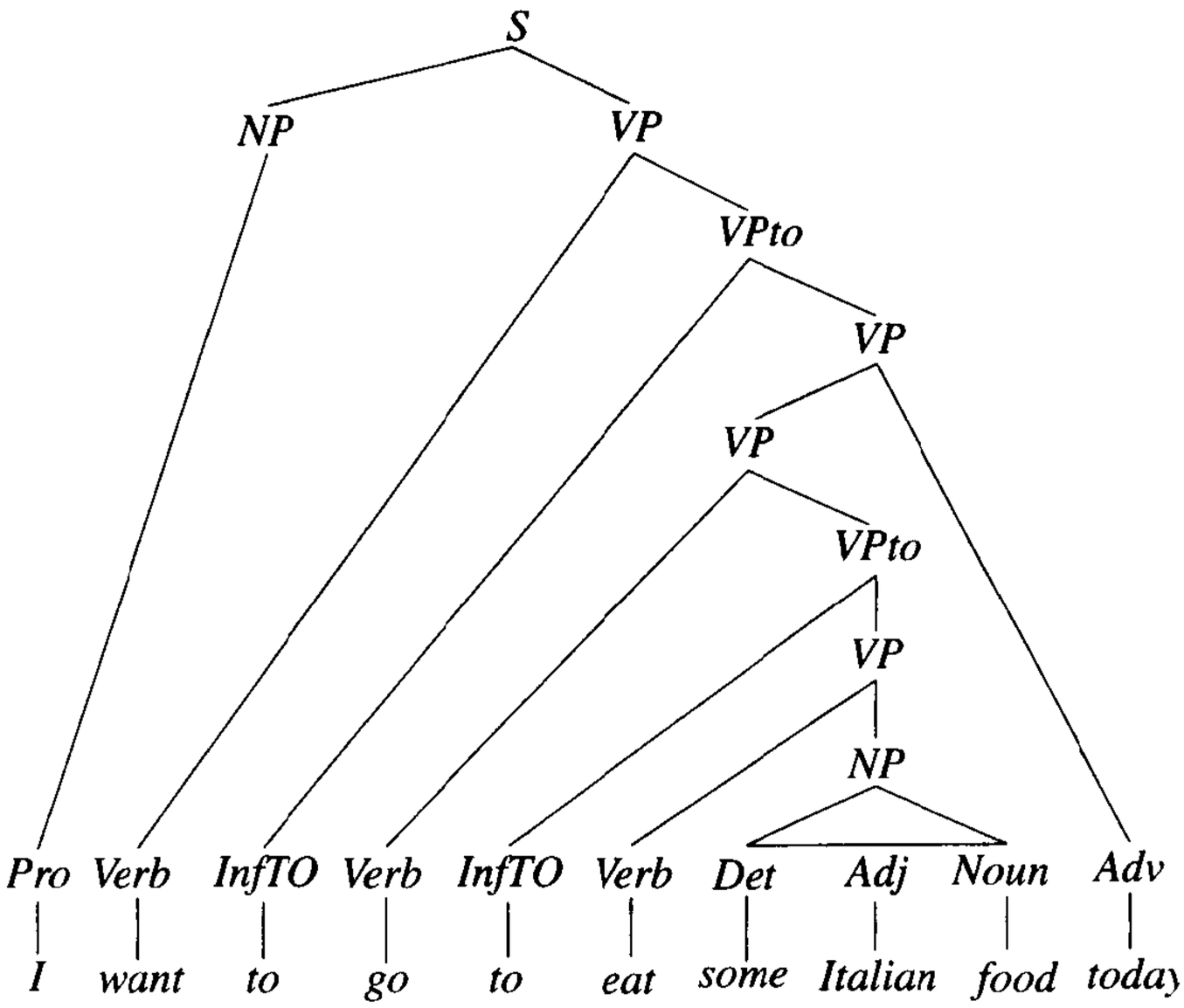


图 15.6 例句（15.22）的句法剖析树。以这个剖析树为例说明语义的内容元素分布于整个剖析树

所有符合这个模式的名词都获得相同的模糊理解，这大致表明了名词可以被形容词修饰。这与我们所知的*Italian food*与*Italian restaurant*这类表达式的意义相差甚远，实际上它们分别表示由特殊方式制成的食物和提供以该方式制成的食物的饭店。遗憾的是，借助深度概括的规则是无法产生这样的理解的。

通过使用语义语法（semantic grammar），这些问题都可以得到解决。语义语法最初是为用于问答和智能辅导(Brown and Burton, 1975)领域的基于文本的对话系统（text-based dialogue system）而设计的。语义语法更直接地面向组合分析的需求。在这个方法中，设计了直接对应于所讨论领域的实体和关系的语义规则和成分。在这样的语法中，关键的语义成分共同出现在单个的规则中，而且规则的概括性与获取敏感的语义分析所需的概括性相一致。

下面讨论上述的两个通用策略是怎样应用于BERP领域的。思考为例句（15.22）中的特殊的信息需求而生成的如下候选规则：

$$InfoRequest \rightarrow User\ want\ to\ go\ to\ eat\ FoodType\ TimeExpr$$

与为成语引入的规则一样，这类规则的右边部分也是非终结符和终结符的自由混和体。在本例中，*User*，*FoodType*和*TimeExpr*表示在该领域以语义为目的的非终结范畴。有了这些，该规则的语义附着就具备了所有必需的信息，通过这些信息可以为这类需求从规则中间成分组合成意义表示。另外，我们不再需要λ表达式，因为这个平直的规则将所有相关的论元都提升到了剖析树的顶端。

现在来研究下面可用于剖析例句中短语*Italian food*的规则：

$$FoodType \rightarrow Nationality\ FoodType$$

这个规则的特殊性质允许产生比前述通用名词规则给出的语义附着更有用的语义附着。更具体地说，它可以生成一个表示，这个表示表明由成分*FoodType*指定的食物是按*Nationality*成分所对应的风格制成的。

在这些领域中使用语义语法的关键原因之一是处理重复和省略的需要。语义语法可以帮助处理这些现象，因为语义语法本质上具有一定的预测能力。更具体地说，基于剖析器主动预测的范畴，语义语法允许剖析器对即将到来的输入做出非常具体的预测。借助这个能力，前指指代和省略元素就能够与具体的语义范畴建立联系。

以下面的 ATIS 例子为例，我们来研究这是怎样实现的：

When does flight 573 arrive in Atlanta?　　(15.23)

When does it arrive in Dallas?　　(15.24)

这类句子可以用下面的规则进行分析，在这个规则中用到了特定领域的非终结符 *Flight* 和 *City*：

$$InfoRequest \rightarrow when\ does\ Flight\ arrive\ in\ City$$

这类规则对确定it所指而给出的信息远多于纯粹的句法规则，纯粹的句法仅仅限制it指向任何以名词短语表示的事物。在实际情况下，这样的系统可以在对话中向前搜寻最近一次使用过 Flight 成分的位置，以确定这个代词的可能指代。第 18 章将详细讨论前指消解（anaphor resolution）。

当然，把一个系统基于语义语法会带来不少缺点，这也不足为奇。最根本的缺点在于该方法几乎完全丧失了重用性（reuse）。将一个领域的句法和语义结合成单一的表示，使得所生成的语法只能用于该领域。相反，原则上保持句法和语义分开的系统，则可以将它们的语法重用于新的领域。丧失重用性的第二个原因是在语法中回避了句法的概括性。这不可避免地导致单个领域的语法在数量上的增长。例如，最初的名词短语规则足够容纳 *Italian restaurant* 和 *Italian food* 这两个短语，但是现在就需要两个不同的规则。事实上，审视 BERP 语料库可以发现对这些短语：*vegetarian restaurant*、*California restaurant* 和 *expensive restaurant*，也需要额外的规则。

另外，语义语法也很容易造成语义的过度生成。作为一个例子，考虑短语 *Canadian restaurant*。它符合上面对民族饭店所给出的规则，并将产生一个意义表示：一个提供加拿大风格的食物的饭店。遗憾的是，几乎可以肯定这是对该短语的不正确的理解，因为在 WSJ 语料库中出现的该短语中没有一次是这个意义，而都是指位于加拿大的饭店。使用语义语法的对话系统基于在限定领域的这种用法比较罕见的假定。

最后，我们应该指出语义语法也许应当被冠以其他称谓，因为这个语法本身从形式上与书中讨论过的任何其他形式体系并没有差别。相应地，也没有针对语义语法的句法分析或语义分析的特殊算法。语义语法可以使用与所用语法体系相符合的任何算法，比如 Earley，或任何其他上下文无关剖析算法。

15.5.2　信息抽取

在自然语言处理的应用，比如问答系统中，对每个输入句子给出一个合理理解是非常重要的，因为提供给用户的错误回答可能导致严重的后果。对于这类应用，面向语义的规则对规则（rule-to-rule）方法是建立对输入句子的彻底理解的一个很好的方法。

然而，其他的应用，比如从商业新闻中抽取合资企业的信息，理解天气预报，或从收音机报道的今天的股票行情中进行简单信息的摘要，都不是特别需要这种细致的理解。这种信息抽取（information extraction）的应用具有两个特点：一方面想获取的知识可以通过相对简单和固定的模板，或带有槽的框架（这些槽需要由文本中的素材来填充）来进行描述；另一方面文本中只有一小部分信息需要填入模板或框架，其他的都可以被忽略。

方法论探讨：信息抽取系统的评测

典型的信息抽取系统与信息检索系统有许多共同点，因此根据信息检索改编了几个标准评价矩阵，包括精度（precision），召回率（recall），误识率（fallout）以及结合矩阵F系数（F-measure）。

召回率是对系统从文本中抽取了多少相关信息进行度量的系数。因此，它是对系统的覆盖面（coverage）的度量。召回率由下面的公式定义：

$$\text{召回率} = \frac{\text{系统所给出的正确的答案数}}{\text{文中可能正确的答案的总数}}$$

精度是对系统返回的信息中有多少是真正正确的进行度量的系数，又称为正确度（accuracy）。精度由下面的公式定义：

$$\text{精度} = \frac{\text{系统所给出的正确的答案数}}{\text{系统给出的答案数}}$$

误识率是对系统忽略文中错误信息的能力进行度量的系数，它的定义如下：

$$\text{误识率} = \frac{\text{系统所给出的不正确答案数}}{\text{文中不正确的答案数}}$$

注意精度和召回率是相反的，对于保持不变的系统，如果追求高精度，就必然带来召回率的降低。类似地，追求系统的高覆盖率，就将带来更多的错误，造成精度的降低。这导致人们使用精度和召回率相结合的度量系数，即F系数。在F系数中，利用参数β来平衡精度和召回率。F系数由下面的公式定义：

$$F = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

当β等于1时，表示给精度和召回率相同的权重。当β大于1时，表示偏爱精度，而当β小于1时，表示偏爱召回率。

例如，第五届信息理解会议（Message Understanding Conference,简称MUC-5(Sundheim, 1993)，是由美国政府组织的信息抽取会议）中的一项任务就是从商业新闻中抽取国际合资企业的信息。下面是取自文献 Grishman and Sundheim(1995)的样例中的两个句子：

bridgestone Sports Co. said Friday it has set up a joint venture in Taiwan with a local concern and a Japanese trading house to produce golf clubs to be shipped to Japan.

The joint venture, Bridgestone Sports Taiwan Co., capitalized at 20 million new Taiwan dollars, will start production in January 1990 with production of 20,000 iron and "metal wood" clubs a month.

信息抽取系统的输出可以是带有几个填充槽的单一模板，也可以是较复杂的分层次关联起来的客体。MUC-5的任务指定后者，即较复杂的输出，要求系统能够生成分层次关联的描述合资企业合伙人、合资公司以及相关的行为、所有权和资金的模板。在图15.7中给出了FASTUS(Hobbs et al., 1997)系统生成的模板结构。

许多信息抽取系统是根据层叠式有限状态机设计的。例如，FASTUS系统所生成的上述模板就是基于层叠的方式的，语言处理的每个层面抽取文本中的一些信息，并将这些信息传送给较高层次的语言处理，如图15.8所示。

许多系统的全部或大部分层次都是基于有限自动机来设计的，尽管在实践中因为单个自动机通过特征记录得以加强（例如在FASTUS系统中），或因为自动机只是作为完整剖析器的一个预处理

阶段，例如(Gaizauskas et al., 1995; Weischedel, 1995)，或因为自动机与其他组件通过决策树相结合(Fisher et al., 1995)，所以从技术上看大多数完整的系统并不是有限状态的。

TIE-UP-1:	
Relationship:	TIE-UP
Entities:	"Bridgestone Sports Co."
	"a local concern"
	"a Japanese trading house"
Joint Venture Company	"Bridgestone Sports Taiwan Co."
Activity	ACTIVITY-1
Amount	NT$20000000
ACTIVITY-1:	
Company	"Bridgestone Sports Taiwan Co."
Product	"iron and "metal wood" clubs"
Start Date	DURING: January 1990

图 15.7 FASTUS(Hobbs et al., 1997)信息抽取引擎对例句生成的模板

序号	层面	说明
1	词例	将字符流转换为单词序列
2	复杂词	识别多词短语、数字和专有名词
3	基本短语	将句子切分为名词词组、动词词组和小品词
4	复杂短语	识别复杂的名词词组和动词词组
5	语义模式	识别语义实体和事件并插入模板
6	合并	合并文本中不同部分对同一实体或事件的所指

图 15.8 FASTUS(Hobbs et al., 1997)中的处理层次。每个层面抽取一种特定的信息，并将这类信息传送给较高的层次

下面让我们跟随文献Hobbs et al.(1997)和Appelt et al.(1995)来概略了解一下FASTUS系统每个层次的具体实现。在词例还原以后，第二阶段识别像 *set up* 和 *joint venture* 这样的多词单元和 *Bridgestone Sports Co.* 这样的名称。名称识别器是一个转录机，该转录机由大规模的特定映射集组成，以便于处理地名、人名、组织名、公司名、联盟名和演出团体名等名称。下面给出的是为获取像 *San Francisco Symphony Orchestra* 和 *Canadian Opera Company* 这样的演出组织名称而设计的典型规则。虽然这些规则是由上下文无关语法写成的，因为不存在递归，所以这些规则可以自动编译为有限状态转录机：

Performer-Org	→	(pre-location) Performer-Noun+ Perf-Org-Suffix
pre-location	→	locname | nationality
locname	→	city | region
Perf-Org-Suffix	→	orchestra, company
Performer-Noun	→	symphony, opera
nationality	→	Canadian, American, Mexican
city	→	San Francisco, London

在第二阶段也可以将 *forty two* 这样的序列转录为正确的数值（在4.6节曾讨论过这个问题）。

FASTUS系统的第三个阶段利用有限状态规则生成了一系列基本短语，比如名词词组、动词词组等。图15.9给出了FASTUS的基本名词短语识别器的输出结果。注意，其中使用了一些特定领域的基本名词短语，比如*Company*和*Location*。

Company	Bridgestone Sports Co.
Verb Group	said
Noun Group	Friday
Noun Group	it
Verb Group	had set up
Noun Group	a joint venture
Preposition	in
Location	Taiwan
Preposition	with
Noun Group	a local concern
Conjunction	and
Noun Group	a Japanese trading house
Verb Group	to produce
Noun Group	golf clubs
Verb Group	to be shipped
Preposition	to
Location	Japan

图15.9 利用Appelt and Israle(1997)中描述的有限状态规则，FASTUS的基本短语识别器在第二阶段所获得的输出结果

回忆一下，第10章曾讲述过如何将这些基本短语组合成复杂的名词词组和动词词组。这是在FASTUS的第四个阶段，通过处理下面的连词和数量短语：

20,000 iron and "metal wood" clubs a month,

以及介词短语：

production of 20,000 iron and "metal wood" clubs a month,

而实现的。第四个阶段的输出是一个复杂的名词词组和动词词组的列表。在第五个阶段，我们利用这个列表，忽略所有没能形成复杂词组的文本，识别复杂词组中的实体和事件，并将这些识别出的客体插入正确的模板。实体和事件的识别是通过手工编写的有限状态自动机实现的，这些自动机的转移基于特定的复杂短语类型，而这些复杂短语类型是通过特定的中心词或特定特征，比如*company*，*currency*或*date*来标注的。

例如，上面给出的新闻中的第一个句子通过下面的两个正则表达式来生成语义模式（其中，NG表示Noun-Group，VG表示Verb-Group）：

- NG(Company/ies) VG(Set-up) NG(Joint-Venture) with NG(Company/ies)
- VG(Produce) NG(Product)

对于第二个句子，除了生成上述的第二个模式以外，还生成了下面的两个模式：

- NG(Company) VG-Passive (Capitalized) at NG(Currency)
- NG(Company) VG(Start) NG(Activity) in/on NG(Date)

图15.10中给出的是这两个句子的处理结果：5个初步的模板的集合。然后这5个模板必须合并成图15.7所示的单一层次结构。合并算法决定是否两个行为（activity）或关系（relationship）的结构一致到足以认定它们所描述的是同一事件，如果是同一事件，就并合它们。因为合并算法必须首先实现指代消解（确定什么情况下两个描述所指的是相同的事件），我们只好将对这一阶段的介绍推迟到第18章。

(1)	Relationship: Entities:	TIE-UP "Bridgestone Sports Co." "a local concern" "a Japanese trading house"
(2)	Activity Product	PRODUCTION "golf clubs"
(3)	Relationship: Joint Venture Company Amount	TIE-UP "Bridgestone Sports Taiwan Co." NT$20000000
(4)	Activity Company Start Date	PRODUCTION "Bridgestone Sports Taiwan Co." DURING: January 1990
(5)	Activity Product	PRODUCTION "iron and "metal wood" clubs"

图15.10 FASTUS系统的第五阶段产生的5个局部的模板。这些模板将通过第六阶段的合并算法生成图15.7所示的最后的模板

我们本节所描述的这类特定领域的模板也已被用于许多限定领域的语义理解和话语理解的应用，包括问答系统中的混合对话管理(Bobrow et al., 1977)。

15.6 小结

本章探讨了句法驱动的语义分析的思想。下面是本章的主要内容：

- 语义分析（semantic analysis）是生成意义表示并将这些意义表示指派给语言输入的一种处理。
- 利用词典和语法中静态知识的语义分析器（semantic analyzer）能够生成上下文无关的字面意义或传统意义。
- 组合性原则（principle of compositionality）说明一个句子的意义可以由它的几个部分的意义组合而成。
- 在句法驱动的语义分析（syntax-driven semantic analysis）中，部分（part）是指一个输入的句法成分。
- 通过一些符号扩展，包括λ表达式（λ-expression）和复杂项（complex-term），可以合成FOPC公式。
- 自然语言中的量词（natural language quantifier）带来一种很难通过组合方法处理的歧义。复杂项可以简洁地转译这种歧义。
- 惯用语（idiomatic language）不遵守组成性原则，但是通过设计特定的语法规则和附加语义等技术，就可以轻而易举地处理惯用语的问题。
- 实用语义分析（practical semantic analysis）系统常常通过几种方式来调整严格的组合方法。

- 基于语义语法（semantic grammar）的对话系统依靠的是能够满足语义需求的语法，而不是满足句法概括性的语法。
- 基于层叠式自动机的信息抽取（information extraction）系统能够在抽取输入中相关信息的同时忽略无关的部分。

15.7 文献和历史说明

前面曾提及，组合性原则传统上归功于Frege，Janssen(1997)讨论了这个归属问题。利用第12章介绍过的范畴语法框架，Montague(1973)证实组合方法可以系统地应用于自然语言的有趣的片断。规则到规则（rule-to-rule）的假设是Bach(1976)首次阐述的。从可计算的系统看，Woods的LUNAR系统(Woods, 1977)是一个基于管道流的句法优先的组合分析系统。基于Gazdar的GPSG方法(Gazdar, 1981, 1982; Gazdar et al., 1985)，Schubert and Pelletier(1982)开发了一个增强的规则到规则的系统。Main and Benson(1983)将Montague的方法扩展到问答领域。

作为不同领域对同一问题平行发展的示例之一，为便于辅助编译器的设计，在程序语言领域的研究者开发出了本质上相同的组合性技术。特别是，Knuth(1968)提出的属性语法的概念使得语义结构和句法结构建立起了一一对应的关系。因此，本章所用的语义附着的风格对于使用YACC风格(Johnson and Lesk, 1978)的编译器工具的用户来说是非常熟悉的。

语义语法归功于Burton(Brown and Burton, 1975)。在同一时期，类似的思想包括语用语法(Woods, 1977)和性能语法(Robinson, 1975)。它们都是关于如何重塑句法语法以满足语义处理的需要的。用于受限领域的大部分现代语法都采纳了语义语法的某些形式，这一点毋庸置疑。

在15.2节中给出的用于英语片断的大部分技术改编自SRI的核心英语工具(Alshawi, 1992)。其他零碎的技术改编自Woods(1977)，Schubert and Pelletier(1982)和Gazdar et al. (1985)。本章没能涵盖许多重要的主题，这是不可避免的。关于处理出现远距离依赖时语义理解的标准串孔分析（gap-threading）方法，可以参阅Alshawi(1992)。ter Meulen(1995)给出了对时态、体和时间信息表示的一种现代方法。在Hobbs and Shieber(1987)和Alshawi(1992)中，可以找到有关量词辖域的各种处理方法。van Lehn(1978)给出了对量词辖域的一组优先选择的资料。多年来，有许多针对复合名词理解的成果。语言学方面的资料包括Lees(1970)，Downing(1977)，Levi(1978)和Ryder(1994)；描述计算方法的资料有Gershman(1977)，Finin(1980)，McDonald(1982)，Pierre(1984)，Arens et al.(1987)，Wu(1992)，Vanderwende(1994)和Lauer(1995)。

有关成语的文献历史久远，内容广博。Fillmore et al.(1988)描述了一种称为构造语法（Construction Grammar）的通用语法框架，成语被置于该基础理论的中心。Makkai(1972)对许多英语成语给出了全面的语言学分析。针对第二外语的学习者也已出版了许多成语词典。从计算的角度看，Becker(1975)首先提出在剖析器中使用成语规则。Wilensky and Arens(1980)首先将这一思想成功地用于PHRAN系统。Zernik(1987)演示了一个可以从上下文中学习这种短语成语的系统。Fass et al.(1992)是有关成语计算方法的论文集。

信息抽取的开创性工作是在Frump系统(DeJong, 1982)的背景下展开的。以后的工作受到美国政府资助的MUC会议(Sundheim, 1991, 1992, 1993, 1995b)的激励。Chinchor et al.(1993)介绍了MUC-3和MUC-4会议中用到的评测技术。Hobbs(1997)把FASTUS系统的灵感部分归功于麻省理工学院的CIRCUS系统(Lehnert et al., 1991)的成功。SCISOR系统是另一个在MUC-3(Jacobs and Rau, 1990)中表现突出的基于宽松层叠和语义预期的系统。因为信息抽取系统缺少从一个领域到另外的领域的

重用性，许多研究工作集中于领域知识的自动获取。在Cardie(1993), Cardie(1994), Riloff(1993), Soderland et al.(1995), Huffman(1996)和Freitag(1998)中介绍了许多监督学习方法。

最后，我们还需要说明，本章跳过了语义分析中关于通过深层意义表示的预期来驱动分析处理的整个分支。这类系统避免了直接表示和使用句法，也几乎用不到类似剖析树这样的知识。在这条研究分支中，最早和最成功的系统包括Simmons(1973, 1978, 1983)和Wilks(1975a, 1975c)。Roger Schank和他的学生开发了一些类似的方法(Riesbeck, 1975; Birnbaum and Selfridge, 1981; Riesbeck, 1986)。在这些方法中，语义处理是通过与单独的词典项相关联的逐条处理程序来引导的。CIRCUS系统(Lehnert et al., 1991)的根源可以追溯到这些系统。

第16章　词汇语义学

"When I use a word", Humpty Dumpty said in rather a scornful tone,"it means just what I choose it to mean - neither more nor less."

Lewis Carroll, *Alice in Wonderland*[①]

How many legs does a dog have if you call his tail a leg?
Four.
Calling a tail a leg doesn't make it a one.

Attributed to Abraham Lincoln[②]

前两章的内容着重论述了整个句子的意义表示。在以前的讨论中，我们很少使用词的意义这一概念。我们对词及其意义之所以有兴趣，只是因为它们恰好提供了生成整个句子的足够意义表示所必需的零碎素材。这种普遍的态度源于下面的观点：尽管单词可能促成句子的意义的内容，但它们本身并没有意义。也就是说，认为单词本身不指向世界，不能判断真或假，不能判断字面意义或象征意义，也不能作为通常整个句子和话段所保留的事物的宿主。这种在语义理论中关于单词角色的狭隘观念，导致了将词典看成是一个缺乏任何语义结构的简单符号片断的列表的观点。

本章正是为了说明这种狭隘的观点错失了许多精彩内容。我们将看到词典呈现出的高度系统化的结构，正是这种结构决定了单词的意义和使用。这种结构是由词和其意义之间的关系以及单个词的内部结构组成的。对这种系统化的、与意义相关的结构的语言学研究称为词汇语义学（Lexical Semantics）。

在继续介绍之前，让我们首先引入一些新的术语，因为迄今为止用过的这些术语都过于模糊。特别是单词 *word*，目前已有各种各样的用法，这增加了本章澄清其用法的难度。因此，作为一种替代方式，我们将重点关注词位（lexeme）的概念，它表示词典中一个单独的条目。词位应该被看成是一个特定的正字法（拼写正确的）形式和音素形式与一些符号的意义表示形式的组合。因此，词典（Lexicon）是有限个词位的列表。注意，这个定义允许复合名词和其他非组合短语作为词典的条目。我们将使用术语"正字形"（orthographic form）和"音素形"（phonological form）来指称一个词位的正确形式部分，用术语涵义（sense）来指称一个词位的意义部分。

有了上述的最小术语体系，让我们回到词位中与意义主题相关的一些事实的发现。最初，我们探索的最佳地点是词典，因为词典存在的目的就是作为词位意义信息的储藏室。而在词典里，首先应该查看的最有趣的地方其实是那些实际上从没有人查看的词位的定义。例如，下面从 *American Heritage Dictionary*(Morris, 1985)摘出的关于 *right*，*left*，*red* 和 *blood* 的定义的片断：

---

① 译文如下，供读者对照参考：
"我用一个词，总是与我想要说的恰如其分，既不重，也不轻。"矮胖子相当傲慢地说。

② 译文如下，供读者对照参考：
如果把一只狗的尾巴当成是一条腿，那么它总共有几条腿？
4条。
把尾巴称为腿，并不会因此多了一条腿。

right　*adj.*　located nearer the right hand esp. being on the right when facing the same direction as the observer.

left　*adj.*　Located nearer to this side of the body than the right.

red　*n.*　the color of blood or a ruby.

blood　*n.*　the red liquid that circulates in the heart , arteries and reins of animals

这些定义里最引人注目的现象是存在令人惊讶的大量迂回。*right* 的定义里有两处直接指向本身，而 *left* 的条目包括一个隐含的自引（self-reference）定义：短语 *this side of the body*，估计表示左侧（*left*）。虽然 *red* 和 *blood* 的条目里没有这种直接的自引定义，但是在它们的定义里都提及了对方。当然，这种迂回在词典的所有定义中都存在，这只是几个极端的例子。最终所有定义都是由依次被其他词位定义的词位来说明。

从一个纯粹的规范观点来看，这种内在的迂回表明词典的条目常常不能称之为定义，不过是用其他词位来描述这个词位而已。只有当词典的用户有足够的能力理解这些组成被查询条目的术语时，这样的条目才可能有用。对于像 *red* 和 *right* 这样显而易见的词位，这种方法将失败，因为最终都没能与外部世界相关联。

幸运的是，尽管存在这些不足，在这类定义中仍然包含着丰富的语义信息。例如，上面的定义清楚地表明 *right* 和 *left* 是类似的词位，它们彼此处于相反或交替的地位。类似地，我们可以发现 *red* 是一种可以用于 *blood* 和 *ruby* 的颜色，而 *blood* 是液体。在本章中将会看到，一旦有了足够大的关于类似事实的信息库，在许多应用中就能够实现复杂的语义任务（即使不能真正理解 *right* 和 *left*）。

总之，通过分析和标记在不同场景下词位之间的关系，可以捕捉到不少单个词位的语义。我们特别感兴趣的有两点，一点是对类似场景下不同词的相同点和不同点的刻画，另一点是在同一场合下对不同词位之间关系的刻画。而对后者的反思使我们产生了以下的思想：词位并不是不可分析的原子符号，而是有内部结构的，这种结构部分地决定了它们在与其他词组合以形成句子时的方式。接下来，在16.4节中将进一步探讨词典的创造性或生成性的概念，包括：词典不应该被看成是一个有限的列表，而应该是创造无限意义的生成器。

在继续讲述之前，我们还要指出，这里给出的词汇语义学的观点既不是纯粹语言学的，也不是只针对改进具有较大限制的“仅句子带有意义”的计算应用，而是面向任何需要利用单词并且通过单词的语义知识可以得以改进的应用。因此，本章的重点在于能够捕获词汇语义信息的计算方法，这些方法可以在许多计算应用中发挥作用。

16.1　词位及其涵义之间的关系

本节介绍词位之间以及词位的涵义之间的各种关系。这里所讲述的关系决不是毫无遗漏的，重点在于那些带有显著的计算本质的关系。正如读者将看到的，我们用到的最基本的工具与在一些场景下将一个词位用另一个词位系统地进行替换有关。这种替换的结果可以揭示出替换的词位之间所具有的或所缺乏的特定关系。

16.1.1　同形关系

本节首先讨论同形关系（homonymy），这也许是词位之间最简单并且从语义角度我们最不感兴趣的关系。从传统上讲，同形关系被定义为一种存在于那些词形（包括字形和音形）相同但意义不同的词之间的关系。具有这种关系的词被称为同形（同音）异义词（homonym）。同形关系的一个经典的例子是 *bank*，它有两个截然不同的意义：金融机构和倾斜的堤岸，参见WSJ中的如下例句：

Instead, a *bank* can hold the investments in a custodial account in the client's name. （16.1）

But as agriculture burgeons on the east *bank*, the river will shrink even more. （16.2）

沿袭词典编纂的传统，我们用在词的正字形上添加上标的方式来表示这种关系，比如bank[1]和bank[2]。这个符号表示它们是两个不同的词位，意义截然不同，也没有关联，只是它们碰巧共用一个正字形。

在以下的讨论中，我们最关注的是同形关系的定义对于识别和刻画这些词位（它们在计算应用中会导致歧义问题）有多少帮助。下面开始对例子*bank*进行详细的探讨。首先，我们注意到bank[1]和bank[2]无论是字形还是音形都是相同的。当然，也有一些词位具有截然不同的意义，但只是字形或音形相同。例如，诸如*wood*和*would*，*be*和*bee*这种词位偶对，虽然发音相同，但是拼写不同。当然，传统上这种例子被认为不是完全的同形关系。同形关系的概念与词典编纂学领域有关，在该领域中只有那些具有相同的引用形式（citation-form）的词条才可能具有同形关系。引用形式是在词典中用于按字母顺序索引单词的正字形式，在英语中对应于我们常说的词根形式。根据这种观点，具有相同发音或不同拼写的单词就不能称为同形词，而是应该称为同音异义词（homophone），不同词位发音相同。

也存在相同字形但不同发音的词位偶对。例如，下面例子中的正字形*bass*有两种完全不同的意义，一种是鱼，而另一种是乐器：

The expert angler from Dora, Mo., was fly-casting for bass rather than the traditional trout. （16.3）

The curtain rises to the sound of angry dogs baying and ominous bass chords sounding. （16.4）

尽管这些例子完全符合传统的同形关系的定义，但是这类词很少出现在任何传统的同形异义词的列表上。人们将相同正字形但意义无关的词位称为同形异义词（homograph）。

最后要指出的是，不同词性的词位通常也不被认为是完全的同形异义词。除正字形的原因之外，利用这个限制就可以排除*would*和*word*这样的例子。做出这样的限制的原因有两个：首先，在我们讨论词性标注时，基于不同的句法环境可以轻易区别出这种不同词性的词位；其次，基于屈折和派生的形态变化（大部分与词性相关），词典项可以呈现出多种不同的形式。

更复杂的是，形态不同却具有相同词形的情况也可能出现在相同词性的词位之间。例如，在下面的WSJ例句中，词位*find*和*found*分别表示的是“查找”和“建立一个机构”的意义：

He has looked at 14 baseball and football stadiums and found that only one-private Dodger Stadium-brought more money into a city than it took out. （16.5）

Culturally speaking, this city has increasingly displayed its determination to found the sort of institutions that attract the esteem of Eastern urbanites. （16.6）

这两个词位具有不同的词根形式，*find*和*found*，然而形态变体*found*是*find*的过去时，也是*found*的词根。

前面曾经指出，人们之所以对同形关系产生兴趣，只是因为它在应用中导致了歧义。而一对词位是否在应用中引起歧义完全取决于该应用的特性。关键问题在于形式上重叠的特性是否容易对给定的应用造成很大的困难。

在拼写校正（spelling correction）中，同音词可能导致真词的拼写错误，或用词错误，比如词位*weather*和*whether*的互换。在这个例子中，由于发音的重叠将导致完全基于文本的系统出现问题。拼写校正中的另一个问题是由类似*find*和*found*的具有部分形态变化的重叠的不完全的同形异

义词导致的。在本例中，像*founded*这样的词形可能表示一个正确使用的过去式，也可能是对不规则动词错误地使用了规则动词的过去式规则。

在语音识别（speech recognition）中，比如*to*, *two*和*too*这样的同音词，会产生很明显的问题。然而，其实像*bank*这样的完全的同形异义词也会导致问题。回顾前面讲过的，依赖语言模型的语音识别系统常常基于N元概率表。对完全的同形异义词来说，尽管它们作为不同的词位出现在不同的语言环境中，但是所有这些不同词位的条目都被合并在一起。这种合并一方面导致该词位与一些非同现的词的相关概率上升，另一方面也导致该词位与同现词的相关概率降低。

当然，文语转换（text-to-speech）系统最容易受到那些发音不同的同形词的影响。这个问题可以通过使用词性标注来避免，比如*conduct*，它的不同发音对应于不同词性。然而，对于*bass*这样的例子，就必须通过其他手段来区分。注意，这种情形与拼写校正是相反的，这里，本质上面向语音的系统正受到正字形问题的困扰。

最后，正如我们将在第17章中详细讨论的，一个信息检索系统的性能在出现同形异义词时将会下降。寻找海水动物*bass*的用户显然不会对讲述乐器的文章感兴趣。

16.1.2　多义关系

让我们继续讨论两个意义相关或无关的涵义。在定义同形关系时，要求所涉及的词位对具有截然不同并且无关的意义。这一点相当关键，如果所涉及的意义之间有关联，则我们处理的是具有多个意义的单个词位，而不是两个独立的词位。单个词位具有多个相关涵义的现象被称为多义关系（polysemy）。注意，我们曾经把词位定位为一个表面形式和一个涵义的组合。这里将这个定义扩展为：一个形式与多个相关涵义的组合。

为使这个概念更具体，研究WSJ语料库中*bank*的如下例子：

While some *banks* furnish sperm only to married women, others are much less restrictive.　（16.7）

显然，在该句中*bank*的意义不是“倾斜的堤岸”，也很明显的是，*furnish sperm*不可能是指一个金融机构里促销的赠品。而且，bank有一批与各种生物实体的存储有关的涵义，比如*blood bank*，*egg bank*和*sperm bank*等。对这种用法的一种处理方法就是创造出$bank^3$，即对bank增加一个不同的词位，使它的意义指向这种“精子库”（sperm bank）的用法。遗憾的是，根据我们对同形关系的定义，这需要本例中*bank*的意义与金融机构的涵义截然不同并且无关，但这种看法似乎不太容易被接受。多义关系的概念允许我们将*bank*的这个涵义理解为与金融机构的涵义相关，而且很可能是派生出来的涵义，而不需要强调这是一个独立的词位。

正如人们所觉察到的，同形关系与多义关系的区分并不都像*bank*这类例子那样直截了当。为了确定两个词位的意义是相关还是无关的，通常我们需要援引两个标准：一个是所涉及的词位的历史，或称为词源学；另一个是说本族语的人如何用这些词位进行构思。在实践中，从这两个标准获得的证据常常不很清晰地组合起来，以便于从词典条目中的多义词里区分出同形异义词。在*bank*这个例子中，词源学表明$bank^1$来自意大利（Italian），而$bank^2$来自斯堪的纳维亚（Scandinavian，瑞典、挪威、丹麦和冰岛的泛称），这鼓励我们将它们列为不同的词位。另外，我们认为例句（16.7）中*bank*的用法与bank1是相关的，一方面是基于对它们的意义类似性的思考，另一方面是由于我们也缺乏存在第三个独立涵义的任何词源学的证据。

当缺乏详细的词源学证据时，一个有用的直觉可以用来区分同形关系和多义关系，这个直觉就是巧合的概念。同形关系通常很容易被理解为历史上的巧合——两个词位碰巧共用一个形式。而多义关系的例子就很难看成是巧合。再回到*bank*的例子，人们很难接受这样的思想：*bank*的所有各种储藏室涵义的各种使用仅仅碰巧与存款机构的涵义相关。

一旦确定我们处理的是一个多义词位，很自然就需要设法管理它的潜在的众多涵义。特别是对于任何给定的单个词位，都应该能够回答下面的问题：

- 它有哪些截然不同的涵义？
- 这些涵义是怎样关联的？
- 怎样才能可靠地区分这些涵义？

这些问题的回答对于语义分析器、搜索引擎、生成器和机器翻译等系统在它们各自的任务中的性能优劣都有很重要的影响。前两个问题将在本节和16.4节中讲述，而最后一个问题将在第17章中详细给出。

确定一个特定的多义词位应该有多少种涵义是一个长期以来让词典编纂学家（直到最近惟一从事大型词典信息库建设的人群）烦恼的问题。大部分词典编纂学家所用的方法是：生成足够多的词条，以便于解释在一些大型例句语料库中所观察到的所有意义的细小差别。假如传统词典的根本用途是帮助用户学习词的各种用法，那这就是一个非常合理的方法。然而，遗憾的是，由于细小差别超过了任何合理计算应用通常所需的程度，因此非常容易导致错误。

为了使区分不同涵义的概念更具体化，研究下面WSJ语料库中有关动词*serve*的例句：

They rarely *serve* red meat, preferring to prepare seafood, poultry or game birds. (16.8)

He *served* as U.S. ambassador to Norway in 1976 and 1977. (16.9)

He might have *served* his time, come out and led an upstanding life. (16.10)

我们可以得出一个合理的论点：每个例句都代表*serve*的一种不同涵义。例如，第一个例子中隐含*的serving red meat*与*preparing seafood*之间的对照表明，在*serve*的涵义与相关的食物准备（food preparation）的概念之间存在着很强的联系。因为在其他例子中都不含有类似的成分，因此可以假定第一个用法与其他两个有明显的不同。接着，我们注意到第二个例子所带有的句法次范畴与其他例子不同，因为它的第一个论元（表示主语的角色）是一个介词短语。我们将在16.3节中讨论，这种句法状态的不同常常表明本质涵义的不同。最后，第三个例子与特定的监狱领域有关。尽管在这个例子中几乎没有提供任何关于监狱的特别信息，但是仍然有一个明显而清楚的意义，而且这个意义在其他例子中无法担当角色。

用于确定现有的两个涵义是否不同的另一项实用技术是：利用连词将两个分开使用的词位放入一个例子中，这种技术有一个相当不好理解的名字：共轭搭配法（zeugma）。研究下面ATIS中的例子：

Which of those flights serve breakfast? (16.11)

Does Midwest Express serve Philadelphia? (16.12)

?Does Midwest Express serve breakfast and Philadelphia? (16.13)

我们在例句的开头处用?表示这个句子在语义上是有某种错误的。创造出来的第三个例子的怪异，表明不存在一种切实方式能让*serve*的单个涵义同时满足*breakfast*和*Philadelphia*的要求。更准确地说，在第一个例子中*serve*所援引的潜在概念不能以任何有意义的方式应用于*Philadelphia*。这个例子表明，我们可以利用语料库中的例句和我们本族语人的直觉，用一种构造方法来揭示不同涵义的存在。

对给定的词位找出恰当涵义集的问题与确定在给定的例子中应该使用词位的哪个涵义的处理完全不同。后者被称为词义排歧（word sense disambiguation），将在第 17 章中详细讨论。在词义排歧中，一般假定每个词典项具有一个固定的涵义集，在 16.4 节中将介绍一个使用不确定涵义集的建议。注意，在实践中这个任务常常是指从预先定义好的涵义集中选出符合词形的上下文的一个涵义的任何处理。换句话说，词义排歧的大部分方法并不有意识地区分同形关系和多义关系。正如在第 17 章中将会看到的，这暗示了这些系统是如何被评价的。

最后，让我们简单讨论一下单个多义词位的各种涵义之间的相关性这一话题。早先我们提出了直觉的概念，即词位的多个涵义不可能因巧合而产生。显然，我们接下来就会提问：如果它们不是碰巧联系在一起的，那么是如何联系在一起的？那些大型词典的编纂者对这个问题并不特别关心，因为只要词典包含这些正确的涵义，至于它们是怎么来的则无关紧要。但是，只要一个应用系统开始尝试处理广泛的各种输入，立刻就会遇到与系统词典中的任何静态涵义都不同的新用法。而通过研究所列涵义中的系统化关系，就可以获得处理这类新用法的意义的能力。这种思想将在16.4节中详细讨论。

16.1.3　同义关系

同义关系的现象非常普遍，从辞典和纵横填字谜的流行就能说明这一点。与同形关系一样，同义关系（synonymy）的概念也具有一个欺骗性的简单定义：具有相同意义的不同词位。自然，在这个定义中并没有说明两个词位表示同一事物意味着什么？我们在16.3节中将给这个问题提供一些答案，目前先不直接回答这个问题，而是通过援引可替换性（substitutability）的概念来继续讲述。可替换性的概念是指：如果两个词位可以在句中相互替换，而不改变句子的意义和可接受性，那么就认为这两个词位是同义词。下面的 ATIS 例子说明了可替换性的概念：

How big is that plane?　（16.14）

Would I be flying on a large or small plane?　（16.15）

在这些例子中，交换 *big* 和 *large* 不会明显影响句子的意义和可接受性。我们可以将它作为 *big* 和 *large* 是同义词的证据，即至少对这些例子来说它们是同义词。注意，这是一个范围非常狭窄的声明。特别是，我们没有提及任何类似于这些例子的 *big* 和 *large* 相对容易出现的上下文环境。

如果我们将可替换性的概念用于表示在所有可能环境中可替换，那么在英语中真正的同义词就非常少，更不必说我们总能发现一些句子，在这些句子里一个传说中的同义词却不能成功地进行替换。在这种情况下，我们回到一个比较实用的概念：如果两个词位在某些（*some*）环境下能够替换，就称这两个词为同义词。这种以实用为目的的概念是大多数词典和辞书中同义词的概念。

在一个指定的场景下给定的同义词候选对能否成功地进行替换，本质上取决于 4 种因素的影响：多义关系、微妙的意义背景、搭配约束和使用域。只有前两个因素涉及意义的概念。

为了探究多义关系对替换性的影响，考虑 WSJ 中的如下例子，在这个例子中 *large* 和 *big* 的替换很明显是不成功的：

Miss Nelson, for instance, became a kind of big sister to Mrs. Van Tassel's son, Benjamin.　（16.16）

? Miss Nelson, for instance, became a kind of large sister to Mrs. Van Tassel's son, Benjamin.（16.17）

失败的原因在于词位 *big* 的多义涵义中存在一个年长的、长大成人的概念这一事实。因为在 *large* 的许多涵义中缺少这个涵义，所以它不能在需要这个语义的环境下替换 *big*。在本例中，所生成的句

子是一个与原句意义不同的句子。在其他情况下，这种替换也可能导致一个句子奇奇怪怪或完全无法理解。

影响同义关系的下一个因素是微妙的意义背景。我们所指的是这样一种情况，两个词位共用一个核心意义，但其他附属事实是与其中一个词位相关的。探讨ATIS语料库中词位*price*和*fare*的用法。从语义角度，它们的核心意义都表示服务的费用。但是，它们不能自由交换。研究下面的ATIS例句：

What is the cheapest first class fare? (16.18)

? What is the cheapest first class price? (16.19)

在本例中用*price*来替换*fare*会导致句子有些古怪。这种古怪的来源很难查明，但是*fare*似乎更适合于表示各种服务的费用（比如coach，business和first class fares），而*price*似乎更适合应用于代表这类服务的票据（ticket）。当然，在该领域对这些词位的用法的更全面解释需要对语料库中的相关例子进行系统的分析。其关键是，尽管这些词共用一个核心意义，但是一些细微的与意义有关的不同会影响它们的使用。

替换性的这两个影响因素都清楚地涉及到词典项的意义。然而，也存在对同义词替换成功与否有影响而与意义没有任何直接关联的其他因素。搭配约束就是这样的影响因素之一，通过一个搭配约束，我们指那种能够通过一些技术（比如第6章的N元模型）捕捉到的词典项之间的联系或吸引力。研究下面的WSJ的例子：

We frustrate 'em and frustrate'em, and pretty soon they make a big mistake. (16.20)

? We frustrate 'em and frustrate'em, and pretty soon they make a large mistake. (16.21)

在本例中，当表示错误（mistake）的紧要性或重要性时，我们更倾向于使用*big*而不是*large*。这不能归因于多义关系的不同，也不能归因于*big*和*large*之间存在任何微妙的背景意义差异。注意，这与*large sister*的例子完全不同，*a large mistake*仍旧可以正确理解，只是用*large*不如*big*那么自然。

最后，使用域，我们是指那些围绕可能的同义词用法的社会因素。这里所指的是那些具有相同的核心意义，但由于一些因素，比如礼貌、社团地位和其他类似社会压力，不能在所有环境下替换的词位。例如，当需要技术或学术的风格时，拉丁源和希腊源的多音节词位常常取代较少音节的词位。

与同形关系中的情形一样，这些同义关系的影响因素在实践中对计算应用具有不同的影响。在第19章和第20章中，将会看到在自然语言生成和机器翻译中意义的类似性、搭配约束和使用域非常重要。而在信息抽取和信息检索领域中，使用的正确性远不及意义识别的概念重要。

16.1.4 上下位关系

在讨论*price*和*fare*时，我们引入了具有类似但不相同的意义的词位对的概念。上下位关系（hyponymy）的概念是基于一种词位对的有限等级，在这种词位对中一个词位是另一个词位的次类。例如，*car*（小汽车）和*vehicle*（交通工具）之间的关系就是一种上下位关系。因为这种关系不对称，所以我们把特定性较强的词位称为概括性较强的词位的下位词（hyponym）。相反地，把概括性较强的词位称为特定性较强的词位的上位词（hypernym）。因此，我们可以说car是vehicle的下位词，而*vehicle*是*car*的上位词。

与同义关系一样，我们也可以利用一个受限的替换来探讨上下位关系的概念。思考下面的方案：

This is a x. ⇒ That is a y.

如果x是y的下位词，则在任何情形下，当左边的句子为真时，右边新产生的句子也必须为真，例如：

This is a car. ⇒ That is a vehicle.

这种受限的替换和在同义关系中讨论过的那种替换有许多重要的不同。在同义关系中生成的句子似乎可以看成是对原句的替换，而新生成句子的目的并不是作为原句的替换，而是仅仅作为对是否存在上下位关系的一种诊断测试。

上下位关系的概念与生物学、语言人类学和计算机科学中的许多重要概念非常接近。

术语“本体知识体系”（ontology）通常是指对一个领域或微世界（microworld）进行分析而获得的不同客体的集合。分类体系（taxonomy）是指把本体知识体系中的元素排列成包含结构的树状分类的一种特别方式。通常，对那些超出成分类的包含关系的分类，存在一系列良构的限制。例如，词位 *hound*（猎犬）、*mutt*（杂种狗）和 *puppy*（小狗）都是 *dog*（狗）的同义词，但如果由它们来构成一个分类体系，就显得很古怪，因为每个词的用于激发关系的概念都是不同的。最后，计算机科学里的对象层次（object hierarchy）基于这样的概念：本体知识体系中的对象被安排在一个分类体系中，并能够接受或继承分类体系中它们的祖先的特征。当然，只有当分类体系中的元素事实上是带有可继承特征的复杂结构的对象时，才会有意义。

因此，上下位关系本身并不能组成一个本体知识体系、范畴结构、分类体系或对象层次。然而，已经证实它们作为这类结构的近似是有用的。

16.2　WORDNET：词汇关系信息库

词汇关系在语言学、心理语言学和计算研究中的作用激发了许多针对这种关系建立大型电子信息库的工作。通常，建立这类信息库的工作都是按照下面两种基本方法中的一种进行的：或者从现有词典或辞书中挖掘信息，或者东拼西凑地手工建立一个信息库。尽管重用现有资源的方法具有明显的优势，最大规模和最广泛使用的英语词汇信息库 WordNet，却是手工建成的(Fellbaum, 1998)。

WordNet 由 3 个独立的信息库组成：名词库、动词库以及形容词和副词共同的库。WordNet 中不包括封闭词类的词汇项。每个库都由一组对应于惟一正字形的词汇条目与每个字形相关的一组涵义构成。图16.1给出的是1.6版WordNet中的词汇规模。这个信息库可以通过浏览器（本地或Internet）直接登录，也可以使用一组 C 库函数通过程序的方式登录。

范畴	惟一词形	涵义数
名词	94 474	116 317
动词	10 319	22 066
形容词	20 170	29 881
副词	4546	5677

图 16.1　当前 WordNet 1.6 版中词汇的规模，依据的是 4 个信息库中的惟一条目和总的涵义数

WordNet涵义条目的最完整形式由一组同义词、一个词典风格的定义或注释和一些用法示例组成。图 16.2 中所示的是 WordNet 中名词条目 *bass* 的简化版。从这个条目可以看出，WordNet 的条目和词位的概念有几个重要的不同。首先，WordNet不包括发音信息，因此并不区分不同发音的词位。例如，本条目的 $bass^4$，$bass^5$ 和 $bass^8$ 都是鱼的涵义，发音为[b ae s]，而其他都是音乐的涵义，

发音为[b ey s]。更概括地讲，在 WordNet 中并不区分同形关系与多义关系。例如，就这个条目而言，bass[1] 和 bass[2]，bass[1] 和 bass[4] 所具有的关系并没有差别。这种保守的处理策略反映出，尽管我们对词的那些截然不同的涵义具备相当可靠的诊断，但是系统地组织这种多义涵义仍是一种不确定和主观的行为。正因为如此，WordNet 的开发者选择简单地列出这些不同的涵义，并不将它们清楚地组织成我们在许多词典中见过的那种层次结构的方式。

The noun "bass" has 8 senses in WordNet.
1. bass - (the lowest part of the musical range)
2. bass, bass part - (the lowest part in polyphonic music)
3. bass, basso - (an adult male singer with the lowest voice)
4. sea bass, bass - (flesh of lean-fleshed saltwater fish of the family Serranidae)
5. freshwater bass, bass - (any of various North American lean-fleshed freshwater fishes especially of the genus Micropterus)
6. bass, bass voice, basso - (the lowest adult male singing voice)
7. bass - (the member with the lowest range of a family of musical instruments)
8. bass - (nontechnical name for any of numerous edible marine and freshwater spiny-finned fishes)

图 16.2 WordNet 1.6 中名词条目 *bass* 的部分内容

图16.3给出了整个动词库中的大致涵义分布。WordNet中的所有动词根据所具有的涵义数目被分级。从图 16.3 可以看出，该分布是极端不对称的，少数条目具有大量的涵义，而大多数条目只具有单个涵义。当处理词典时普遍存在这种分布，称为Zipf分布(Zipf, 1949)。动词库的多义程度大于名词库。这与英语中的动词数远少于名词数，但它们的意义却更具有可塑性的事实是一致的。

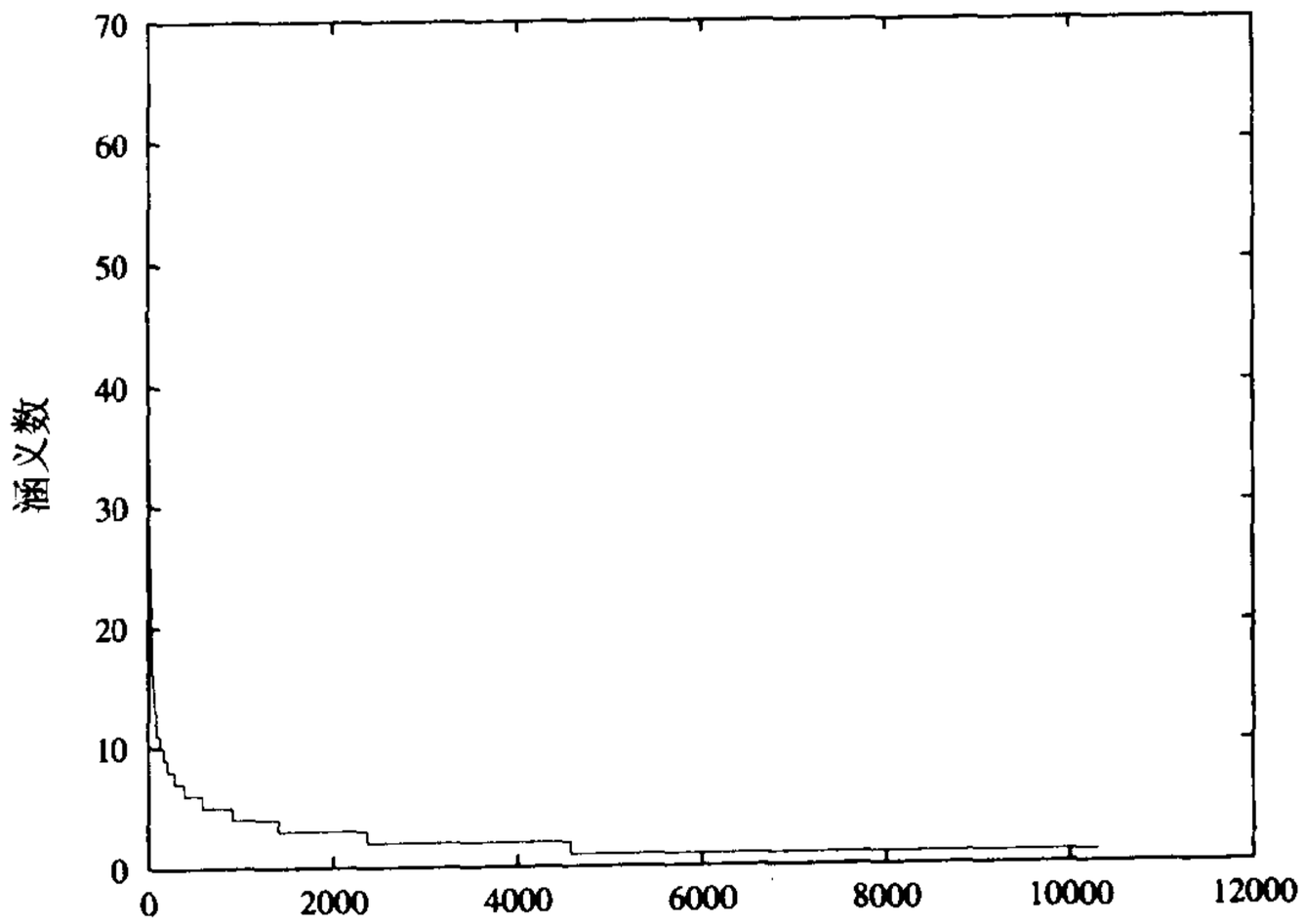


图 16.3 WordNet 中动词的涵义分布。*y* 轴表示每个 WordNet 条目的涵义数，*x* 轴按动词的多义程度将它们分级。从图中可以看出，少数动词条目具有很高的多义程度，而大多数条目只具有单个涵义

当然，词汇条目的简单列表并不比普通在线词典有更多用途。WordNet的影响力在于它的领域无关的词汇关系。这些关系存在于WordNet的条目、涵义和同义词集里。这种关系只限于相同词性的项（对大部分来说），或从更实用的角度讲，只限于处于同一信息库的项。在图 16.4、图 16.5 和

图 16.6 中，分别给出了这三个信息库所对应的关系的子集，并带有简单的定义和示例。对 WordNet 内容的完整讨论超出了本书的范围，因此这里只讨论它的两个最有用和最成熟的特征：它的同义词集和上下位关系。

关系	定义	示例
Hypernym	From concepts to superordinates	*breakfast* → *meal*
Hyponym	From concepts to subtypes	*meal* → *lunch*
Has-Member	From groups to their members	*faculty* → *professor*
Member-Of	From members to their groups	*copilot* → *crew*
Has-Part	From wholes to parts	*table* → *leg*
Part-Of	From parts to wholes	*course* → *meal*
Antonym	Opposites	*leader* ⟺ *follower*

图 16.4　WordNet 中的名词关系

关系	定义	示例
Hypernym	From events to superordinate events	*fly* → *travel*
Troponym	From events to their subtypes	*walk* → *stroll*
Entails	From events to the events they entail	*snore* → *sleep*
Antonym	Opposites	*increase* ⟺ *decrease*

图 16.5　WordNet 中的动词关系

关系	定义	示例
Antonym	Opposite	*heavy* ⟺ *light*
Adverb	Opposite	*quickly* ⟺ *slowly*

图 16.6　WordNet 中的形容词和副词关系

WordNet中同义词的基本原则与上一节的讲述是相同的。如果WordNet中的两个条目在有些上下文环境中能够成功地进行替换，则认为它们是同义词。WordNet同义词的特殊理论和实现都是围绕同义集（synset），即同义词的集合这一概念来组织的。下面是同义集的一个例子：

```
{chump, fish, fool, gull, mark, patsy, fall guy,
sucker, schlemiel, shlemiel, soft touch, mug}
```

该同义集的词典风格的定义或注释说明该同义集表示的是易受骗和易被利用的人（a person who is gullible and easy to take advantage of）。因此，在这个同义集中的每个词条都可以在一些场景下表达这个概念。实际上，WordNet的许多条目的涵义是由这类同义集组成的。具体地说，正是这个原样的同义集及其定义和例句，拼凑成了同义集中所列条目的所有涵义中的一个涵义。

从一个更理论化的观点看，每个同义集都可以表示一个语言中已经词汇化的概念。因此，同义集有些类似于第 14 章讨论过的概念。不过，WordNet 不是用逻辑项来表示概念，而是通过把可用于表达概念的词典条目组成列表来表示概念。这种观点引出这样一个事实：正是同义集，不是词典条目或单个涵义，参与了图 16.4、图 16.5 和图 16.6 中的大部分语义关系。

WordNet中的上下位关系与上一节讨论过的概念直接对应。每个同义集通过上位关系和下位关系与紧靠的更普遍化或更具体化的同义集相关联。为了找到一系列更普遍化或更具体化的同义集，可以简单地跟随一个上位和下位关系的传递链。为了使这个概念更具体，我们来研究图16.7中$bass^3$和$bass^7$的上位关系链。

在这个上下位关系的描述中，更普遍化的同义集在相连的首行缩排行中接连给出。第一条链从低音歌手（bass singer）的概念开始。它的紧靠的上位对应于一个更宽泛的概念：歌手（singer）。沿着这条链最后到演艺人员（entertainer）和人（person）这样的概念。

```
Sense 3
bass, basso --
(an adult male singer with the lowest voice)
=> singer, vocalist
   => musician, instrumentalist, player
      => performer, performing artist
         => entertainer
            => person, individual, someone...
               => life form, organism, being...
                  => entity, something
               => causal agent, cause, causal agency
                  => entity, something

Sense 7
bass --
(the member with the lowest range of a family of
musical instruments)
=> musical instrument
   => instrument
      => device
         => instrumentality, instrumentation
            => artifact, artefact
               => object, physical object
                  => entity, something
```

图 16.7 词位 *bass* 的两个独立涵义的上下位关系链。注意这两个链完全不同，只是会聚于实体

第二条链是一个完全不同的链，从乐器（musical instrument）开始，接着到器具（instrument）、工具（device）和物体（physical Object）。最终两条链都与同义集实体（entity）相接，本质上实体是作为层次体系最顶端的一个占位符的。

16.3 词的内在结构

在讨论了词位及其意义成分之间的关系以后，现在就可以开始探讨应该由什么组成这些意义成分。前两章清楚地讲述了有关意义的方法，其关键概念在于：意义表示是以基本的谓词-论元结构为基础的。在组成这样的表示时，我们假定词位的某些类倾向于促成谓词和谓词-论元结构，而其他的倾向于促成论元。本节详细探讨词位的意义被结构化，以支持这个概念的系统化方法，特别是探讨这样一个概念：词位的各个意义表示之间具有可分析的内在结构，并且正是这个内在结构与语法相结合，才确定了良构句子中词位之间的关系。

16.3.1 题元角色

题元角色（thematic role）是由 Gruber(1965)和 Fillmore(1968)首次提出的[①]。

题元角色是一组范畴，它为刻画动词的某些论元的特征提供了浅层语义语言。例如，思考WSJ中的下面两个片断：

Houston's Billy Hatcher broke a bat. (16.22)

He opened a rawer. (16.23)

① 实际上 Fillmore 将它们称为深层格（deep case），以形态学的格来比喻。

根据第 14 章的谓词演算事件表示，对这两个句子可以得到下面的部分表示：

$$\exists e,x,y\ Isa(e,Breaking) \wedge Breaker(e,BillyHatcher) \wedge BrokenThing(e,y) \wedge Isa(y,BaseballBat)$$

$$\exists e,x,y\ Isa(e,Opening) \wedge Opener(e,he) \wedge OpenedThing(e,y) \wedge Isa(y,Door)$$

在这个表示中，动词 *break* 和 *open* 的主语的角色分别为 *Breaker* 和 *Opener*。每种可能发生的事件的深层角色（deep role）都是特定的，比如 *Breaking* 事件中的 *Breaker*，*Opening* 事件中的 *Opener*，*Eating* 事件中的 *Eater* 等。但是 *Breaker* 和 *Opener* 有一些共同点。它们都是有意志的行为人，常常是有生命的，并且对它们的事件负有直接因果责任。题元角色（thematic role）是表示这种共性的一种方式。我们把这两个动词的主语称为施事（agent）。因此 AGENT 是表示抽象概念（比如意志的因果关系）的题元角色。类似地，两个动词的直接宾语 *BrokenThing* 和 *OpenedThing*，都是受动作的某种影响的无生命的典型宾语。这种参与者（participant）的题元角色就是主题（theme）。

尽管题元角色并没有一个标准的集合，但是在计算系统中有不少常用的角色。例如，从对例句（16.24）的任何直接理解中，可以得知“Cockwell 先生折断了锁骨”，但是并没有任何关于他是这个不幸事件的施事的暗示。这种参与者可以用体验者（experiencer）加以标识，而直接受到影响的参与者，即本例中的“锁骨”（callarbone）仍被指派为 THEME 角色。

A company soccer game last year got to rough that Mr. Cockwell broke his collarbone and an associate broke an ankle.　　（16.24）

在例句（16.25）中，玻璃破碎（glass breaking）的直接原因是地震（earthquake），所以似乎应该是候选的施事角色。然而，这似乎有些奇怪，因为地震并不是那种可以有意识地做任何事的参与者。长期以来，这类例子是各种题元角色理论产生大量争议的源泉。有两种常见的处理方法：仍给地震（earthquake）指派施事（AGENT）角色，但假设预期的意义与核心的有生命的/有意志的AGENT意义之间存在某种比喻的关联；或添加一个角色，称之为自然力量（force），类似于AGENT，但是不包含任何意志的概念。我们将沿用后一种方法，在 16.4 节再回到比喻的概念。

The quake broke glass in several downtown skyscrapers.　　（16.25）

最后，在下面例子中，主语（*It*）指一个事件的参与者（在本例中是其他人的胳膊肘），它的角色在breaking事件中作为一些其他AGENT或FORCE的工具。这类参与者称为工具（instrument）。

It broke his jaw.　　（16.26）

图 16.8 给出了一小部分常用的题元角色以及各自意义的大致描述。图 16.9 对每个角色都给出了例句。注意，这个角色列表决不是完全确定的，也并不针对于任何一个题元角色理论。

题元角色	定义
AGENT	The volitional causer of an event
EXPERIENCER	The experiencer of an event
FORCE	The non-volitional causer of the event
THEME	The participant most directly affected by an event
RESULT	The end product of an event
CONTENT	The proposition or content of a propositional event
INSTRUMENT	An instrument used in an event
BENEFICIARY	The beneficiary of an event
SOURCE	The origin of the object of a transfer event
GOAL	The destination of an object of a transfer event

图 16.8　一些常用的题元角色及其定义

题元角色	定义
AGENT	*The waiter* spilled the soup.
EXPERIENCER	*John* has a headache.
FORCE	*The wind* blows debris from the mall into our yards.
THEME	Only after Benjamin Franklin broke *the ice*...
RESULT	The French government has built a *regulation-size baseball diamond*...
CONTENT	Mona asked *"You met Mary Ann at a supermarket"?*
INSTRUMENT	He turned to poaching catfish, stunning them *with a shocking device*...
BENEFICIARY	Whenever Ann Callahan makes hotel reservations *for her boss*...
SOURCE	I flew in *from Boston.*
GOAL	I drove *to Portland.*

图 16.9 各种题元角色的典型例句

关联理论的应用和浅层语义解释

计算系统中题元角色的常见用途是作为浅层语义语言。例如，在第21章中我们将讲述题元角色有时在机器翻译系统中被当成是一个有用的中间语言的一部分。

题元角色的另一个用途，也是Fillmore(1968)最初动机的一部分，是作为概念结构或常识中的语义角色（比如*Breaker*和*DirvenThing*）以及它们的具体语言的表面语法的实现（比如主语和宾语）之间的中间层。Fillmore注意到存在着典型的模式，决定了应该由是动词的那个论元来担当主动句的主语。为了指派主语角色，提出了下面的层级，现在常称为题元层级（thematic hierarchy）(Jackendoff, 1972)：

AGENT ≻ INSTRUMENT ≻ THEME

（施事）≻（工具）≻（主题）

因此，如果一个动词的题元描述包括一个AGENT，一个INSTRUMENT和一个THEME，则AGENT被实现为主语。如果题元描述只包括一个INSTRUMENT和THEME，则INSTRUMENT成为主语。在确定主动句的直接宾语或被动句的主语时，题元层级按相反的方向来使用。这里是Fillmore(1968)以动词*open*给出的例句：

John *opened* *the door.* (16.27)

AGENT THEME

John *opened* *the door* *with* *the key.* (16.28)

AGENT THEME INSTRUMENT

The key *opened* *the door.* (16.29)

INSTRUMENT THEME

The door *was* *opened* *by* *John.* (16.30)

THEME AGENT

在过去30年里，该方法引导人们在概念结构和语法功能之间的映射方面做了大量广泛的研究工作，这一研究领域通常被称为"关联理论"（linking theory）。例如，许多文献，比如Talmy(1985)，Jackendoff(1983)和Levin(1993)，都指出动词的语义性质可以帮助预测它们能够具有哪些表面的交替（alternation）。交替是概念（深层）角色与语法功能之间的不同映射的集合。例如Fillmore(1965)

和许多后续文献都讨论过“与格交替”（dative alternation），某些动词（比如 *give*，*send* 和 *read*）可以具有一个 AGENT，一个 THEME 和一个 GOAL，允许 THEME 作为宾语出现，GOAL 在介词短语中，见例句（16.31a），或 GOAL 作为宾语出现，THEME 作为第二宾语，见例句（16.31b）：

a. *Doris gave/sent/read the book to Cary.*　（16.31）
AGENT　THEME　GOAL

b. *Doris gave/sent/read Cary the book.*
AGENT　GOAL　THEME

包括 Green (1974)，Pinker (1989)，Gropen et al.(1989)，Goldberg (1995)和 Levin (1993)在内的许多文献，详见 Levin(1993，p.45)。已经证明这种交替出现在动词的特殊语义类型中，包括“未来拥有的动词”（verbs of future having），如 *advance*，*allocate*，*offer* 和 *owe*，“传送动词”（send verb），如 *forward*，*hand* 和 *mail*，“投掷动词”（verbs of throwing），如 *kick*，*pass* 和 *throw*，以及许多其他的类型。

类似地，Talmy(1985)跟随 Lakoff(1965，p.126)指出了“情感”（affect）动词，比如 *frighten*，*please* 和 *exasperate* 等都可以用 THEME 作为主语，如在例句（16.32）中所示，或以 EXPERIENCER 作为主语，并以 THEME 作为介词宾语，如在例句（16.33）中所示。

a. *That frightens me*　（16.32）
THEME　EXPERIENCER

b. *That interests me*
THEME　EXPERIENCER

c. *That surprises me*
THEME　AGENT

a. *I am frightened of that.*　（16.33）
EXPERIENCER　THEME

b. *I am interested in that.*
EXPERIENCER　THEME

c. *I am surprised at that.*
EXPERIENCER　THEME

Levin(1993)总结了 80 个这种交替，包括在每种语义类型中动词的详尽列表，以及语义限制、特例和其他特性。该列表已被用于许多计算模型，例如(Dang et al., 1998; Jing and McKeown, 1998; Lapata and Brew, 1999)。

尽管上面所总结的类型的研究已经表明在动词语义和句法实现之间存在一种关系，但是是否通过具有语义层级的语义角色的小集合来传达这种关系却不很清楚。例如，尽管属于正确的语义类型（比如，*donate*，*return* 和 *transfer*），许多动词并不容许与格交替。另外，正如上面所给出的，许多动词不遵守任何标准语义层级（像 *Ling sent Mary the book* 这样的与格交替，有一个 GOAL 作为直接宾语，紧跟一个间接的 THEME，而 THEME 应该是最好的直接宾语）。

一个更严重的问题是语义角色只能在语义到句法的一般映射中担当很小的角色。这是因为语义角色只与 *NP* 和 *PP* 论元的语法角色的确定有关，并不参与动词的其他论元和其他谓词的实现。我们在第 11 章的图 11.3 中曾描述过许多这种可能的论元，比如句子的补语（*Sfin*，*Swh-*，*Sforto*），动

词短语（*VPbrst*，*VPto*等）或引用（*Quo*）。而且，语义角色只对动词论元的映射起作用，但是其他词类，例如名词也有论元（*destruction of the city*, *father of the bride*）。

对语义角色的这些问题有许多可能的处理方法。许多系统以实用为目的而继续使用它们，比如在机器翻译中作为中间语言或在浅层语义诠释中作为适宜的语义层。其他研究者认为语义角色应该被看成是一种附带的现象，而不是一个独特的表示层。例如，跟随Foley and van Valin(1984)，Dowty (1991)认为语义角色不是一组集合，而是只有两个组合的概念：原型施事（PROTO-AGENT）和原型受事（PROTO-PATIENT）。确定一个动词的论元是否是原型施事，可以从动词的深层概念的结构意义的蕴涵中得到预测。概念结构中的语义角色与语法功能之间的映射可以通过简单的规则进行（最可能是原型施事的论元是主语，最可能是原型受事的论元是宾语，或被动句中的主语）。Dowty的两个规则导致直接指向动词的深层概念结构，因此语义角色完全不再出现于任何表示层中。

Dowty模型的一个缺陷是从事件及其参与者的潜在概念结构不能总是预测出语义角色。例如，Fillmore(1977)指出能够描述同一商业事件（commercial event）的不同动词可以选择不同的方式来表达事件的参与者。例如，在Amie和Benson之间涉及3美元和1个三明治的交易可以用下面任意一种方式来描述：

a. Amie bought the sandwich from Benson for three dollars. （16.34）
（Amie花三美元从Benson处买了那块三明治。）
b. Benson sold Amie the sandwich for three dollars.
（Benson以三美元卖给Amie那块三明治。）
c. Amie paid Benson three dollars for the sandwich.
（Amie付给Benson三美元来买那块三明治。）

动词*buy*，*sell*和*pay*，选择商业事件的不同视角，并选择潜在参与者与题元角色的不同映射来实现这种视角。这三个动词具有完全不同的映射的事实暗示：动词的语义角色必须在动词的词典条目中列出，从潜在的概念结构是不能预测的。

这个事实，再加上前面所提到的动词交替不能完全从语义角色预测的事实（例如*donate*这样的特例），导致许多研究者认为任何有用的计算词典都需要列出每个动词（或形容词或其他谓词）句法和语义的组合的可能性。对每个动词列出组合可能性的另一个好处是也可以给出每个语义框架的概率。

FrameNet项目(Baker et al., 1998; Low et al., 1997)是近期为列出英语中许多谓词的这些元素而做的一次尝试。一个词的FrameNet条目列出了它所带的每个论元集，包括语义角色、句法短语和它们的语法功能的可能集合。FrameNet中所用到的语义角色远比我们曾描述过的那十个例子更具体。每个FrameNet的语义角色被定义为框架（frame）的一部分，而每个框架又是领域的一部分。例如，认知（Cognition）领域存在一些框架，包括静态认知（static cognition）（比如*believe*，*think*和*understand*），思考（cogitation）（比如*brood*，*ruminate*）和判断（judgment）（比如*accuse*，*admire*和*rebuke*）等框架。所有的认知框架都定义语义角色COGNIZER。在判断框架中，COGNIZER被指向JUDGE，该框架也包括一个EVALUEE，一个REASON和一个ROLE，这里是几个例子(Johnson, 1999):

Judge	**Kim** respects Pat for being so brave
Evaluee	Kim respects **Pat** for being so brave
Reason	Kim respects Pat **for being so brave**
Role	Kim respects Pat **as a scholar**

每个条目也被标注了图 11.3 中所描述的短语类型以及语法功能（主语、宾语或补语）。例如，这里是判断动词*appreciate*的部分FrameNet条目，我们只给出该动词的主动涵义，完整的条目也包括被动涵义。下面的例句（有的经过裁剪）来自英国国家语料库（British National Corpus）：

a. JUDGE　　REASON　　EVALUEE
NP/Subj　　NP/Obj　　PP(in)/Comp
I　　still appreciate　　good manners　　in men.

b. JUDGE　　EVALUEE　　REASON
NP/Subj　　NP/Obj　　PP(for)/Comp
I　　could appreciate　　it　　for the music alone.

c. JUDGE　　REASON
NP/Subj　　NP/Obj
I　　appreciate　　your kindness.　　(16.35)

d. JUDGE　　EVALUEE　　ROLE
NP/Subj　　NP/Obj　　PP(for)/Comp
He　　did not appreciate　　the artist　　as a dissenting voice.

对应地，动词*appreciate*的另一个涵义是作为类似*understand*的静态认知的动词，静态认知动词具有诸如 COGNIZER 和 CONTENT 之类的角色，这里是一些例子：

a. COGNIZER　　CONTENT
NP/Subj　　Sfin/Comp
They　　appreciate　　that communication is a two-way process.

b. COGNIZER　　CONTENT　　(16.36)
NP/Subj　　Swh-/Comp
She　　appreciated　　how far she had fallen from grace.

研究这些例句可以清楚地得出一些关于不同语义角色的实现的普遍性原则。JUDGE，COGNIZER和AGENT一般常实现为主动句的主语。ROLE常实现为介词*as*的PP。CONTENT常实现为某种类型的*S*。在这种细粒度的层次上表示语义角色则可使指向句法的映射更清晰。

为了给句子自动指派语义角色，Gildea et al.(2000)给出了一个利用FrameNet训练的随机算法。对每个待标的成分，算法计算每个可能的语义角色的概率。这些概率，利用手工标注的FrameNet数据库进行训练，与动词、待标成分的中心语、动词的语义（主动，被动）以及待标成分的句法范畴（*NP*，*S*和*PP*等）和语法功能（主语，宾语）有关。

像FrameNet这样的方案的主要不足在于，在定义每个领域、每个框架的语义角色时需要大量的人力。Riloff and Schmelzenbach(1998)给出了一个基于语料库的方法，以便于从未标注文本中获取语义角色集。

16.3.2　选择限制

选择限制（selectional restriction）的概念可用于通过容许词位对那些能够与它们在一个句子中出现的词位和短语设置某些语义限制，以增加语义角色。更具体地说，选择限制是由一个词位基于能够填满该词位相关的各种论元角色的概念而施加的语义约束。正如许多其他类型的语言学约束，选择限制最容易观察到的位置也就是它们被违反的地方。研究下面曾在第14章中讨论过的例子：

I wanna eat someplace that's close to ICSI.　　(16.37)

该句有两种可能的剖析结果分别对应于动词*eat*的不及物情况和及物情况。这两个剖析依次导致两个不同的语义分析。在不及物的情形下，短语*someplace that's close to ICSI*是一个附加语，它修饰

动词短语指定的事件，而在及物的情形下，它对“吃”（eating）事件提供一个真正的论元。后一种情形在结构和解释上都类似于下面的例子，这里名词指定了被吃的东西。

I wanna eat some really cheap Chinese food right now. （16.38）

沿着这些技术路线对例句（16.37）进行分析将导致一种语义上病态的结构。这种病态结构表明确实存在 *eat* 对它的 THEME 角色施加的选择限制：它（THEME 角色）必须是一种能吃的东西。在该场景中被提议为 THEME 角色的短语，很难被解释为是可吃的（除非 Godzilla 作为说话人认为早先讨论过），这种解释表明存在类似句法的不合语法性的语义。这种特殊病态结构是由所谓的选择限制违背（selectional restriction violation）引起的，选择限制违背是指语义角色的填充者的语义与谓词施加于该角色的约束之间不一致的情形。

在选择限制能够用于实践之前，对它的不规范描述需要通过许多方法加以精炼。首先需要改进的是对选择限制正确位置的陈述。正如 16.1 节的讨论，词位常常具有许多各种各样的涵义，这些涵义可以对它们的论元施加不同的约束。因此，选择限制针对的是特定涵义，而不是整个词位。研究下面关于词位 *serve* 的例句：

Well, there was the time they served green-lipped mussels from New Zealand. （16.39）

Which airlines serve Denver? （16.40）

Which ones serve breakfast? （16.41）

例句（16.39）说明的是 *serve* 的“烹饪”（cooking）的涵义，常限制它的 THEME 角色为某种食品。例句（16.40）说明的是 *serve* 的“提供商业服务”（provides a commercial service to sense of serve）的涵义，它的 THEME 被约束为某种可以确认的地理或行政实体。第三个例子中的 *serve* 的涵义与第一个非常接近，说明的是 *serve* 的特定某一餐的菜单的涵义。对于多义词位的相同语义角色的这些不同限制，可以加入词典中同一词位的不同涵义。正如在第 17 章中将会讨论的，这一点明显地提示我们选择限制可用于上下文中歧义的消解。

注意，由不同词位以及同一位词的不同涵义所施加的选择限制可能出现在一个范围广泛的具体化层次中，有些词位表达的是很一般化的概念范畴，而其他表达却是非常具体化的范畴。研究下面关于动词 *imagine*，*lift* 和 *diagonalize* 的例句：

In rehearsal, I often ask the musicians to imagine a tennis game. （16.42）

Others tell of jumping over beds and couches they can't imagine clearing while awake. （16.43）

I can't even imagine what this lady does all day. （16.44）

Atlantis lifted Galileo from the launch pad at 12:54 p.m. EDT and released the craft from its cargo bay about six hours later. （16.45）

when the battle was over, Mr. Kruger lifted the fish from the water, gently removed the hook from its jaw, admired it , and eased it back into the lake. （16.46）

To diagonalize a matrix is to find its eigenvalues. （16.47）

如果已经知道 *imagine* 的意义，就不会惊讶于它对能够填充 THEME 角色的概念几乎没有语义约束。另一方面，它的 AGENT 角色被限定为人或其他有生命的实体。相反，例句（16.45）和例句（16.46）中 *lift* 的涵义将它的 THEME 角色限制为可提升的东西，在例子中说明 THEME 是一个能够

包含飞行器（spacecraft）和鱼（fish）的概念。为了满足所有的实际用途，这个概念最好通过一个普遍化的概念，比如物体（physical object）来捕捉。最后，*diagonalize* 对它的 THEME 角色的填充者施加了一个非常具体化的约束，也就是它必须是一个矩阵（matrix）。

这些例子的目的是为了说明关于选择限制的一个重要事实：这些由词典用于表示选择限制的概念、范畴和特征都不是有限语言容量的一部分，而是和词典本身一样可扩展。这一点也是选择限制与语言的一些其他有限特征的区别（这些有限特征用于定义词位，包括词性、语义角色和语义基元等）。

在继续讲述之前，应该指出动词并不是惟一能够对它们的论元施加选择限制的词类。其实，任何谓词表现的词位都可以对填充它的论元角色的概念施加任意语义约束。研究下面的例子，用来说明一些非动词词性的选择限制：

Radon is a naturally occurring odorless, tasteless gas that can't be detected by human senses.　（16.48）

What is the lowest fare for United Airlines flight four thirty?　（16.49）

Are there any restaurants open after midnight?　（16.50）

例句（16.48）中的形容词 *odorless* 和 *tasteless* 被限制于一个能够拥有气味或味道的概念。类似地，正如我们先前在 16.1 节中讨论的，名词 *fare* 被限制于公共交通的各种形式。最后，介词 *after* 的论元必须直接或间接地指定时间点。

选择限制的表示

通过扩展第14章采用的面向事件的意义表示，可以直接捕捉到选择限制的语义。我们知道，一个事件的表示是由象征事件的单个变量、表示这类事件的谓词和指定事件相关角色的一系列变量和关系组成的。忽略 λ 结构的问题，采用语义角色而不是深层事件角色，像 *eat* 这样的动词的语义贡献可以表示为：

$$\exists e,x,y\ Eating(e) \wedge Agent(e,x) \wedge Theme(e,y)$$

从上面的表示中可以知道的所有关于 y（THEME 角色的填充者）的信息是：通过 THEME 关系，它与 *Eating* 事件相关联。为加入 y 必须是可食用的东西这一选择限制，我们可以通过简单地添加一个新的项来实现，如下所示：

$$\exists e,x,y\ Eating(e) \wedge Eater(e,x) \wedge Theme(e,y) \wedge Isa(y,EdibleThing)$$

当遇到像 *ate a hamburger* 这样的短语，语义分析器能够形成如下的表示：

$$\exists e,x,y\ Eating(e) \wedge Eater(e,x) \wedge Theme(e,y) \wedge Isa(y,EdibleThing) \wedge Isa(y,Hamburger)$$

这个表示是完全合理的，因为假如在知识库中有一个合理的事实集，y 在范畴 *Hamburger* 中的成员属性与它在范畴 *EdibleThing* 中的成员属性是一致的。相应地，诸如 *ate a takeoff* 的短语的表示将是非良构的，因为在一个类似事件的范畴，比如 *Takeoff* 中的成员属性与范畴 *EdibleThing* 中的成员属性不一致。

尽管该方法足够捕捉选择限制的语义，但直接使用时仍存在两个问题。首先，采用 FOPC 来进行施加选择限制的简单任务有“牛刀杀鸡”之嫌。完全可以采用更简单的形式体系，通过较少的计算开销来实现。第二个问题是，该方法预先假定存在一个大规模的关于组成选择限制的概念的事实逻辑知识库。遗憾的是，尽管这类知识库正在建设，但是还没有普遍使用，而且几乎没有达到任务所需规模的知识库。

一个更实际的方法，至少对英语来说，是利用WordNet信息库中的上下位关系。在该方法中，语义角色的选择限制是用WordNet的同义集而不是逻辑概念来表述的。如果填充语义角色的词位是由谓词给语义角色指定的同义集的上位词中的一个，则这个给定的意义表示可以被判断为良构的。研究该方法用于下面的 *ate a hamburger* 例子的情形。在 WordNet 的 60 000 个同义集中，包括下面的一个注释为：*any substance that can be metabolized by an organism to give energy and build tissue*（任何能够被生物体新陈代谢以供给能量和构建组织的物质）的同义集:

{ food, nutrient }

有了这个同义集，我们可以将它指定为对动词 *eat* 的 THEME 角色的选择限制，也就是将这个角色的填充者限定为在这个同义集或其下位词中。幸运的是，图 16.10 中 *hamburger* 的上位关系链，显示 *hamburger* 确实是食物。

```
Sense 1
hamburger, beefburger --
(a fried cake of minced beef served on a bun)
=> sandwich
   => snack food
      => dish
         => nutriment, nourishment, sustenance...
            => food, nutrient
               => substance, matter
                  => object, physical object
                     => entity, something
```

图 16.10　WordNet 中有关 Hamburger 可食用的根据

注意，在该方法中并不需要角色的填充者与限定的同义集之间严格匹配，而是只要填充者把限定同义集作为它的最终上位词中的一个，就可以满足选择限制。在 *hamburger* 的例子中，选择限定同义集比 *hamburger* 高出 5 个上位关系层。

当然，这种方法也容许单个词位在不同的具体化层次满足限制。例如，研究将该方法用于我们早先讨论过的动词 *imagine*，*lift* 和 *diagonalize* 的 THEME 角色时的情形。让我们将 imagine 的 THEME 角色限制为同义集{entity, something}，*lift* 的 THEME 角色限制为同义集{object, physical object}，*diagonalize* 角色限制为{matrix}。这种处理正确地容许 *imagine a hamburger* 和 *lift a hamburger*；而且也正确地排除了 *diagonalize a hamburger*。

注意，这个方法依赖于在 WordNet 中出现的词位恰好能够指定所有可能的选择限制所需的准确概念。遗憾的是，我们并没有十足的理由相信在一个语言中用于选择限制的概念集恰好能够被这个语言中的词位所包容。在 WordNet 中通过使用搭配（比如 *physical object* 和 *snack food*），在一定程度上改善了这种情形。

下面更直接地说明这个问题，有许多分类体系介于诸如 CYC(Lenat and Guha , 1991)的常识知识库和诸如 WordNet 的词典库之间。这些混合模型中所包含的客体，不一定对应于单个词典项，而是对应于与语法和语义相关的那些概念。在大多数情况下，这些分类体系的上层部分用于表示领域和语言无关的概念，比如，物体、状态、事件和有生性等。PENMAN Upper Model 是这些分类体系中最成熟的模型之一，我们将在第 20 章中对它进行详细的讨论。

16.3.3　基元分解

在本章和前面几章所描述的意义表示的理论都明确地具有词汇的性质。句子的意义表示是由与

单个词位非常对应的原子符号组成的。但是，除了题元角色，这些词典表示中没有过多的内部结构。基元分解（primitive decomposition）或成分分析（componential analysis）的概念就是为了补充这样一个结构。

为了探讨这些概念，研究由 McCawley (1968)的讨论所促成的如下例子：

Jim killed his philodendron（注释：philodendron 是一种被称为喜林芋的植物）　（16.51）

Jim did something to cause his philodendron to become not alive.　（16.52）

从真值条件的观点，这两个句子具有相同的意义。然而，这不是一种简单的同义关系，因为 *kill* 与例句（16.52）中的任何一个单独词位都不是同义的。确切地说，*kill* 的这个涵义等同于第二个句子中更基本（more fundamental）元素的精确组合。援引第 14 章中的规范形式的概念，我们可以认为这两个例子应该具有相同的意义表示——例句（16.52）的潜在意义表示。把像 *kill* 这样的单一谓词转换为更复杂的谓词集合可以看成是把词的意义拆开或分解为若干个更简单、更基础成分的组合。在本例中，更基础的（或许不可分）成分是与词位 *do*，*cause*，*become not* 和 *alive* 有关的意义表示。

有趣的是这项技术也被用于解释本节曾讨论过的一些词句的交替(Lakoff, 1965; Dowty, 1979)。研究下面的例子：

John opened the door.　（16.53）

The door opened.　（16.54）

The door is open.　（16.55）

这种分解方法认为在所有例句中都有一个与 *open* 相关联的类似状态的单个谓词。这些例句之间意义的不同源于该单个谓词与 *cause* 和 *become* 的潜在基元之间的组合。

注意这个基元分解的方法是非常有限的。在这个方法中，像 DO，CAUSE 和 BECOME 这样的基元与一个可扩展的谓词集（对应于语言中的大部分动词）相关联，因此这个表示方案仍旧具有词典的性质。一种更激进的方法宣称这些内容谓词本身还能被分解为更基本元素的集合。人们提出了许多关于这类基元的方法，在自然语言处理领域应用最广泛的基于基元的表示系统是概念依存方法，简称 CD 方法（Conceptual Dependency）(Schank, 1972)。在这个方法中，11 个基元谓词被用于代表所有类似谓词的语言表达式。图 16.11 所示的是 10 个基元及其意义的简单说明。

基元	定义
ATRANS	The abstract transfer of possession or control from one entity to another.
PTRANS	The physical transfer of an object from one location to another
MTRANS	The transfer of mental concepts between entities or within an entity.
MBUILD	The creation of new information within an entity.
PROPEL	The application of physical force to move an object.
MOVE	The integral movement of a body part by an animal.
INGEST	The taking in of a substance by an animal.
EXPEL	The expulsion of something from an animal.
SPEAK	The action of producing a sound.
ATTEND	The action of focusing a sense organ.

图 16.11　一个概念依存基元集

作为该方法的例子，研究下面的句子及其CD表示：

The waiter brought Mary the check. (16.56)

$$\exists x,y\ Atrans(x) \wedge Actor(x,Waiter) \wedge Object(x,Check) \wedge To(x,Mary) \wedge Ptrans(y) \wedge Actor(y,Waiter) \wedge Object(y,Check) \wedge To(y,Mary)$$

这里，动词*brought*被转换为ATRANS和PTRANS两个基元，以表示服务员（waiter）不但在物理上将支票（check）交给了Mary，而且也把支票的控制权传给了她。注意，在CD中每个基元都与固定的题元角色相关联，以表示动作中各种各样的参与者。

这种组成方法并不局限于动词意义的表示。同样的概念也可以用于将名词分解为更基本的概念。研究将下面的词位*kitten*，*puppy*和*child*分解为更基本元素的例子：

$$\exists x\ Isa(x,Feline) \wedge Isa(x,Youth)$$
$$\exists x\ Isa(x,Canine) \wedge Isa(x,Youth)$$
$$\exists x\ Isa(x,Human) \wedge Isa(x,Youth)$$

这里基元代表的是更基本的客体范畴，而不是动作范畴。利用这些基元，这些词位之间的相近关系以及相关术语*cat*，*dog*和*person*可以用下面的类似公式来捕捉：

$$\exists x\ Isa(x,Feline) \wedge Isa(x,Adult)$$
$$\exists x\ Isa(x,Canine) \wedge Isa(x,Adult)$$
$$\exists x\ Isa(x,Human) \wedge Isa(x,Adult)$$

在自然语言处理中，基元的主要应用领域是语义分析和机器翻译。在语义分析中，本质上基元是用于组织推理处理的。推理规则可以只与少量的基元相关联，而不再需要编译由详细词典项所假定的大量特殊意义。应该注意，在名词性词的表示中使用继承层级已经在很大程度上取代了基元分解。在第21章中将看到，在机器翻译中强调的是将基元作为语言无关的意义表示，即中间语言（interlinguas）。

16.3.4 语义场

16.1节描述的词汇关系具有明确的局部特征，在这些关系中也没有用到词位的内部结构。语义场（semantic field）的概念是为了捕捉某个单独领域的整个词汇集之间的更完全或更整体的关系。研究从ATIS语料库中抽取的下列词汇集：

reservation, flight, travel, buy, price, cost, fare, rates, meal, plane

当然，序列中的许多词位之间可能存在着各自的词汇关系，然而这些关系集合在一起并不能完全地解释这些词位是怎样关联的。显然，它们的定义都是与以飞机旅行为常识背景信息的关联知识块相关的。这类背景知识在各种框架下得到了研究，被分别称为框架(Fillmore, 1985)，模型(Johnson-Laird, 1983)或脚本(Schank and Albelson, 1977)，并在许多计算框架中起到了中心的作用。部分内容将在第18章中介绍。

早先讨论过的FrameNet项目(Baker et al., 1998)为这类知识的获取提供了丰富的资源。在FrameNet中，属于一个特定领域的指向行动、事件、题元角色和客体等的词位与包含于表示该特定领域的框架的概念相连接。就像当前大多数知识库中所做的尝试，这些框架被安排为一个层级，以便于具体的框架能够从更抽象的框架中继承角色。当前FrameNet努力的方向在于创建数千个语义框架的词汇条目。覆盖的领域包括：卫生保健（HEALTH CARE），机会（CHANCE），感觉（PERCEPTION），交通（COMMUNICATION），事务（TRANSACTION），时间（TIME），空间

（SPACE），身体（BODY），运动（MOTION），生命阶段（LIFE STAGES），社交环境（SOCIAL CONTEXT）和认知（COGNITION）等。

16.4　语言的创造性与词典

到目前为止，本章所讲述的词汇语义学的方法都是通过从大型静态词典中检索所需的意义表示来说明单词的意义。一种更实际的观点则认为意义表示可能需要动态地生成，静态词典只是间接地对意义表示的内容做贡献。按这种观点，词典中许多显而易见的多义词都归因于这种生成能力。当然，这种生成能力不是无限的，也不是毫无章法的，而是受许多生成模型的约束。这些生成模型能够系统地把词汇的、语法的、上下文的以及常识性的知识进行融合，以生成我们每天话语中的新意义。在下面的讨论中，我们将简短地给出两个比较重要也比较普遍的模型：隐喻和换喻。

16.4.1　隐喻

提到隐喻（metaphor），我们就想到这样的情形：所指和所思的概念恰好是由表示其他完全不同类型的概念（other completely different kinds of concept）的单词和短语来表达的。在所有语言中隐喻都很普遍，而且大部分多义现象，包括词典中所列的多种涵义以及更多未列出的涵义，都被归因于隐喻。为了使这个概念更具体，研究 WSJ 语料库中的如下句子：

That doesn't scare Digital, which has grown to be the world's
second-largest computer maker by poaching customers of IBM's
mid-range machines.　（16.57）

让我们探讨本例中使用 *scare* 的动机。动词 *scare* 在 WordNet 中有两个紧密联系的涵义：使惊恐（to cause fear in），使丧失勇气（to cause to lose courage）。尽管思考本例所用的是哪个涵义是件有意义的事，但更有意义的是研究一个公司失去勇气或具有勇气到底意味着什么。对这个句子来说，无论采用这两个涵义中的哪一个，都不得不面对这样的情形：公司可以像人一样拥有害怕（fear）或勇气（courage）这样的情感。当然，它们并不具有这样的能力，但是我们确实以这样的方式来表达它们和思考它们，就仿佛它们具有这样的能力。*scare* 的这种基于隐喻的用法容许我们把一个公司作为人来看待，我们称之为“公司比做人”（CORPORATION AS PERSON）的隐喻。

当然，这个隐喻算不上是 *scare* 的新颖用法，也不是什么特殊用法，而只是把公司比做人的一种传统方式，在 WSJ 的如下例句中还可以看到单词 *resuscitate*（复苏），*hemorrhage*（大出血）和 *mind*（意见）的类似用法：

Fuqua Industries Inc. said Triton Group Ltd., a company it helped
resuscitate, has begun acquiring Fuqua shares.　（16.58）

And Ford was **hemorrhaging**; its losses would hit $1.54 billion in 1980.　（16.59）

But if it changed its **mind**, however, it would do so for investment reasons, the filing said.（16.60）

每个例子都显示出了对基本的“公司比做人”的隐喻的精巧使用。前两个例子把健康的概念扩展为对一个公司财政状况的表达，而在第三个例子中把 *mind* 用于公司以捕捉公司策略的概念。

像“公司比做人”这样的隐喻构造被称为约定俗成的隐喻（conventional metaphor）。Lakoff and Johnson(1980)令人信服地证明我们每天所遇到的许多隐喻表达都是由少数简单而常见的模式来激发的。

16.4.2 换喻

换喻（metonymy）这个术语是指我们用一些与它有紧密联系的其他概念来表达该概念时的情形。研究 WSJ 中的如下例子：

GM killed the Fiero because it had dedicated a full-scale to building the plastic-bodied car... (16.61)

本例中 *kill* 大致表示结束某种正在进行的工作或行为。在本例中，这种正在进行的行为是指创立、经销及出售一种特别类型的汽车。这种用法的潜在隐喻是把行为比做有生命的事物（ACTIVITY AS LIVING），使得这种行为的中止可以被称为杀死（Killing）。更值得注意的是，在这个例子中并没有明显地提到任何行为。特别是，*kill* 的 THEME 角色明确地指向 *the Fiero*。为了使这个隐喻所表达的意义成立，该短语就不能指向汽车，而是应该指向整个销售和生产的工作。在这个例子中，工作的最终成果被用于表达为生成这个成果而付出的整个工作，这被我们称为“产品替换过程”（PRODUCT FOR THE PROCESS）换喻。

关于更常见的换喻的例子，常用的模式有“作家替换作家的作品”，如 *He likes Shakespeare.*（他喜欢的莎士比亚的作品）；“地点替换机构”，如 *The White House had no comment.*（美国政府没有评论）。注意，尽管这些模式表面上与隐喻很类似，但它们的结构和用法都非常不同。与隐喻不同（连接的是完全不同的概念），这些模式连接的是有紧密联系的概念。实际上，隐喻是用不同类型的概念来表示某个概念的，而换喻只不过是一种提及相关概念的方式。例如，上面给出的白宫（*White House*）的例子并不是要把实际建筑中的任何财产划归行政机构，仅仅是行政机构的一种间接表达方式。

16.4.3 隐喻和换喻的计算方法

隐喻和换喻的计算方法被分成两种基本类型：基于约定的方法和基于推理的方法。这两种方法都假定：利用静态词典的组合语义分析在面对隐喻和换喻这样的输入时无法产生足够的意义表示。因此，正是通过这些后续处理来生成这类输入的正确语义表示。

基于约定的方法的关键在于：通过直接应用语言中关于约定俗成的隐喻和换喻的特定知识(Norvig, 1987; Martin, 1990; Hayes and Bayer, 1991; Veale and Keane, 1992; Jones and McCoy, 1992)来诠释这类比喻语言。这些系统的基本假设是，借助约定俗成的隐喻的形式化表示一个语义分析器，可以将组合方法导出的意义表示替换为能够更接近地反映出输入的真实意义的意义表示。这些方法假定少量约定的隐喻的核心集合可以表示为系统知识的一部分，并且这些知识足够解释各种各样的输入。

基于推理的方法避开了隐喻和换喻的约定俗成概念而另辟蹊径，认为比喻语言的处理最好采用一般的推理技术，而不是当成与某个特定语言现象相关的问题来处理。因此，采用这种方法的计算模型通常基于已有的针对更一般推理任务的模型。对于隐喻，最主要的方法是类比推理(Russell, 1976; Carbonell, 1982; Gentner, 1983; Fass, 1988, 1991, 1997)。该方法认为隐喻依赖于组合方法从输入导出的意义表示与捕捉输入的预期意义的正确表示之间的内在结构相似性。

16.5 小结

本章介绍了与词汇项意义有关的各种问题。下面是本章的主要内容：

- 词汇语义学（lexical semantics）是一门研究词位以及单个词位内部与意义相关的结构之间涉及意义的系统化联系的学科。

- 同形关系（homonymy）指词形（包括字形和音形）相同但意义不同的词之间的关系。
- 多义关系（polysemy）指的是单个词位具有多个相关涵义的概念。
- 同义关系（synonymy）指具有相同意义的不同词位之间的关系。
- 上下位关系（hyponymy）指具有类别包含（class-inclusion）关系的词位之间的关系。
- 语义场（semantic field）被用于捕捉某个单独领域的整个词位集之间的语义关系。
- WordNet 是一个大规模的英语词汇信息库。
- 题元角色（thematic role）是通过对各类动词的相似角色的归纳从深层语义角色的细节中抽象出的一组范畴。
- 语义的选择限制（selectional restriction）容许词位对那些与它们在一个句子中同现的成分设置某些语义限制。
- 基元分解（primitive decomposition）容许根据基元词汇的有限子集来表示单个词位的意义。
- 像隐喻和换喻这样的生成手段是语言中的普遍现象，原则上它们所生成的新意义一般不会包含在静态词典中。

16.6　文献和历史说明

Lyons(1977)和 Cruse(1986)是关于词汇语义学的经典语言学文献。在 Pustejovsky and Bergler (1992), Saint-Dizier and Viegas(1995)以及Klavans(1995)等文献中包含对词汇语义学计算成果的介绍。

Martin(1986)和 Copestake and Briscoe(1995)讨论了多义词表示的计算方法。Fellbaum (1998)是关于WordNet的最全面的文献集。有许多研究者使用现有词典作为词汇资源。使用Merriam Webster 词典的 Amsler(1980, 1981)是最早的研究文献之一。近期许多系统(Boguraev and Briscoe, 1989)使用了电子版的现代英语朗文词典（Longman's Dictionary of Contemporary English）。

题元角色或格角色，可以追溯到Fillmore(1968)和Gruber(1965)的工作。Fillmore的研究对自然语言处理工作具有深远和直接的影响。在很长一段时期里，几乎所有自然语言理解的工作都是使用 Fillmore 格角色的某个版本。早期在这一支脉的工作归功于 Simmons(1973, 1978, 1983)。

Katz and Fodor(1963)首次把选择限制作为一种刻画语义的良构性方法。McCawley(1968)第一个指出选择限制不能局限于有限的语义特征，而是不得不从大规模的无约束世界知识中抽取。

Lehrer(1974)是关于语义场的经典文献。最近有关这个话题的论文是Lehrer and Kittay(1992)。Baker et al.(1998)中描述了正在进行的 FrameNet 项目。

使用基元、成分和特征来定义词典项已有很长的历史。Nida(1975)对成分分析研究给出了一个全面的综述。长期以来 Wierzbicka(1996)一直倡导在语言的语义中使用基元。另一个突出成果是 Jackendoff(1983, 1990)的概念语义（Conceptual Semantic），它融合了题元角色和基元分解。在计算方面，Schank(1972)的概念依存仍然是自然语言处理中使用范围最广泛的基元集合。Wilks(1975a)是倡导将基元用于机器翻译以及一般自然语言理解的先行者。近期，Dorr(1993, 1992)在她的机器翻译工作中使用 Jackendoff 框架进行了大量计算研究。

Ortony(1993)是关于隐喻的很有影响力的论文集。Lakoff and Johnson(1980)是有关隐喻和换喻的经典理论研究。Pustejovsky(1995)引入了生成词典（Generative Lexicon)的概念，这个概念框架反对把词典看成是静态的词汇库，而是赞成一种更动态的观点。Russell(1976)给出了一个有关隐喻的最早的计算方法。其他早期研究包括：DeJong and Waltz(1983)，Wilks(1978)和 Hobbs(1979b)。近期有关隐喻分析的计算研究包括：Fass(1988, 1991 1997)，Martin(1990)，Veale and Keane (1992)，Iverson and Helmreich(1992)以及Chandler(1991)。在Martin(1996)中给出了关于隐喻以及其他类型的比喻语言的计算方法的综述。

第17章　词义排歧与信息检索

Oh are you from Wales?
Do you know a fella named Jonah?
He used to live in whales for a while.
Groucho Marx[①]

本章介绍与词汇语义处理（lexical semantic processing）有关的一些主题。这里指一些使用单词意义的应用，但是它们与组合句子分析和话语理解等更复杂的任务具有不同程度的差别。

首先讲述的主题是在理论和实践上都具有非常重要意义的词义排歧（word sense disambiguation）。回想我们在第16章中曾指出的，词义排歧是指审查上下文中的词例并确定每个单词所使用的确切涵义的处理。正如我们将看到的，由于单词的涵义具有一些迷惑人的特性，这并不是一件简单的任务，但是在特定的合理假设下，鲁棒的算法能够取得较高的精度。

讲述的第二个主题是信息检索（information retrieval），这是一个非常广泛的领域，包括一系列与各种媒体的存储、分析和检索有关的主题(Baeza-Yates and Ribeiro-Neto, 1999)。本章所关注的只是满足用户信息查询要求的文本文档的存储和检索。我们感兴趣的是把用户需求表达为单词以及把文档根据它们所包含的单词进行表示的方法。在17.3节中将介绍向量空间模型（vector space model），目前许多检索系统（包括大部分网络搜索引擎）所使用的都是它的改进模型。

17.1　基于选择限制的排歧

在第15章讨论组合语义分析的大部分章节中，我们都忽略了词汇歧义的问题。至此，应该很清楚这不是一个十分合理的方法。如果缺乏有效的手段来选取输入单词的正确涵义，那么词典中的大量同形词和多义词将借蜂拥而至的候选释义以排山倒海之势击溃任何解决方法。在已知句法的词性标注信息的情形下，对这种歧义问题有两种基本处理方法。在语义分析的规则到规则（rule-to-rule）的集成方法中，正确的单词涵义的选择是语义分析中消除非良构语义表示的副产品。在独立方法中，涵义排歧是在语义分析之前独立进行的。本节将讨论前一种方法中选择限制的角色。独立的方法将在17.2节中详细讨论。

大部分集成方法在进行排歧处理时所用到的知识源都是选择限制和类型层级。它们被用于排除不正确的涵义，以减少语义分析中出现的歧义数量。在集成的规则到规则方法的语义分析中，选择限制被用于阻止包含选择限制违背的成分的意义表示的形成。通过阻止这种非良构的成分，语义分析器只需处理带有较少歧义的意义表示。这种通过消除不正确涵义导致的有缺陷的表示来发现正确涵义的能力，可以看成是一种间接的词义排歧方式。尽管该方法的语言学基础可以追溯到Katz and Fodor(1963)，但是较成熟的计算探索应该归功于Hirst(1987)。

---

① 译文如下，供读者参考（请注意英文中 wales, whales 和 while 的发音很接近）：
哦，你来自威尔士？
你认识一个叫 Jonah 的小伙子吗？
他过去经常会在 whales 住一段时间。

作为该方法的例子，研究 WSJ 中的如下例句，仅考虑词位 *dish* 的用法：

"In our house, everybody has a career and none of them includes washing **dishes**," he says.（17.1）

In her tiny kitchen at home, Ms. Chen works efficiently, stir-frying several simple **dishes**, （17.2）
including braised pig's ears and chicken livers with green peppers.

上面的例句包含词位 dish 的两个涵义。第一个指盛我们吃的东西的物体（器皿），而第二个指实际的饭菜。在理解这两个例子时不会产生歧义的事实归因于 *wash* 和 *stir-fry* 对它们 PATIENT 角色施加的选择约束，以及与 *dish* 的两个涵义相关的语义类型信息。*wash* 施加的限制与 *dish* 的食物的涵义有冲突，因为它表示的事物通常不能被清洗。类似地，*stir-fry* 的限制与 *dish* 的人工制品的涵义有冲突，因为它不表示可以吃的东西。因此，在这两个例子中通过消除与选择限制不匹配的涵义，谓词选出了歧义论元的正确涵义。

研究 WSJ 和 ATIS 中有关歧义论元 serve 的如下例子:

Well, there was the time they **served** green-lipped mussels from New Zealand. （17.3）

Which airlines **serve** Denver? （17.4）

Which ones **serve** breakfast? （17.5）

例句（17.3）中 *serve* 的涵义需要某种食物作为它的 PATIENT，例句（17.4）中的涵义需要某种地理或政治的实体，最后一个例句需要的是菜单。如果我们假设 *mussels*，*Denver* 和 *breakfast* 都是没有歧义的，那么正是这些例子中的论元选出了动词的正确涵义。

当然，我们也会遇到谓词和论元都具有多种涵义的情形。研究下面的 BERP 中的例子：

I'm looking for a restaurant that **serves** vegetarian **dishes**. （17.6）

假定 *serve* 有 3 个涵义，*dish* 有 2 个涵义，在本例中就会产生 6 种可能的涵义组合。然而，由于 6 种组合中只有一种不违背选择限制，因此可以直接确定 *serve* 和 *dish* 的正确涵义，这里谓词和论元共同选出了正确的涵义。

尽管可以采用各种方法将这种排歧方式集成于语义分析器，但是最直接的方法仍然是沿用第15章所介绍的规则到规则的策略。在这种集成方法中，一旦生成了句法成分，它们相应的意义表示的片断就将被组合，并检查是否含有选择限制的违背。这些包含选择限制违背的表示将不再用于进一步的处理。

该方法需要在用于语义分析的知识结构中附加对两种信息的访问，即论元的层级类型信息和谓词对论元的语义选择限制信息。回想第16章，WordNet提供了对这两个信息进行计算编码的知识。类型信息可以通过作为谓词论元的意义结构的中心语的下位关系信息获得。论元角色的选择限制信息可以通过将正确的 WordNet 同义集与每个作为谓词的词汇项相关联来进行计算编码。

17.1.1　选择限制的局限性

利用选择限制来进行词义排歧也给理论和实践带来了许多问题。这些问题的首要征兆就是在类似下面的例句中由于可利用的选择限制过于空泛而很难惟一地选出正确涵义：

What kind of **dishes** do you recommend? （17.7）

在这个例子中，我们或者依赖于 17.2 节将讨论的独立方法，或者利用第 18 章将讨论的具有较多上下文的话语知识。

更多的问题产生于那些确实明显违反了选择限制而又是完全良构和可解释的例子,因此任何基于严格地排除这种解释的方法都会导致严重的问题。研究 WSJ 中的如下例句:

But it fell apart in 1931, perhaps because people realized you can't **eat** gold for lunch if you're hungry.(17.8)

短语 *eat gold* 明显地违反了 *eat* 对它的 PATIENT 角色施加的选择限制。然而,这个例子是良构的。关键点在于由 can 't 建立的否定环境先于选择限制的违反。从这个例子可以清楚地看出:如果在一个较宽的上下文中选择限制的违背变得可以接受,那么这时对选择限制的任何单纯的局部(或规则到规则) 分析都不可能成功。

下面的例句说明了选择限制的第二个问题:

In this two championship trials, Mr. Kulkarni **ate** glass on an empty stomach, accompanied only by water and tea. (17.9)

虽然这个例句所描述的事件有点不寻常,但是句子本身从语义角度并不是非良构的,尽管违背了 *ate* 的选择限制。这类例子说明题元角色和选择限制仅仅大致近似于它们所表示的深层概念。不能期望用它们来解释 *ate* 的用法,这是需要关于动作“吃”到底是什么的深层次常识的。最多,它们能够反映出这样的概念:被吃的东西通常是可吃的。

最后,正如在第 16 章中的讨论,隐喻和换喻也会对这种方法带来挑战。研究 WSJ 中的如下例句:

If you want to **kill** the Soviet Union, get it to try to **eat** Afghanistan. (17.10)

其中, *kill* 和 *eat* 对 PATIENT 角色施加的典型选择限制将排除所有可能的字面涵义,而不给系统留下任何可能的意义。许多系统在这种情形下会触发后备机制,用于解释隐喻和换喻(Fass, 1997)。

根据 Hirst(1987)的观察,这种例子常常导致所有涵义被排除,以至于停止意义分析。一种减轻这种问题的方法是把选择限制看成优先选择但不是严格必需的。虽然多年来有许多应用该方法的示例(Wilks, 1975c, 1975b, 1978),但是只有 Resnik(1997)的研究经过了较全面的经验评估,其中用到了选择关联系数(selectional association) 的概念。选择关联系数是对一个谓词和支配该谓词论元的类别之间的相关程度的概率的度量。Resnik(1997)给出了使用 WordNet 中的上下位关系并与一个标注了动词论元关系的语料库相结合来获得这些关联系数的方法。

Resnik(1998)表明这些选择限制可用于有限形式的词义排歧。笼统地讲,该算法选择具有最高的选择关联系数的涵义作为论元的正确涵义,这里的选择关联系数是指谓词与论元的祖先的上位词中的任何一个之间的关联系数。Resnik(1997)报道: 对动宾关系利用该技术可以获得44%的正确率,比采用最大频率法的基准系统的正确率(28%) 有较大提高。该方法的局限性在于它只考虑在谓词没有歧义时选择论元的正确涵义。在谓词和论元都具有歧义时,则需要一个更复杂的决策标准。

17.2 鲁棒的词义排歧

用于排歧的选择限制方法由于需要太多信息而不能用于大规模实际应用。即使利用WordNet也很难找到所有谓词角色的完整选择限制信息和所有可能填充者的涵义的完整类型信息。另外,正如我们在第10章、第12章和第15章中所讲述的,在引入非限定文本的环境下很难获得所有输入完整和精确的剖析。

为了解决这些问题,多年来已经开发出了许多采用更适中规模信息的鲁棒的独立排歧系统。与词性标注系统一样,这些系统也以独立方式运行并对从其他处理来的输入信息几乎没有做太多假设。

下面将探讨如何把监督、自举和非监督机器学习方法用于这个问题，然后将研究构建独立标注系统时机器可读词典所起的作用。

17.2.1　机器学习方法

在机器学习方法中，通过训练（train）系统以完成词义排歧的任务。通过这些方法学习得到的是一个分类器，这个分类器可以用于为那些未经训练的实例从固定数目的涵义中指派一个涵义。这些方法随训练语料的性质、所用语料的规模、人工干预的程度、所用语言学知识的类型以及生成的输出的不同而不同。它们的共同点在于：是从数据获取任务所需的知识而不是通过人工分析。在探讨这些方法时需要牢记的首要问题是该方法是否具有规模，也就是说，是否可能将该方法应用于整个语言的绝大部分词汇？

输入：特征向量

在大多数方法中，原始输入是由两部分组成的：需要排歧的单词，我们称之为目标词（target word），以及目标词所嵌入的一部分文本，我们称之为上下文（context）。我们以下面的方式来处理原始输入：

- 通常利用第 8 章所讲述的高精度方法对输入进行词性标注。
- 可能用围绕目标词的较大或较小的文本片断来代替原始上下文。
- 常常对上下文中的所有单词进行一些词干处理，或更复杂的形态处理。
- 也有少数系统，利用部分剖析的某些形式或依存剖析来确定题元或语法角色和关系。

在这些最初的处理之后，我们需要从输入中抽取用于捕捉与学习任务有关的信息的固定特征集。这个学习任务由两个步骤组成：选择相关的语言学特征以及将它们编码为学习算法可用的形式。一个简单的特征向量（feature vector）由一系列能很容易地将最常用的语言信息进行编码并且适用于大部分学习算法的数值（numeric value）或标称值（nominal value）组成。

在训练词义排歧（简称为 WSD）系统时所用的语言特征可以大致分为两类：搭配特征和共现特征。通常，术语“搭配”指两个单词项之间可以量化的特定位置的关系。搭配特征所编码的信息是关于占据目标词的左边或右边的特定位置的词汇。典型的特征包括单词、单词的词根以及单词的词性等。这些特征可以有效地用于对局部的词汇和语法信息进行编码，而这些局部的词汇和语法信息常常可以精确地分离出一个已知的涵义。

作为这类特征编码的例子，研究下面需要对单词 *bass* 进行排歧的例子：

An electric guitar and **bass** player stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps. （17.11）

则可生成下面的由目标词左边和右边各两个单词以及它们相应的词性所组成的特征向量：

```
[guitar, NN1, and, CJC, player, NN1, stand, VVB]
```

第二种类型的特征由这些相邻词的同现数据组成，此时并不考虑每个词的具体位置。在这个方法中，以单词本身（或它们的词根）为特征。在围绕目标词的区域中，单词出现的次数被称为特征值。这个区域常常定义为一个以目标词为中心的固定大小的窗口。为了使该方法具有可操作性，一小部分常用实义词被选为特征。这种特征可以有效地捕捉出现目标词的话语的普通主题。这些特征也有助于识别那些用于特定领域的单词的涵义。

例如，从WSJ语料库抽出的包含单词*bass*的句子集可以统计出12个最常用的实义词：*fishing*, *big*, *sound*, *player*, *fly*, *rod*, *pound*, *double*, *runs*, *playing*, *guitar*, *band*。可以由这12个词为特征组成一个共现向量。用这些词为特征，以句子中的10个单词数为窗口，则例句（17.11）可以表示为下面的向量：

```
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0]
```

我们将会看到，大部分用于排歧的鲁棒的方法都同时利用了搭配特征和共现特征。

监督学习方法

在监督学习方法中，词义排歧系统是从与所用测试集分布相同的具有代表性的已标注实例集中学习得到的。这是把监督学习方法用于产生分类器的应用。在这类方法中，同时需要一个学习系统和一个训练集，这个训练集是由带有特征编码的输入与它们的正确标记或正确类别共同组成的。该系统的输出是一个能够为新的带有特征编码的输入指派标记的分类系统。

Bayesian分类器(Duda and Hart, 1973)、决策表(Rivest, 1987)、决策树(Quinlan, 1986)、神经网络(Rumelhart et al., 1986)、逻辑学习系统(Mooney, 1995)以及邻近方法(Cover and Hart, 1967)等都适用于该模式。这里将只讨论朴素Bayes（naive Bayes）方法和决策表方法，因为它们是词义排歧中最常用的方法。

用于WSD的朴素Bayes分类器的方法基于下面的假设：为输入向量选择最佳涵义等同于为该指定向量选择最可能的涵义。换言之：

$$\hat{s} = \underset{s \in S}{\operatorname{argmax}} P(s|V) \tag{17.12}$$

在这个公式中，S表示合乎该向量关联目标词的词义的涵义集，s表示在S中每个可能的涵义，V代表输入上下文的向量表示。与大部分的情形相同，很难直接获取这个方程的统计信息。因此，我们需要以下面常见的Bayes方式重写该公式：

$$\hat{s} = \underset{s \in S}{\operatorname{argmax}} \frac{P(V|s)P(s)}{P(V)} \tag{17.13}$$

显然，指定向量与涵义之间可获取的相关数据由于太过稀疏而无法使用。然而，在已标注训练集中包含大量的有关在指定涵义的上下文中单个特征值偶对的信息。因此，我们可以做出下面的独立假设（这是该方法名称的来源，我们也曾在词性标注、语音识别以及概率剖析中成功地使用过这个假设）：我们朴素地（natively）假设特征之间彼此独立。利用这个假设则可以产生下面的对$P(V|s)$的近似：

$$P(V|s) \approx \prod_{j=1}^{n} P(v_j|s) \tag{17.14}$$

换言之，在指定涵义的条件下，我们可以通过向量单个特征的条件概率的乘积来估算整个向量的条件概率。

已知上面的方程式，训练一个朴素Bayes分类器就等同于从已标注涵义的训练语料库中收集关于目标词的每个涵义与单个特征之间的相关统计信息。为了具体化这个概念，让我们回到例句（17.11）。这个例句所需的单个统计信息可能包括单词player出现在每个bass涵义的右边紧靠位置的概率，或者包括单词guitar出现在每个bass涵义的左边相隔一个单词的位置的概率。

回到方程式（17.13），项$P(s)$是每个涵义的先验概率，它恰好对应于每个涵义在标注涵义的语料库中所占的比例。最后，由于$P(V)$对所有可能的涵义都是相同的，因此不会影响最终涵义的排列顺序，可以忽略不计而得到下式：

$$\hat{s} = \underset{s \in S}{\operatorname{argmax}} P(s) \prod_{j=1}^{n} P(v_j|s) \tag{17.15}$$

显然，第6章讨论的所有关于零计数和平滑的技术都可以用在此处。

方法论探讨：WSD 系统的评测

用于评测词义排歧的基本标准就是简单的精度：正确标注的单词所占的百分比。与这个标准相比较的主要基准是最常见涵义（most frequent sense）标准(Gale et al., 1992)：当一个系统只是简单地选择每个单词的最常见涵义时系统的性能。

使用精度时需要能够获取测试集中单词的正确涵义。幸运的是，目前有两个大型的标注涵义的语料库可用：SEMCOR 语料库(Landes et al., 1998)由用 WordNet 涵义标注的部分 Brown 语料库组成，SENSEVAL 语料库(Kilgarriff and Rosenzweig, 2000)是由 HECTOR 语料库和词典项目推动的一个标注语料库。

由于使用简单的精度而产生的复杂性是：用于评测的涵义的性质会对结果产生极大的影响。特别是，同形异义词之间使用较粗的区分所得出的结果（比如 *bass* 的音乐和鱼的涵义）不能简单地与基于使用细粒度的区分所得出的结果（比如那些传统词典或类似 WordNet 的词典资源中的涵义）进行对比。

第二个复杂性是不得不使用比精度更复杂的标准：部分可信度（partial credit）。例如，把 bass 的某个特定音乐涵义与鱼的涵义弄混淆，显然比把它与其他音乐涵义弄混淆更糟糕。有了这个标准，一个完全涵义匹配将获得全部的可信度，而选择粗略性的涵义将获得部分可信度。当然，这个方案完全依赖于所用特定词典涵义的组织。

目前，词义排歧系统已有可用的标准评测体系。特别是，SENSEVAL 成果(Kilgarriff and Palmer, 2000)所提供的词义排歧评测体系，正如MUC(Sundheim, 1995b)和TREC(Voorhees and Harman, 1998)分别为信息抽取和信息检索所提供的评测体系。

在一次评测许多监督学习方法的大型实验中，Mooney(1996)指出朴素Bayes分类器和神经网络的性能最好，它们在为包含单词 *line* 的例句语料库指派 *line* 的6个涵义中的一个时都取得了约73%的正确率。

决策表分类器（decision list classifier）等同于大部分程序语言中的简单分支语句。在一个决策表分类器中，一连串的测试被应用于每个以向量为编码的输入。如果这个测试成功，则返回与该测试相关的涵义。如果这个测试失败，则进行序列中的下一个测试。继续这个过程直到列表的结尾，这时一个默认测试将简单地返回最常见的涵义。

图17.1给出了为从 *bass* 的音乐涵义中区分出鱼的涵义的任务而设计的部分决策表。第一个测试表示，如果单词 *fish* 在输入文本中的任何位置出现，则 bass[1] 是正确的答案。如果没有出现，则依次进行每个后续的测试直到返回真值。与分支语句一样，在列表的结尾处包含一个返回真值的默认测试。

学习决策表分类器的过程是由基于训练数据特征的单个测试的生成和排序组成的。许多方法都可以用于生成这种列表。在 Yarowsky(1994)所用的方法中，每个单独的特征值偶对组成一个测试。然后这些测试按照在整个训练集中各自的精度来排序，测试的精度基于它的对数似然值：

$$\text{Abs}\left(\text{Log}\left(\frac{P(Sense_1|f_i = v_j)}{P(Sense_2|f_i = v_j)}\right)\right) \tag{17.16}$$

根据这个计算值，通过对列表中的测试进行简单排序就可以从这些测试中生成决策表。Yarowsky (1996)指出，这种技术在各种二元决策任务中都取得了超过 95% 的正确率。

应该指出，这里的训练方法与标准决策表学习算法有相当大的不同。这些方法的细节以及理论依据可以参阅 Rivest(1987)或 Russell and Norvig(1995)。

规则		涵义
fish within window	⇒	**bass**[1]
striped bass	⇒	**bass**[1]
guitar within window	⇒	**bass**[2]
bass player	⇒	**bass**[2]
piano within window	⇒	**bass**[2]
tenor within window	⇒	**bass**[2]
sea bass	⇒	**bass**[1]
play/V bass	⇒	**bass**[2]
river within window	⇒	**bass**[1]
violin within window	⇒	**bass**[2]
salmon within window	⇒	**bass**[1]
on bass	⇒	**bass**[2]
bass are	⇒	**bass**[1]

图 17.1 一个经简化的决策表用于从 *bass* 的音乐涵义中区分出鱼的涵义。取自 Yarowsky(1996)

自举方法

监督方法的一个主要缺点是需要大规模的已标注涵义的训练集。自举方法（bootstrapping approach）(Hearst, 1991; Yarowsky, 1995)消除了对这种大规模训练集的依赖，只需每个感兴趣词位的每个涵义的少量例句。这些标注的例句被当成种子（seed）供上面所介绍的任何一种监督学习方法使用，以便于训练出一个最初的分类器。然后，这个最初的分类器被用于从剩余的未标注语料库中抽取较大的训练集。重复这个处理将生成一系列精度和覆盖率逐步提高的分类器。

该方法的关键点在于能够从小的种子集生成较大规模的训练集。为了获得成功，它必须只包括那些在最初分类器中具有高可信度的实例。然后，较大的训练集用于生成一个更精确并具有更高覆盖率的新分类器。随着每次的重复处理，训练语料不断增加而未标注语料不断减少。与大部分重复方法一样，这个处理可以持续进行，直到训练集的错误率足够低或直到未标注语料中没有高于阈值的例子。

许多方法都可以用于生成这些自举方法的最初的种子训练集。Hearst(1991)通过从原始语料库手工标注小的例子集的方式来生成种子集。这个方法主要有下面三个优点：

- 可以保证种子集中的实例是正确的，从而使学习者不至于从错误的实例开始。
- 人工分析不但可以选出正确的例子，而且可以选出每个涵义的典型例子。
- 非常容易实现。

另一个可供选择的有效技术是查找包含与目标涵义有紧密联系的单词或短语的那些句子。Yarowsky(1995)称之为“一个搭配一个涵义”（One Sense per Collocation）的约束，给出的实验结果证明这种方法能够产生相当好的结果。为了说明这种技术，考虑我们希望为*bass*鱼的涵义和音乐的涵义生成一个合理的种子集的情形。不用苦思冥想，就可能提出用fish作为bass[1]的合理指示者，而用*play*作为bass[2]的合理指示者。图17.2所示的是这种用字符“fish”和“play”在从WSJ抽出的*bass*例句库中查找而得到的部分结果。

当然，我们也希望能够自动给出这些相关的单词。Yarowsky(1995)对选择有效的相关单词给出了两个建议：从机器可读词典的条目中抽取，或利用第6章讲述过的那些搭配统计来选择种子。对于一个涉及12个单词的大致的二元指派，Yarowsky(1995)报告的平均性能达96.5%。在这些实验中，利用相关性找出的种子句通过自举方法而获取训练集，这些训练集被用于为每个单词训练一个决策表分类器。

Klucevsek **play**s Giulietti or Titano piano accordions with the more flexible, more difficult free **bass** rather than the traditional Stradella **bass** with its preset chords designed mainly for accompaniment.

We need more good teachers – right now, there are only a half a dozen who can **play** the free **bass** with ease.

An electric guitar and **bass play**er stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps.

When the New Jersey Jazz Society, in a fund-raiser for the American Jazz Hall of Fame, honors this historic night next Saturday, Harry Goodman, Mr. Goodman's brother and **bass play**er at the original concert, will be in the audience with other family members.

The researchers said the worms spend part of their life cycle in such **fish** as Pacific salmon and striped **bass** and Pacific rockfish or snapper.

Associates describe Mr. Whitacre as a quiet, disciplined and assertive manager whose favorite form of escape is **bass fish**ing.

And it all started when **fish**ermen decided the striped **bass** in Lake Mead were too skinny.

Though still a far cry from the lake's record 52-pound **bass** of a decade ago, "you could fillet these **fish** again, and that made people very, very happy," Mr. Paulson says.

Saturday morning I arise at 8:30 and click on "America's best-known **fish**erman," giving advice on catching **bass** in cold weather from the seat of a bass boat in Louisiana.

图 17.2　利用 *play* 和 *fish* 与 *bass* 的相关性从 WSJ 中抽取的 bass 例句

非监督方法：单词涵义的发现

词义排歧的非监督方法在训练中避免使用任何已标注涵义的数据。在这些方法中，以表示未标注实例的特征向量为输入，然后根据一个相似矩阵将它们分为一些聚类（cluster）。这些聚类可以被表示为其组成部分的特征向量的平均值，并且手工标注为已知的涵义。对于那些未经训练的具有特征编码的实例，可以通过指派与它们最接近的聚类（根据相似矩阵判断）的涵义而加以分类。

幸运的是，聚类（clustering）是一个被深入研究过的问题，已经有许多标准算法可应用于以数值向量为结构的输入(Duda and Hart, 1973)。在语言应用中经常使用的一种技术称为合并聚类（agglomerative clustering）。在这项技术中，最初每N个训练实例被指派为一个单独的类，然后通过持续合并最相似的两个类，以自底向上的方式形成新的类。这种处理一直持续到到达指定的类组数，或类组之间满足一些全局的良好性标准。为了避免大量训练实例而造成计算开销，随机抽样可用于最初的训练集(Cutting et al., 1992b)，以达到相似的结果。

非监督方法不使用手工标注数据的事实，给所有聚类结果的评测带来了许多挑战。下面是非监督方法不得不面对的一些最重要的问题：

- 可能并不知道用于训练数据的实例的正确涵义。
- 几乎可以肯定聚类中所包含的训练实例具有各种各样的涵义。
- 聚类的数目与被排歧目标词的涵义数目几乎总是不等的。

Schütze的实验(Schütze, 1992, 1998)构建了将非监督聚类用于词义排歧的完全的应用系统。尽管实际技术相当复杂，但是核心方法仍然是非监督聚类。Schütze的结果表明对于近似的二元区分，非监督技术的结果能够逼近那些监督和自举方法的结果，大部分实例的结果接近对比结果的90%。与大部分监督方法一样，该方法也只是对少数单词样例进行了测试。

17.2.2 基于词典的方法

所有这些方法的主要缺陷都是规模问题，都需要大量的工作为词典中的每个歧义条目生成一个分类器。由于这个原因，使用这些方法的大部分实验所报告的结果都来自2~12个词典条目（Ng and Lee(1996)的研究工作是一个值得注意的特例，给出了对121个名词和70个动词的排歧结果）。把这些方法中的任何一个用于处理一种语言中的所有歧义单词，都将是一项浩大的工程。因此，大规模排歧的尝试都转向利用第16章曾讲述的机器可读词典（machine readable dictionary）。在这类方法中，词典不但提供了构建涵义标注系统所需的意义，而且提供了将使用的目标涵义。

这种方法的首次实现归功于Lesk(1986)。在该方法中，首先从词典中检索出排歧单词的所有涵义定义，然后把这些涵义中的每一个与上下文中的所有其他单词的词典定义相比较。与这些上下文单词之间具有最多重叠的涵义被选为正确的涵义。注意，在该方法中上下文单词的各种涵义是简单地混在一起计算的。

为了更具体地说明该方法，我们来研究Lesk的例子，在已知*pine*和*cone*的定义的情形下为短语*pine cone*中的单词*cone*选择正确的涵义。

pine 1 kinds of evergreen tree with needle-shaped leaves
2 waste away through sorrow or illness
cone 1 solid body which narrows to a point
2 something of this shape whether solid or hollow
3 fruit of certain evergreen trees

在这个例子中，Lesk的方法将选择cone[3]作为正确的涵义，因为这个条目中的两个单词*evergreen*和*tree*与*pine*条目中的单词相重叠，而其他条目中的单词都不与*pine*的定义中的单词相重叠。对于从Austen的Pride and Prejudice（《傲慢与偏见》）和一个AP新闻专线的文章中选取的较短的样例，Lesk报告的精度为50%~70%。

该方法的主要问题在于：目标单词的词典条目相对较短，可能并不能提供足够的资料以生成理想的分类器，因为用于上下文的单词和它们的定义必须在与正确的涵义定义所包含的单词具有直接重叠时才会有用①。对这个问题的一种修正方法是，扩充分类器中所用单词的列表，把一些相关的但在单独涵义定义中并没有出现的单词也包括进来。这可以通过加入那些在定义中用到了目标单词的单词来得以实现。例如，单词*deposit*在*bank*的American Heritage Dictionary(Morris, 1985)的定义中并没有出现，但是*bank*却出现在*deposit*的定义中出现。因此，bank的分类器可以将deposit扩充进来作为一个恰当的特征。

当然，仅仅知道*deposit*与*bank*相关并不会带来很大帮助，因为我们不知道是与*bank*的哪个涵义相关。特别是，为了把*deposit*作为一个特征，不得不了解它的定义中所用的是*bank*的哪个涵义。幸运的是，许多词典和辞典的条目中都包括了一个称为学科代码（subject code）的标签，它大致对应于主要的概念范畴。例如，在《现代英语朗文词典》（Longman's Dictionary of Contemporary

① 的确，Lesk(1986)也注意到所用系统的性能似乎与词典条目的长度有关。

English，简称 LDOCE)(Procter, 1978)的 bank 条目中，用学科代码 EC（Economics）表示 *bank* 的金融领域的涵义。已知这种学科代码，就能够猜测出带有学科代码 EC 的扩充术语应该与 *bank* 的这个涵义而不是任何其他涵义相关。Guthrie et al.(1991)报告的结果是，对于细粒度的 LDOCE 区分的正确率为 47%，而对较粗的区分的正确率为 72%。

17.3　信息检索

信息检索是一个处于成长期的领域，它涵盖了与各种媒体的存储和检索有关的广泛主题。本节关注的是文档的存储以及随后对用户查询信息的检索，特别是普遍采用的基于单词的索引和检索方法。目前大部分信息检索系统基于对语义的组合性原则的极端诠释，即文档的意义只取决于它所包含的单词。再次回顾第 16 章开头所引用的疯狂的帽匠的对话，对于这些系统来说，*I see what I eat* 和 *I eat what I see* 的意义完全相同，因为两个句子包含的单词完全相同。组成句子进而组成文档的单词的顺序和单词群组对它们的意义不起任何作用。因为忽略了句法信息，这些方法常常被称为“单词包”（bag of word）方法。

在继续讲述之前，我们首先介绍几个新的术语。在信息检索中，文档（document）泛指系统中索引以及检索的文本单元。根据应用的不同，文档可以指像新闻文章或百科全书条目那样的字面概念，也可以指像段落或句子那样的小文本单元。在基于网络的应用中，它也可以指网络页面、部分页面或整个网站。文档集（collection）表示用于满足用户需要的一组文档。检索词（term）指文档集中出现的词汇项，也可能包括短语。最后，查询条件（query）表示由一组检索词表达的用户查询信息。

我们将要详细研究的特别的信息检索任务被称为特定检索（ad hoc retrieval）。在这个任务中，假定独立用户向检索系统发出查询请求，系统则会返回一个按可能顺序排列的大概有用的文档集。其他基于词汇的相关信息检索任务将在 17.4 节进行讨论。

17.3.1　向量空间模型

在信息检索的向量空间模型（vector space model）中，文档和查询条件被表示为说明检索词出现的特征向量(Salton, 1971)。更准确地说，它们被表示为由文档集里出现的检索词组成的特征向量，每个特征值表示在指定的文档中某个特定检索词的出现或缺席。这些向量可以表示为：

$$\vec{d}_j = (t_{1,j}, t_{2,j}, t_{3,j}, \cdots, t_{N,j})$$

$$\vec{q}_k = (t_{1,k}, t_{2,k}, t_{3,k}, \cdots, t_{N,k})$$

符号$\vec{d}_j$和$\vec{q}_k$表示特定的文档和查询条件，而各种t特征表示这N个检索词作为整体出现在文档集里。我们首先考虑这些特征的值为 1 或 0 时，即说明某个检索词出现或缺席于一个问答或查询条件时的情形。对于这种方法来说，确定一个文档和查询条件的相关性的简单方法是确定它们共同的单词数。这可以通过下面的简单相似性计算公式来完成：

$$sim(\vec{q}_k, \vec{d}_j) = \sum_{i=1}^{N} t_{i,k} \times t_{i,j} \tag{17.17}$$

在这个方程中，查询条件向量$\vec{q}_k$和文档向量$\vec{d}_j$之间的相似性是通过对它们的共同单词数进行简单相加来衡量的。

显然，采用二元值的特征会带来一个问题，我们无法捕捉到对于文档的意义来说一些检索词比另一些检索词更重要这一事实。一个有用的一般化原则是，用数值的权重（weight）代替 0 和 1，以说明各种检索词在特定文档和查询条件中的重要性。那么，我们可以将向量一般化为：

$$\vec{d}_j = (w_{1,j}, w_{2,j}, w_{3,j}, \cdots, w_{n,j})$$

$$\vec{q}_k = (w_{1,k}, w_{2,k}, w_{3,k}, \cdots, w_{n,k})$$

把文档特征化为检索词权重的向量，容许我们把整个文档集看成是一个权重矩阵，$w_{i,j}$表示检索词i在文档j中的权重。这个权重矩阵被特别地称为检索词乘文档矩阵(term-by-document matrix)。按照这种观点，矩阵的列表示文档集里的文档，而矩阵的行表示检索词。

把在该模型中用于表示文档(和查询条件)的特征看成是多维空间中的一组维度是有用的。相应地，作为那些特征的值的权重，可用于确定文档在该空间中的位置。当用户的查询条件被转换为表示空间中的一点的向量时，则可以认为位于该查询条件附近的文档与它的相关性大于位于远处的文档。

把文档和查询条件特征化为向量的形式为特定检索系统提供了所有基本构件。一个文档检索系统可以接受用户的查询条件，生成它的向量表示，然后把它与表示所有已知文档的向量进行对比，最后给出排序后的结果。该结果是一个按与查询条件的相似性排序的文档列表。

研究图 17.3 所示的该方法的例子。该图给出的是一个对应于检索词 *speech*，*language* 和 *processing* 的三维简单空间。该空间中的 3 个向量分别表示由第 1 章、第 7 章和第 13 章的章节标题文字所组成的文档，分别以 Doc1，Doc7 和 Doc13 表示。如果使用原始的文档词频作为权重，则 Doc1 由向量(1, 2, 1)表示，Doc7 由向量(6, 0, 1)表示，Doc13 由向量(0, 5, 1)表示。图中清楚地表明，这个空间捕捉到了这几章内容如何相关的某些直觉。第 1 章的内容由于是综合的，与第 7 章和第 13 章都还算类似，而第 7 章和第 13 章由于讲述的是不同的主题，它们的距离很远。

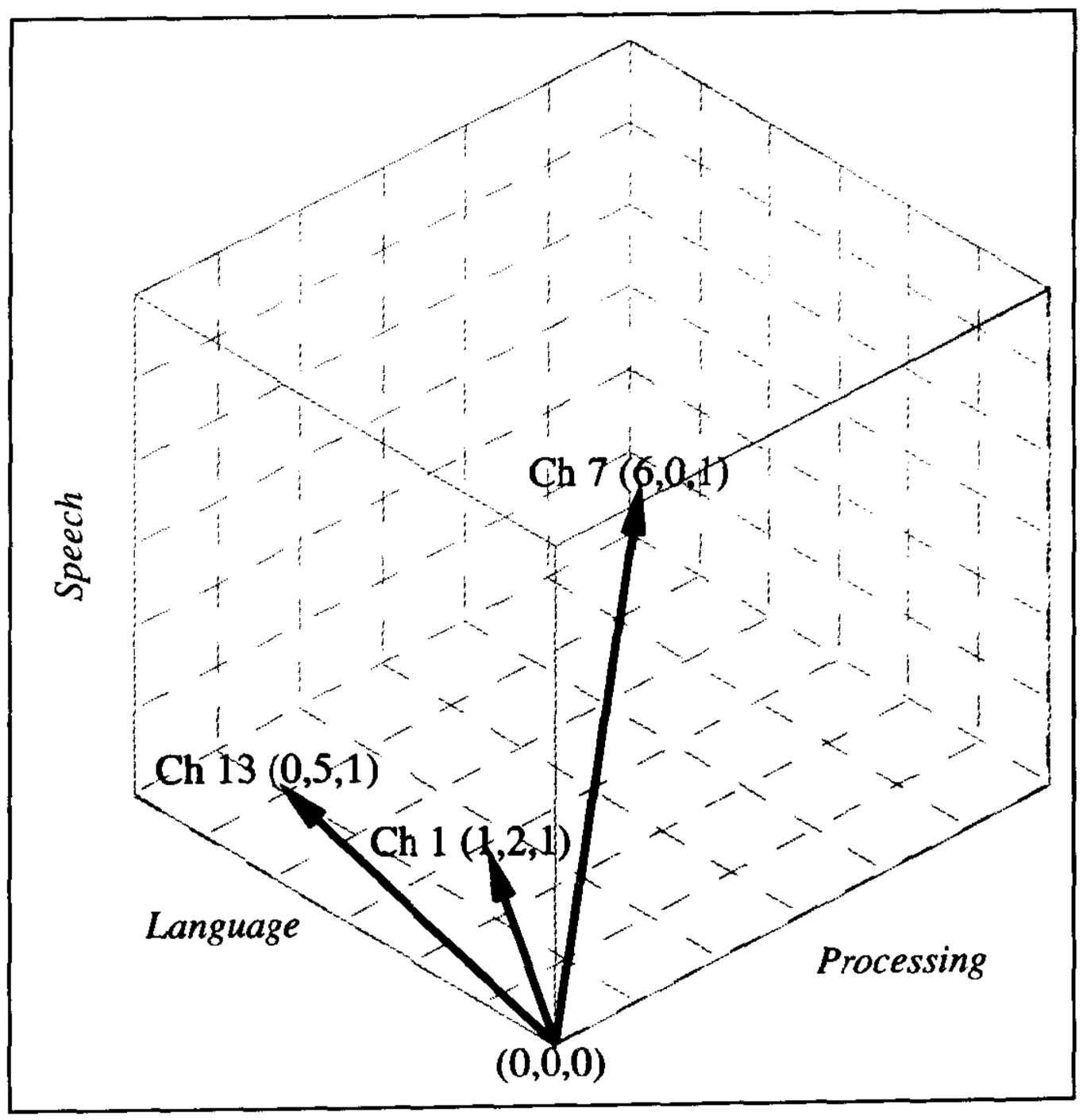


图 17.3　由第 1 章、第 7 章和第 13 章的章节标题文字组成的文档所表示的简单三维向量空间

遗憾的是，在向量空间的这个实例中过于强调每个文档的各个坐标的绝对值。例如，Doc7 的 *speech* 维度的重要性并不在于它的值是6，而是它是文档意义的最主要的贡献者。类似地，Doc1 的

具体值1，2和1并不重要，重要的是这3个维度具有大致相似的权重。例如，假定一个新文档的权重为3，6和3，则尽管检索词权重的大小不同，仍可以感觉到它与Doc1非常接近。

我们可以通过归一化（normalizing）文档向量来实现这个结果。简单地说，“归一化”这个术语指把所有向量转换为标准长度。通过把向量的每个维度都除以向量的总长度（定义为$\sum_{i=1}^{N} w_i^2$），可以将它转换为单位长度。这实际上消除了空间中一个文档的确切长度的重要性，而强调文档向量相对原点的方向。

将该技术用于我们的3个样例文档，则可以得到下面的检索词乘文档矩阵A，这里的列表示Doc1，Doc7和Doc13，而行表示检索词*speech*，*language*和*processing*：

$$A = \begin{pmatrix} 0.41 & 0.81 & 0.41 \\ 0.98 & 0 & 0.16 \\ 0 & 0.98 & 0.19 \end{pmatrix}$$

可以验证得到：采用这种方案，Doc1和假定的(3, 6, 3)文档的归一化向量最终是相同的。

现在回到确定向量之间相似性的主题。将早先给出的相似性公式中的二元值更新为数值的权重，则得到下面的方程：

$$sim(\vec{q}_k, \vec{d}_j) = \vec{q}_k \cdot \vec{d}_j = \sum_{i=1}^{N} w_{i,k} \times w_{i,j} \tag{17.18}$$

这个方程表示向量之间的点乘（dot product）。通常，两个向量之间的点乘并不用做相似性的度量，因为它对于各种维度的绝对大小过于敏感。然而，已经归一化的向量之间的点乘具有一个有用并且直观的解释：它计算的是两个向量之间夹角的余弦。当两个文档相同时，它们的余弦值是1；而当它们正交时（没有共同的检索词），它们的余弦值则是0。

注意，如果由于某些原因向量没有被存储为归一化的形式，那么该归一化可以直接引入下面的相似性计算公式：

$$sim(\vec{q}_k, \vec{d}_j) = \frac{\sum_{i=1}^{N} w_{i,k} \times w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,k}^2} \times \sqrt{\sum_{i=1}^{N} w_{i,j}^2}} \tag{17.19}$$

当然，在文档集相对稳定并且需要对许多查询条件进行处理的情形下，一次性归一化文档向量并存储它们，比在相似性计算时引入归一化更合理。

让我们来讨论在一些小例子背景下相似性度量是如何进行的。研究下面特意选出的由*speech*，*language*和*processing*组成的查询条件。将这个查询条件转换为一个向量并把它归一化，则可以得到向量(0.57, 0.57, 0.57)。计算该向量和3个文档向量之间的余弦，可以得出它与Doc1最接近（余弦值是0.92），接着是Doc13（余弦值是0.67），最后是Doc7（余弦值是0.65）。这个顺序与我们关于该查询条件和这些文档之间的关系的直觉是十分接近的。

现在来研究一个较短的只包含检索词*speech*和*processing*的查询条件。处理这个查询条件，以生成归一化的向量(0.70, 0, 0.70)。计算这个向量和我们的文档之间的余弦，现在Doc7最接近（余弦值是0.80），接着是Doc1（余弦值是0.58），而Doc13最远（余弦值是0.13）。

17.3.2　检索词加权

在实践中，对于表示文档和查询条件的向量，其检索词权重的指派方法对于一个检索系统的性能具有非常大的影响。已经证实有两个因素对获取有效的检索词权重十分关键：检索词在单个文档中出现的频率和检索词在整个文档集里的分布。我们从简单的概念开始讲述，如果一些检索词在一个文档中经常出现，那么它们就比那些在该文档中出现次数较少的检索词更能表现出该文档的意

义，因此应该给予较高的权重。这个因素的最简单形式被称为词频（term frequency），即检索词在一个文档中出现的原始频率(Luhn, 1957)。

第二个因素所考虑的是检索词在整个文档集里的分布。只在少数几个文档中出现过的检索词对于区分这几个文档与文档集里的其他文档非常有用。同时，在整个文档集里经常出现的检索词很难用于区分文档。因此，我们需要的是一种能够优先考虑在较少的文档里出现过的检索词的计算方法。分数N/n_i恰好给出了这种计算，其中N是文档集里的文档总数，n_i是文档集里出现过检索词i的文档数。出现该检索词的文档越少，该检索词的权重就越大。最小的权重1指派给在所有文档中出现过的检索词。由于许多文档集的文档数目非常巨大，通常我们利用对数函数对该值进行同比缩小，即下面的倒排文档频率（inverse document frequency）的检索词权重(Sparck Jones, 1972)：

$$\text{idf}_i = \log\left(\frac{N}{n_i}\right) \tag{17.20}$$

将词频因素与上面的因素相结合则生成了称之为tf·idf的加权方案：

$$w_{i,j} = \text{tf}_{i,j} \times \text{idf}_i \tag{17.21}$$

即，检索词i在文档j的向量中的权重等于它在j中的总频率乘以整个文档集里它的倒排文档频率的对数。经过一些细微调整，这个加权方案几乎被所有向量空间检索模型用于指派文档检索词的权重。

尽管我们对文档和查询条件采用的是相同的表示，但是并不清楚是否应该对它们采用相同的加权方案。在许多特定的检索环境中，比如网络搜索引擎，用户的查询条件毕竟与文档有很大不同。例如，AltaVista搜索引擎的超大规模查询条件集（确切的数目为1 000 000 000）的分析表明，平均查询条件长度约为2.3个词(Silverstein et al., 1998)。在这种环境下，查询条件的原始词频就很难作为一个有用的因素。因此，Salton and Buckley(1988)建议以下面的公式作为查询条件的加权公式，$\max_j \text{tf}_{j,k}$表示文档k中最常出现的检索词的频率。

$$w_{i,k} = \left(0.5 + \frac{0.5\text{tf}_{i,k}}{\max_j \text{tf}_{j,k}}\right) \times \text{idf}_i \tag{17.22}$$

方法论探讨：信息检索系统的评测

信息检索系统是通过相关性（relevance）的概念来评测的，也就是说，通过人来判断某个文档与查询条件相关。系统检索相关文档的能力是通过召回率（recall）标准来评估的（第15章曾提及）：

$$\text{召回率} = \frac{\text{系统返回的相关文档数}}{\text{文档集里相关文档的总数}}$$

当然，通过简单地返回文档集里的所有文档，一个系统可以获得100%的召回率。系统的精度表示基于一个特定查询条件返回的文档中有多少确实是相关的，这可以通过下面的精度（precision）标准加以评估。

$$\text{精度} = \frac{\text{返回的相关文档数}}{\text{返回的文档数}}$$

大部分系统并不直接给出相关判断，而是针对查询条件对文档集进行排序，这一事实使得计算这些系数变得很复杂。为了处理这种情况，我们对输出指定一组分界点，并对排在分界点以上的那些文档计算平均精度。我们也可以指定一组不同的召回率级别，并在这些级别上对平均精度进行计算。后一种方法称为精度-召回率曲线，如图17.4所示。从图中可以看出，比较两个系统的性能是很困难的。在如图所示的对比中，一个系统在召回率处于较高和较低的级别时精度较高，而另一个系统在召回率处于中间级别的区域具有较高的精度。对这些曲线的另一个可选度量标准是把召回率和精度结合为一个单独的值，也就是15.5节曾介绍的F系数。

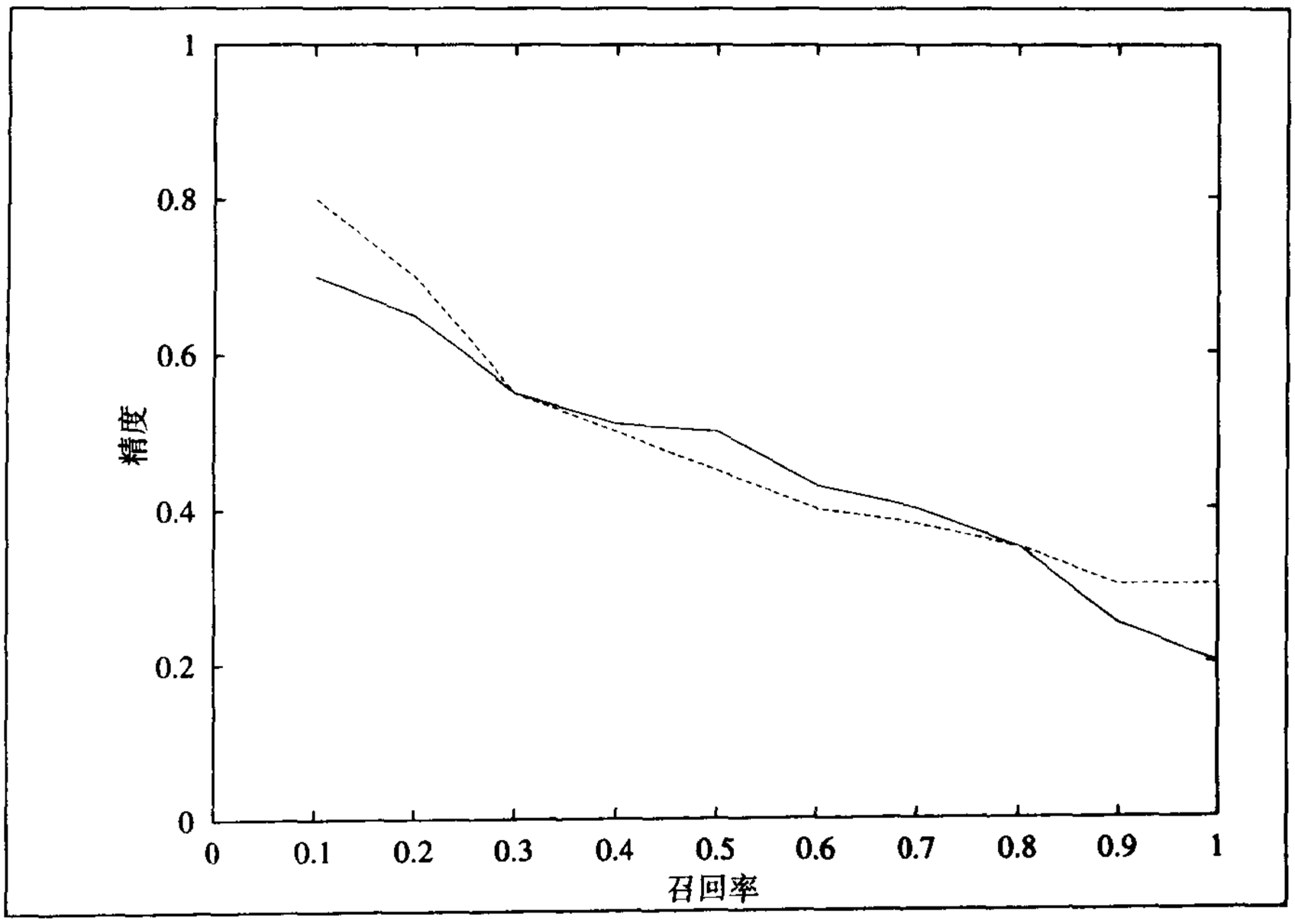


图 17.4　两个假定系统的精度 - 召回率曲线。根据指定的召回率级别分别计算出返回文档集的平均精度，以绘制曲线。例如，对这两个系统的返回文档集分别划出分界点，以保证召回率为 30%，然后分别计算这两个系统的平均精度，在本例中它们恰好都是 55%

美国政府资助的文本检索会议（Text REtrieval Conference，简称 TREC）评测给信息检索的各种任务和技术提供了严格的实验平台。与 MUC 评测一样，TREC 提供了大量用于训练和评测的文档集以及统一的评分系统。训练资源是由查询条件（在 TREC 中称为主题，topic）和相关判断集以及文档集共同组成的。Voorhees and Harman(1998)对最近的会议给出了详细的介绍。所有会议的详情可以参阅 NIST（National Institute of Standards and Technology）网站的 TREC 页面。

17.3.3　检索词的选择和创造

到目前为止，我们还是假定以常规的单词作为文档集里文档的索引单元。对这个假设的两个变化分别涉及词干处理（stemming）和禁用词表（stop list）的使用。

词干的概念把我们又带回了第 3 章中形态分析的话题。词干处理所面临的基本问题是，是否应该把词汇项的形态变体单独列出、单独计算，还是应该将它们分解为统一的词根形式。例如，不进行词干处理，术语 *process*，*processing* 和 *processed* 将被单独处理，在检索词乘文档的矩阵中也具有不同的词频；而如果进行词干处理，则它们将被合并为一个单独的检索词，只有一个相加后的频率值。采用词干处理的主要优点是容许特定的查询词与包含该词的任何形态变体的文档相匹配。第 3 章所描述的 Porter 词干处理器(Porter, 1980)常常被用于英文文档集的检索。

该方法带来的一个问题是它忽略了一些有用的特征。例如，研究将 Porter 词干处理器用于单词 *stocks*（股票）和 *stockings*（长袜）。在本例中，Porter 词干处理器将这些表层的形式处理为单个词根 *stock*。自然这会导致关注 *stock prices*（股票价格）的查询条件返回关于 stockings（长袜）的文档，而针对 *stockings*（长袜）的查询也将返回有关 *stock*（股票）的文档[①]。从更技术的角度看，词干处理可能通过找到带有形态上与查询条件相关的检索词的文档来提高检索的召回率，但是由于返

① 这个例子来自一个著名搜索引擎受到了公众的质疑，他们关于 *stock prices* 的查询结果却包含一些相当色情的站点（因为这些站点包含大量的检索词 *stockings*）。作为回应，发言人宣布他们将努力解决这个奇怪的单词问题。

回了语义上无关的文档也可能降低精度。正是由于这个原因，目前几乎没有网络搜索引擎采用词干处理。Hull(1996)给出了一系列的实验以研究词干处理的影响。

另一个常用技术是禁用词的使用，它所关注的问题是什么样的单词才能被容许索引。禁用词表（stop list）是一个高频词的简单列表，这些词不能用于表示文档和查询条件。通常采用禁用词表的动机有两个：高频的封闭词类的检索词被认为几乎不具有语义权重，因此对检索的帮助很小；排除这些词可以大大节省用于建立检索词和包含它的文档之间的映射的倒排索引文件空间。采用禁用词表的不利方面是很难查找包含禁用词的短语。例如，按照 Frakes and Baeza-Yates(1992)给出的从 Brown 语料库获得的常用禁用词表，短语 *to be or not to be* 将被转换为短语 *not*。

17.3.4 同形关系、多义关系和同义关系

因为向量空间模型只基于简单的检索词的应用，所以研究各种词汇语义现象对模型的可能影响是有意义的。研究包含具有 *tooth* 和 *dog* 这两个涵义的单词 *canine* 的查询条件。包含 *canine* 的查询条件将被判断为与使用这两个涵义中的任何一个的文档相似。然而，用户很可能只对其中一个涵义感兴趣，包含另一个涵义的文档将被判为无关。因此，同形关系和多义关系可能导致系统返回与用户所需信息无关的文档，从而造成精度的降低（reducing precision）。

再研究包含词位 *dog* 的查询条件。该查询条件将被判断为与常常出现单词 *dog* 的文档相似，但是可能无法找出使用像 *canine* 这样的同义词的文档，以及使用像 Malamute（爱斯基摩狗）这样的下位词的文档。因此，同义关系和上下位关系可能导致系统错过相关文档，从而造成召回率的降低（reducing recall）。

注意，武断地称多义关系导致精度的降低，同义关系导致召回率的降低都是不准确的，因为精度和召回率都是相对于一个固定的分界点的（参见“方法论探讨：信息检索系统的评测”）。由于多义关系导致处于分界点以上的每个无关文档都会占用固定数量的返回文档集里的一个位置，这可能将一个相关文档推向分界点以下，从而导致召回率的降低。类似地，因为同义关系错失了一个相关文档，所以一个无关文档可能占据返回文档集的这个位置，潜在地也可能降低精度。

这些讨论很自然地导出一个问题：词义排歧是否对信息检索有帮助？目前对这个问题的证据是不一致的，一些实验报告称使用词义排歧可以提高检索性能(Schütze and Pedersen, 1995)，而另一些报告称没有提高检索性能，甚至降低了检索性能(Krovetz and Croft, 1992; Sanderson, 1994; Voorhees, 1998)。

17.3.5 改进用户的查询条件

改善信息检索系统性能的最有效方法之一是寻找改进用户的查询条件的方法。本节所介绍的这些技术都已经被证明对于信息检索任务具有不同程度的效果。

在向量空间模型中，改善信息检索系统性能的最有效的单个方法是使用相关反馈（relevance feedback）(Rocchio, 1971)。在该方法中，用户向系统提交一个查询条件，系统返回给用户少量的检索结果文档，然后要求用户指定哪些文档是满足其需求的。接着，根据用户指定的相关和不相关文档中的检索词的分布，重组用户原始的查询条件。将重组后的查询条件作为一个新的查询条件提交给系统，并给用户返回新的检索结果。通常该技术的第一次循环就能够对检索性能带来极大的改善。

该技术实现的形式化基础直接来自向量模型的一些基本的几何学的直觉。特别是，我们希望将表示用户原始查询条件的向量推（push）向已经被发现是相关的文档，并且推离已经被判断为无关的文档。这可以通过加入一个表示原始查询条件的相关文档的平均向量，并减去一个表示无关文档的平均向量来实现。

更形式化地，让我们假设 $\vec{q}_i$ 表示用户的原始查询条件，R 是从原始的查询条件返回的相关文档数，S 是无关文档数，并且 $\vec{r}$ 和 $\vec{s}$ 分别表示在相关文档集和无关文档集里的文档。另外，假定 β 和 γ 的范围是从 0 到 1，并且 $\beta + \gamma = 1$。已知这些假设，则可用下式表示一个标准的相关反馈的更新公式：

$$\vec{q}_{i+1} = \vec{q}_i + \frac{\beta}{R}\sum_{j=1}^{R}\vec{r}_j - \frac{\gamma}{S}\sum_{k=1}^{S}\vec{s}_k$$

在该公式中，因子 β 和 γ 表示的参数可以通过实验加以调整。从直觉上看，β 表示新向量应该推向相关文档多远，而 γ 表示应该推离无关文档多远。根据 Salton and Buckley(1990)的报告，当 $\beta = 0.75$ 和 $\gamma = 0.25$ 时结果较好。

我们应该注意到，采用相关反馈的系统的评测相当复杂。特别是，由第一次重组后的查询条件所产生的结果常常会有很大改善。这并不令我们感到意外，因为在第一轮利用了用户告诉系统相关的文档。避免这种性能上的夸大的最好方法是只计算剩余文档集（residual collection）的精度和召回率，剩余文档集是指从原始文档集里去除任何一轮提交给用户判断的文档后剩余的文档集。通常这使得系统的原始性能低于第一次查询条件时所获得的性能，因为现在最相关的文档被去除了。然而，当比较不同的相关反馈机制时该技术就非常有效。

另一种可供选择的改进查询条件的方法着眼于组成查询条件向量的检索词，而不是查询条件向量本身。在查询扩展（query expansion）中，用户的原始查询条件被扩展，以包含与原始检索词相关的一些检索词。典型的实现方法是从文档集里与用户的原始检索词高度相关的词表中增加一些检索词。虽然这种高度相关的检索词的词表常常被称为同义词词典或类属词典（thesaurus），但它是基于相关性而不是同义关系的，它只是被不准确地与标准术语联系在一起。

遗憾的是，通常像类属词典这样的可利用资源对于大部分文档集都是不适合的。因此，需要借助类属词典生成（thesaurus generation）方法，从文档集的全部或部分文档中自动生成一个基于相关性的词典。采用检索词聚类（term clustering）是类属词典生成方法中最主流的方法之一。回顾前面提到的检索词乘文档的矩阵的特征：矩阵的列表示文档，矩阵的行表示检索词。因此，在词典生成中，矩阵的行可以被聚类以形成同义词集，然后这个同义词集被加入查询条件以提高系统的召回率。

通常该技术有两种实现方法：可以从文档集里一次性地生成一个类属词典(Crouch and Yang, 1992)，或者从原始查询条件返回的文档集里动态生成一个类似同义词的检索词集(Attar and Fraenkel, 1977)。注意，第二种方法需要花费更多精力，因为实际上对每个查询条件返回的文档都要生成一个小的类属词典，而不是对整个文档集生成一个类属词典。

17.4　信息检索的其他任务

正如早先所指出的，特定检索并不是信息检索中惟一的基于单词的任务。其他一些重要任务包括文档分类、文档聚类、文本分段和文本摘要等。

文档分类（document categorization）任务是指把新文档指派给一个预先设定的文档类别的处理。在这个任务中，关键是找出属于某个类别的文档的有用特征，以生成分类器。尽管这可以通过手工来完成，但标准方法是采用监督机器学习方法，特别是可以通过已标注正确类别的文档集的训练来获取分类器。在17.2节中提到的词义排歧的监督学习方法也可以用于该任务。当分类的目的是把一个文档传送给一个用户或一组兴趣相同的用户时，通常把它称为分发（routing）。术语“过滤”

(filtering)用于一种特殊的情形，其中分类任务是指接收或拒绝一个文档，比如用于屏蔽垃圾邮件的电子邮件过滤器。

分类任务假定已有一个现有文档分类体系或聚类体系。相反地，文档聚类的任务就是为给定的文档集生成或找到一个合理的聚类体系。与词义排歧的情形相同，一个合理的聚类被定义为能够使类内文档具有最大相似性，而类间文档具有最小相似性的分类体系。在特定检索背景下采用该技术的两个主要动机是：效率和聚类假设。

效率动机源于当前文档集里巨大的文档数。回顾一下，上一节描述的检索方法需要每个文档与文档集里的每个文档进行对比。如果一个文档集可以被分为N个概念一致的聚类，则查询条件首先与这N个聚类的表示进行对比，然后只对排列在最前面的一个或几个聚类采用传统的检索，这样就大大节省了与所有距离较远的聚类进行比较而带来的耗费。

聚类假设(cluster hypothesis) (Jardine and van Rijsbergen, 1971)进一步强化了这个论点，宣称从一个聚类后的文档集进行检索不但会提高检索效率，而且会改善整体的检索性能，即提高召回率和精度。该假设背后的基本概念是根据主题划分文档，相关文档会在同一聚类中被发现，并且由于无关文档位于那些不参与检索的聚类中，从而避免了无关文档。尽管这个假设似乎很合理，但它并不被所有实验结果所支持。实验结果随所采用的聚类算法和文档集的不同而有很大不同(Willett, 1988; Shaw et al., 1996)。

聚类方法的另外一个有前景的应用是对响应用户查询条件而返回的文档，而不是整个文档集进行聚类。Hearst and Pedersen(1996)证实该技术提供了聚类假设所承诺的许多优点。

在文本分段(text segmentation)中，较长的文档被自动分成语义上一致的较短的语段(chunk)。这对于拥有大量的具有各种主题的长文档的领域非常有用。文本分段既可以用于低于文档层的检索，也可以直观地指导用户查看返回文档的相关部分。对较长文档中的子部分，分段算法常常采用类似向量的表示。一个文档的具有相似余弦值的相邻子部分比具有较大差别的余弦值的相邻子部分更有可能是关于相同的主题的。大致地说，这种相邻文档片断之间的相似性的不连续性可以用于将较长的文档分成子部分(Salton et al., 1993; Hearst, 1997)。

文本摘要(text summarization) (Sparck Jones, 1997)是指生成一个原始文档的简短且概要的版本的处理。该问题通常有两种处理方法。在基于知识(knowledge-based)的方法中，首先对原始文档进行语义分析，以生成该文本意义的表示，然后把该表示传给文本生成器，以生成能够表达原文的要点并满足长度限制的概要文本。有关文本生成的详细内容将在第20章给出。在基于选择的摘要(selection-based summarization)中，首先根据非常简单的词频和话语结构启发式信息对所有句子指派代表句子重要程度的权重值，然后通过确定阈值来生成摘要文档，权重值高于该阈值的所有句子的长度信息相加应该满足规定的摘要长度。

17.5 小结

本章探讨了词汇语义处理的两个主要领域：词义排歧和信息检索。

- 词义排歧(word sense disambiguation)系统为具有上下文环境的词例从预先设定的涵义集中指派一个涵义。
- 基于选择限制(selectional restriction-based)的方法既可以用于谓词也可以用于论元的词义排歧，但是需要大量的有关角色填充者的语义角色限制和层级类型的信息。
- 词义排歧的机器学习方法使得自动生成鲁棒的词义排歧系统成为可能。

- 监督（supervised）方法利用已标注正确涵义的文本集来训练分类器。
- 自举（bootstrapping）方法容许采用训练语料极少的监督方法。
- 非监督（unsupervised）基于聚类的方法试图从未标注的文本中找到词义的表示。

- 机器可读词典（machine readable dictionary）为能够覆盖广泛领域的词义排歧系统的生成提供了便利。
- 信息检索的主流模型都以单词包（bag of word）来代表文档和查询条件的意义。
- 在向量空间模型（vector space model）中，文档和查询条件都被看成是多维空间的一些向量。通过向量之间夹角的余弦可以对文档和查询条件或文档和其他文档之间的相似性进行度量。
- 可以利用相关反馈（relevance feedback）或基于类属词典的查询扩展（query expansion）重组查询条件而对其加以改善。

17.6　文献和历史说明

词义排歧的根源可以追溯到早期的一些计算机应用。通过一个单词周围的小窗内的上下文对其进行排歧的思想是由 Warren Weaver(1955)首次在机器翻译的背景下提出的。早期首次提出的其他思想包括利用词典排歧(Masterman, 1957)、用于排歧的Bayes模型的监督训练(Madhu and Lytel, 1965)以及利用聚类进行词义分析(Sparck Jones, 1986)等。

在面向人工智能的自然语言处理系统的背景下，涌现出了一大批排歧方面的研究工作。大部分此类自然语言处理系统都表现出了一些形式的词义排歧能力，这些成果使得词义排歧成为工作的核心。其中最有影响的成果包括Quillian(1986)和Simmons(1973)的语义网，Wilks(1975c, 1975b, 1975a)的优选语义（Preference Semantics），以及 Small and Rieger(1982)和 Riesbeck(1975)的基于单词的理解系统。Hirst 的 ABSITY 系统(Hirst and Charniak, 1982; Hirst, 1987, 1988)利用一种基于语义网的被称为“标记传递”（marker passing）的技术，代表了此类系统的最高水平。与这些主要符号方法一样，大部分词义排歧的连接方法依赖于带有手工编写表示方式的小规模词典(Cottrell, 1985; Kawamoto, 1988)。

应该指出的是，词义排歧的许多研究工作是在认知科学和心理语言学领域开展的。在该领域中通常使用另外一个名字，即词义消解（lexical ambiguity resolution）来表示词义排歧。Small et al.(1988)给出了这个视角的各种论文。

最早在词义排歧中采用鲁棒的经验方法的是Kelly and Stone(1975)，对英语中的1790个歧义单词手工编写了一系列排歧规则。Lesk(1986)首次将机器可读词典用于词义排歧。新墨西哥州立大学利用《现代英语朗文词典》（LDOCE）所做的是利用机读词典的最详尽的研究之一。在文献 Wilks et al.(1996)中有对此的详细描述。有关词典涵义的粒度过于细小或缺少恰当组织的问题在文献Dolan(1994)以及 Chen and Chang(1998)中有详细论述。

对词义排歧的监督机器学习方法的当代研究始于 Black(1988)，将决策树用于排歧。由于这些方法需要大规模的已标注文本，人们转向研究自举方法(Hearst, 1991; Yarowsky, 1995)。Ng and Lee(1996)以及McRoy(1992)研究了如何加权以及合并用于许多鲁棒系统的不同来源的证据。有关非监督方法的研究相对较少。最早尝试把聚类用于单词涵义研究的是 Sparck Jones(1986)。Zernik(1991)成功地将一个标准的信息检索聚类算法应用于该问题，并且给出了关于检索性能改善的评测。较新的详尽聚类研究可以在文献 Pedersen and Bruce(1997)以及 Schütze(1997, 1998)中找到。

注意，在所有这些引人瞩目的努力中，只有下面3个涉及对一个句子中的所有单词歧义进行排歧。文献 Kelly and Stone(1975)描述的系统对句子进行多次处理，以便于在后续处理中利用容易排

歧的结果。Cowie et al.(1992)采用模拟退火模型并行查找希望的涵义集。Veronis and Ide(1990)采用从机器可读词典中自动构建的神经网络的抑制和激励来进行计算。

Ide and Veronis(1998)给出了词义排歧研究目前的历史及现状的全面综述。Ng and Zelle(1997)从机器学习的视角给出了一个更集中的回顾。Wilks et al.(1996)给出了一批词典和基于语料库的实验，并对一些早期研究工作进行了详细描述。

大家普遍认为是Luhn(1957)首次推动了基于文档内容的全自动文档索引。多年来，基于Cornell大学Salton的SMART项目(Salton, 1971)衍生出了信息检索的许多最重要的概念，包括向量模型、检索词加权方案、相关反馈以及采用余弦作为相似性度量等。在检索词加权中采用倒排文档频率的思想归功于 Sparck Jones(1972)。相关反馈的最初思想归功于 Rocchio(1971)。向量空间模型的另一种可替换模型（本书没有涉及）是概率模型（probabilistic model）。概率模型的 Bayes 网络版最初由 Robinson and Sparck Jones(1976)证明是有效的，该模型也是被广泛使用的 INQUERY 系统(Callan et al., 1992)的原型。Crestani et al.(1998)是对信息检索中的概率模型的全面回顾。

聚类假设是由 Jardine and Van Rijsbergen(1971)提出的。Willett(1988)对该领域的主要成果给出了批评性的回顾。Mather(1998)给出了一个独立于算法的聚类度量标准，可用于对各种聚类算法的性能进行评测。有关文档分类及其相近的应用过滤和分发的论文可以在文献Lewis and Heyes(1994)中找到。有关分发的最近一个例子是 AT&T 的“How May I Help You?”任务，它的目的是把用户的提问分发到15个可能的类别，比如第三方付费，或对方付费电话等。一旦系统确定电话的分类，该电话将被转给合适的接线员。该任务的分类精度接近80%，尽管语音识别的单词正确率仅仅50%左右(Gorin et al., 1997)。

通常从两种视角对文本分段进行研究：基于话语结构的坚实理论的方法和基于词汇文本连贯的方法(Morris and Hirst, 1991)。Hearst(1997)描述了一种鲁棒的基于词汇连贯的向量模型的技术。我们将在第 18 章和第 20 章对基于坚实的话语模型的技术进行讨论。

文本摘要的研究始于Luhn(1958)的摘要自动生成。有关文本摘要的论文集可以参阅文献 Hovy and Radev(1998)。

向量空间模型的一个重要扩展被称为“潜在语义索引”（Latent Semantic Indexing，简称 LSI）(Deerwester et al., 1990)。为了从原始的检索词乘文档的矩阵中找出高阶秩，它以奇异值分解方法为手段来降低向量模型的维度。Berry et al.(1999)对向量模型中的降低维度的方法给出了有用的综述。尽管一开始LSI是作为一种信息检索方法的，但目前它已被使用于各种实际应用系统中，包括词典获取模型(Landauer and Dumais, 1997)、问答系统(Jones, 1997)以及最近的作文打分(Landauer et al., 1997)。

Baeza-Yates and Ribeiro-Neto(1999)中介绍了许多信息检索领域的最新发展趋势。文献Frakes and Baeza-Yates(1992)是对信息检索的更具体的介绍，并且包含了大量有用的C代码。较早期的经典文献包括 Salton and McGill(1983)和 van Rijsbergen(1975)。该领域的许多经典论文可以在 Sparck Jones and Willett(1997)中找到。当前的最新研究成果是通过 ACM 的信息检索特殊兴趣小组（SIGIR）来出版的。周期性的TREC会议论文集包括了由美国政府组织的标准评测结果。该领域最主要的期刊是 Journal of the American Society of Information Sciences，ACM Transactions on Information Systems，Information Processing and Management 以及 Information Retrieval。

第四部分　语用的计算机处理

语用学是对语言与使用环境之间的关系（的某些部分）的研究。使用环境包括像人和物这样的本体，因此语用学涉及如何将语言用于指示（以及回指）人和物的研究。使用环境也包括话语的上下文，因此语用学涉及话语结构的形成以及会话时听话人如何理解谈话对象的研究。本书的这个部分将研究所指判定（reference resolution）的算法、获取对白和会话话语结构的计算模型以及对话中话段的解释模型等，还将讨论建立会话智能代理（agent）时各种模型的角色以及这种会话智能代理的会话管理模块的设计，最后将介绍自然语言生成，特别是话语的功能。

第18章 话　　语

Gracie: Oh yeah...and then Mr. and Mrs. Jones were having matrimonial trouble, and my brother was hired to watch Mrs. Jones.
George: Well, I imagine she was a very attractive woman.
Gracie: She was, and my brother watched her day and night for six months.
George: Well, what happened?
Gracie: She finally got a divorce.
George: Mrs. Jones?
Gracie: No, my brother's wife.

George Burns and Gracie Allen in *The Salesgirl*[①]

到目前为止，本书集中讨论的都是出现于单词和句子层面的语言现象。毫无疑问，通常语言不是由孤立无关的句子组成的，而是由搭配在一起的相关句子群组成的。我们将这种句子群称为话语（discourse）。

你现在所读的章节本身就是一个话语的例子。实际上它是话语的一种特殊类型：独白（monologue）。独白是以一个说话人（这个术语可以用于表示作者）和一个听话人（这个术语也可以表示读者）为特征的。独白中的交流是单向的，即从说话人到听话人。

在读完本章后，也许你会和一个朋友一起谈论本章，这时是一种非常自由的交流。这种话语称为对话（dialogue）。在这种情形下，每个参与者轮流充当说话人和听话人。与典型的独白不同，对话通常由许多不同类型的交流行为组成：提问、回答、更正等。

最后，计算机系统自诞生之日起就不断地被人们研究来进行人机交互（human-computer interaction，简称HCI）。人机交互与普通的人与人的对话是不同的，部分原因在于目前计算机系统在参与自由无约束的会话方面仍存在局限性。能够进行人机交互的系统常常采用一些策略来约束会话，这些策略只容许在受限的解释背景下理解用户的话段。

尽管这三种话语形式具有许多话语处理的共同问题，但是它们各自的特点使得常常需要对它们采用不同的处理技术。本章将集中讲述常用于独白解释的技术，而对话解释和人机交互HCI的技术将在第19章中介绍。

话语层引起的现象在语言中是非常普遍的。研究例句（18.1）所示的话语。

John went to Bill's car dealership to check out an Acura Integra.
He looked at it for about an hour. （18.1）

---

① 译文如下，供读者对照参考：
Gracie：哦，是的……Jones先生和太太的婚姻出现了问题，我哥哥被雇来监视Jones太太。
George：喔，在我想像中她是一个非常迷人的女人。
Gracie：的确是，我哥哥对她日夜不停地监视了6个月。
George：那么，发生了什么事？
Gracie：她最终离婚了。
George：Jones太太？
Gracie：不，我哥哥的妻子。

代词 *he* 和 *it* 代表的分别是什么？读者无疑会很容易领会到 *he* 代表 John 而不是 Bill，*it* 代表 the Integra 而不是 Bill's car dealership。而在本章开头给出的交流中，George 似乎很难领会 Gracie 所说的 *she* 是指谁。

这两个例子有什么不同吗？借助什么方式能使得听话人很容易理解例句（18.1）？我们能够对这种处理方式建立计算模型吗？本章将回答这类问题。在 18.1 节中，我们将介绍解释所指语（referring expression）（例如代词）的方法。然后，在 18.2 节中会着手解决如何确立连贯话语的问题。最后，在 18.3 节中会阐述确定话语结构的方法。

因为语言中的话语层现象是普遍存在的，所以解决它们的算法对于各种应用都非常关键。例如，像 ATIS（航空旅行信息系统，见第 9 章）这样的与查询界面交互的对话理解系统常常包含代词和类似的表达。因此当用户对 ATIS 系统说出例句（18.2）时，

I'd like to get from Boston to San Francisco, on either December 5th or December 6th. It's okay if it stops in another city along the way. （18.2）

为了采取正确的行动，系统首先必须确定 *it* 表达的是用户想乘的航班。

类似地，信息抽取系统（见第 15 章）不得不经常从包含代词的话段中抽取信息。例如，如果一个信息抽取系统遇到下面的例句（18.3）：

First Union Corp is continuing to wrestle with severe problems unleashed by a botched merger and a troubled business strategy. According to industry insiders at Paine Webber, their president, John R. Georgius, is planning to retire by the end of the year. （18.3）

为了抽取正确的事件，系统必须正确地识别出 *their* 所指的是 *First Union Corp*（而不是 *Paine Webber*）。

同样地，许多文本摘要系统通过一个程序从原文中选择重要的句子，然后利用它们形成摘要。例如，考虑包含例句（18.3）的新闻文本。这类系统也许可以确定第二个句子（而不是第一个句子）足够重要，应该被包括在摘要中。但是，第二个句子中包含一个依赖于第一个句子的代词，如果不首先确定代词的所指名称就把第二个句子放入摘要，则很容易在摘要中产生对它的不同理解。类似地，自然语言生成系统（见第 20 章）中必须带有足够的代词化（pronominalization）模型，以产生连贯的可理解的话语。简言之，任何可以想到的语言处理应用都需要确定代词以及相关表达的指称。

18.1　所指判定

本节研究关于所指（reference）的问题，它是说话人使用类似例句（18.1）中的 *John* 和 *he* 这样的表达来指示名字为 *John* 的人的过程。在讨论之前需要先定义一些术语。用于实现所指的自然语言表达被称为所指语（referring expression），它指向的实体被称为所指的对象（referent）。因此，在例句（18.1）中 *John* 和 *he* 是所指语，而 John 是它们所指的对象。为了区分所指语和它们所指的对象，我们用斜体表示前者。作为一种方便的简化表达，我们有时说某个所指语指向某个对象，例如，我们可以说 *he* 指向 *John*。虽然如此，但是读者应该牢记其真正含义是：说话人进行了这样一个动作，即说出 *he* 用于表示 *John*。两个所指语用于指向同样的实体称为同指（corefer），因此例句（18.1）中 *John* 和 *he* 是同指关系。所指语的另一个术语是先行词（antecedent），它以一种方式准许使用另一个所指语，例如在提及 *John* 以后的表达中就允许用 *he* 来表示 *John*，我们称 *John* 为 *he* 的先行词。提及一个先前已经被引入话语的实体称为复指（anaphora），使用的所指语称为复指语（anaphoric）。因此例句（18.1）中的代词 *he* 和 *it* 是复指语。

自然语言给说话人提供了各种指向实体的方式。假如你的朋友有一辆Acura Integra汽车，你想提及它。依赖于实施的话语上下文（discourse context），有多种选择：*it*，*this*，*that*，*this car*，*that car*，*the car*，*the Acura*，*the Integra* 或 *my friend's car* 等。然而，无论在哪个上下文中都不可能在所有这些选项中自由选择。例如，如果听话人预先对你的朋友的汽车没有任何了解，如果该汽车从未被提及，如果该汽车并不紧邻话语的参与者，即话语的情境上下文（situational context），就不能简单地说 *it* 或 *the Acura*。

出现这种情形的原因在于所指语的每个类型都暗含关于位置的不同信息，这种位置是指说话人认为所提及的对象在听话人的看法集中所占据的位置。这些具有特殊地位的看法的子集形成了听话人对正在进行的话语的心理模型，我们称之为话语模型（discourse model）(Webber, 1978)。话语模型包括话语所指实体的表示以及它们所承担的关系。因此，为了成功地生成并解释所指语，系统需要两个部分：一是构造话语模型的方法，该模型能够随着它所表示的话语的动态变化而演化；二是各种所指语暗含的信息与听话人的看法集之间的映射方法，后者包括该话语模型。

我们将按照话语模型的两个基本操作来讲述。当话语中首次提及所指对象时，我们称它的表示被唤起（evoke）而进入模型。而其后当再次提及时，我们称从模型中访问（access）它的表示。图 18.1 给出了这两个操作以及关系的说明。

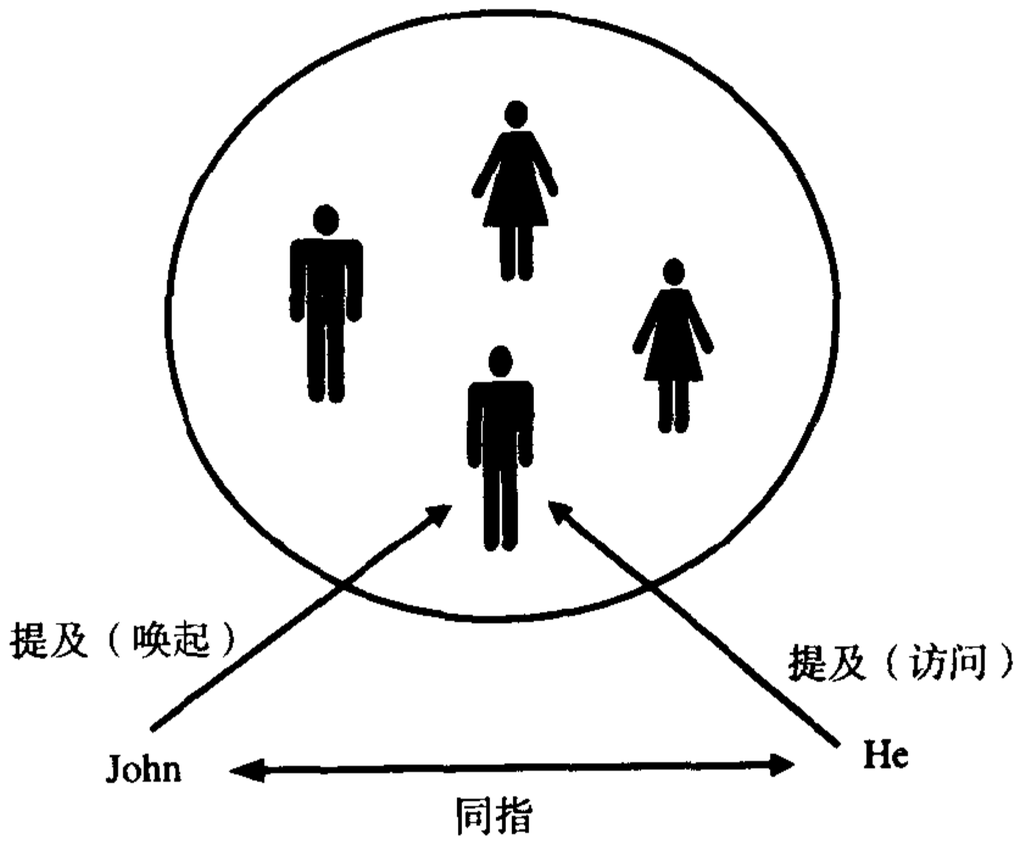


图 18.1　所指操作以及关系

我们对所指的讨论将限于实体，尽管话语也包括对许多其他类型对象的所指。下面来研究改编自 Webber(1991)的例句（18.4）。

According to John, Bob bought Sue an Integra, and Sue bought Fred a Legend. （18.4）

a. But *that* turned out to be a lie.

b. But *that* was false.

c. *That* struck me as a funny way to describe the situation.

d. *That* caused Sue to become rather poor.

e. *That* caused them both to become rather poor.

例句（18.4a）中 *that* 的所指对象是言语行为（见第 19 章），例句（18.4b）中是一个命题，例句（18.4c）中是一种描述方式，例句（18.4d）是一个事件，而例句（18.4e）是几个事件的组合。该领域仍旧期待着能够解释这些所指类型的鲁棒方法。

18.1.1　所指现象

自然语言所提供的所指现象的集合的确非常丰富。本节将给出几种基本所指现象的简短描述。首先概述 5 种所指语：不定名词短语（indefinite noun phrase）、有定名词短语（definite noun phrase）、代词（pronoun）、指示词（demonstrative）和单个复指（one-anaphora）等。然后，叙述使所指判定问题复杂化的三类所指对象：推理对象（inferrable）、不连续集（discontinuous set）和类属（generic）。

不定名词短语　不定所指将一个新的实体（对听话人来说）引入了话语环境。不定所指最常见的形式以限定词 *a*（或 *an*）为特征，如例句（18.5）所示，但是它也可以以其他量词为特征，例如例句（18.6）中的 *some* 或例句（18.7）中的限定词 *this*。

I saw *an Acura Integra* today.　（18.5）

Some Acura Integras were being unloaded at the local dealership today.　（18.6）

I saw *this awesome Acura Integra* today.　（18.7）

这类名词短语唤起了对一个新的实体的表示，该表示满足进入话语模型的约定的描述。

不定冠词 *a* 并不表示该实体对于说话人是可确认还是不可确认的。在某些情形下，这导致了特定/非特定的歧义。例句（18.5）只有特定的解释，因为说话人心里所想的是一个特定的 Integra，特指她看到的那辆汽车。而在例句（18.8）中，两种解释都有可能。

I am going to the dealership to buy an Acura Integra today.　（18.8）

更确切地说，说话人可能已经选定了 Integra（特定），也可能正计划挑选她喜欢的 Integra（非特定）。这种解释可以通过一些上下文中后续的所指语来排歧；如果该表达式是确定的，则该解释也是特定的（我希望他们仍有那款车），而如果该表达式是不确定的，则解释是非特定的（我希望他们有我喜欢的车）。然而，这个规则也有特例，例如确定表达在特殊的模型环境（我准备把它停在我的车库里）却具有非特定的解释。

有定名词短语　确定所指被用于指示听话人可以分辨的实体，这个实体可能在话语环境中已经提到过（因此已被表示于话语模型中），可能包含于听话人关于世界的看法集里，也可能该客体本身的描述就包含了它的惟一性。

在例句（18.9）中给出了所指对象可以从话语环境分辨的例子。

I saw an Acura Integra today. *The Integra* was white and needed to be washed.　（18.9）

在例句（18.10）和例句（18.11）中，我们分别给出了从听话人的看法集和从内在的惟一性来分辨实体的例子。

The Indianapolis 500 is the most popular car race in the US.　（18.10）

The fastest car in the Indianapolis 500 was an Integra.　（18.11）

有定名词短语所指需要从话语模型或听话人的看法集来访问实体。在后一种情形下，也唤起该所指对象的表示而使它进入话语模型。

代词　确定所指的另一种形式是代词化，如例句（18.12）所示。

I saw an Acura Integra today. *It* was white and needed to be washed.　（18.12）

使用代词的所指与全部使用有定名词短语的所指相比受到更强的约束，在话语模型中需要所指对象具有高度的活力或显著性。代词指示的实体被引入的位置通常（但不是总是）不超过所进行话

语向后的一两句，而有定名词短语常常可以指向更远处。这一点可以通过例句（18.13d）和例句（18.13d'）的不同来说明。

a. John went to Bob's party, and parked next to a beautiful Acura Integra. （18.13）

b. He went inside and talked to Bob for more than an hour.

c. Bob told him that he recently got engaged.

d. ?? He also said that he bought *it* yesterday.

d.' He also said that he bought *the Acura* yesterday.

讲到最后一个句子时，Integra已经不再具有足够的显著性，而这种显著性对于允许代词指向它是必要的。

代词也可能参与到提前指代（cataphora），如例句（18.14）所示，在代词所指对象出现之前就提及了代词。

Before *he* bought *it*, John checked over the Integra very carefully. （18.14）

其中，代词 *he* 和 *it* 都出现在它们的所指对象引入之前。

代词也出现在一种量化环境中，认为它们是被绑定的，如例句（18.15）所示。

Every woman bought *her* Acura at the local dealership. （18.15）

在贴切的解释中，*her* 不是指上下文中的一些女人，而是像一个变量绑定于量化表达 *every woman*。本章将不再研究这种代词的绑定解释。

指示词 指示代词（如 *this* 和 *that*）的表现，与像 *it* 这样简单的确定代词的表现有些不同。它们既可以单独出现，也可以作为限定词，例如 *this Acura*，*that Acura*。两个指示词之间的选择通常与空间邻近关系的一些概念有关：*this* 表示比较接近，而 *that* 预示间隔较远。空间距离可能是根据话语参与者的场景来判断的，如例句（18.16）所示。

[John shows Bob an Acura Integra and a Mazda Miata] （18.16）

Bob (pointing): I like *this* better than *that*.

另外，也可以按照话语模型中的概念关系对隐喻的距离加以解释。例如，研究下面的例句（18.17）。

I bought an Integra yesterday. It's similar to the one I bought five years ago. （18.17）

That one was really nice, but I like *this one* ever better.

其中，*that one* 指五年前买的 Acura（较远的时间距离），而 *this one* 指昨天买的汽车（较近的时间距离）。

单个复指 如例句（18.18）所示，单个复指混合了确定和非确定所指的特性。

I saw no less than 6 Acura Integras today. Now I want *one*. （18.18）

这里所用的 *one* 大致可以重述为短语 *one of them*，其中 *them* 指复数的所指对象，或一个通用的所指对象，如例句（18.18）所示，而 *one* 是从这个集中选择的一个成员(Webber, 1983)。因此，*one* 可以唤起一个新的实体进入话语模型，但是为了描述这个新的实体，它必须依赖于一个已有的所指对象。

这里 *one* 的用法应该从形式上区别于例句（18.19）中非特定代词的用法，以及例句（18.20）中数字 *one* 的用法。

One shouldn't pay more than twenty thousand dollars for an Acura. （18.19）

John has two Acuras, but I only have one. （18.20）

推理对象　以上描述了几种类型的所指语，现在把注意力转向几种有趣的所指对象类型，它们使所指判定问题变得复杂了。首先，我们研究所指语并不指向文中已经明显地被唤起的实体，而是指向与唤起的实体之间有推理关系的实体时的情形。这种所指对象被称为推理对象（inferrable）(Haviland and clark, 1974; Prince, 1981)。研究例句（18.21）中的 *a door* 和 *the engine* 表达方式。

I almost bought an Acura Integra today, but a *door* had a dent and *the engine* seemed noisy. （18.21）

非确定的名词短语 *a door* 通常会把一个新的门（door）引入话语环境，但在这个例子中听话人可以推理出更多的东西：它不是指任何一个门，而是指 *Integra* 的其中一个门。类似地，使用确定名词短语 *the engine* 通常假定一个引擎（engine）之前已经被唤起了，或者可以惟一地被识别。这里没有明确提及引擎，但是听话人可以推理出所指对象是前面所提及的 Integra 的引擎。

可推理对象也可以用于指定话语中话段所描述的处理的结果。我们来研究下面的菜谱(Webber and Baldwin, 1992)：

Mix the flour, butter, and water. （18.22）

a. Kneed *the dough* until smooth and shiny.

b. Spread *the paste* over the blueberries.

c. Stir *the batter* until all lumps are gone.

任何一个表达，*the dough*（固体），*the batter*（液体）和 *the paste*（介于固体和液体之间），都可以用于表示第一个句子所描述动作的结果，但是它们所预示的结果的性质都是不同的。

不连续集　在一些例子中，我们使用像 *they* 和 *them* 这样的复数所指语来指向一同唤起的实体集，例如使用复数表达（*their Acuras*）或联合的名词短语（*John and Mary*）：

John and Mary love their Acuras. *They* drive *them* all the time. （18.23）

而且，复数所指也可能指向由文中的不连续短语唤起的实体集。

John has an Acura, and Mary has a Mazda. *They* drive *them* all the time. （18.24）

这里，*they* 指 John 和 Mary，而 *them* 指 Acura 和 Mazda。注意，这个例子中的第二句子通常需要两个解释或分别的解释，即 John 驾驶 Acura，Mary 驾驶 Mazda，而不是他们都驾驶汽车。

类属　使所指问题更复杂的是存在类属所指。我们来研究例句（18.25）：

I saw no less than 6 Acura Integras today. *They* are the coolest cars. （18.25）

其中，对于 *they* 的所指，最自然的解释并不是第一个句子中提到的 6 个 Integra，而是一般来说的 Integra 类型。

18.1.2　同指的句法和语义约束

在描述了自然语言中能够找到的各种所指现象后，现在我们来研究怎样的算法才能识别出所指语的所指对象。任何一个成功的所指判定算法都需要采用的一个步骤是：基于某些相对严格的约束过滤出可能的所指对象的集合。这里给出对其中一些约束的描述。

数的一致（Number Agreement） 所指语和它们的所指对象在数上必须一致。对英语来说，这意味着对单数所指和复数所指进行区分。图 18.2 中所示的是根据数划分的代词的分类。

单数	复数	未指定
she, her, he, him, his, it	we, us, they, them	you

图 18.2 英语代词系统中数的一致

下面的例句用于说明数的一致的约束。

John has a new Acura. It is red. （18.26）

John has three new Acuras. They are red. （18.27）

*John has a new Acura. They are red. （18.28）

*John has three new Acuras. It is red. （18.29）

人称和格的一致（Person and Case Agreement） 英语有三种不同的人称：第一、第二和第三人称。图 18.3 给出的是根据人称划分的代词的分类。

	第一人称	第二人称	第三人称
主格	I, we	you	he, she, they
宾格	me, us	you	him, her, them
属格	my, our	your	his, her, their

图 18.3 英语代词系统中人称和格的一致

下面的例句用于说明人称和格的一致的约束。

You and I have Acuras. We love them. （18.30）

John and Mary have Acuras. They love them. （18.31）

*John and Mary have Acuras. We love them.（其中 *We*=John and Mary） （18.32）

*You and I have Acuras. They love them.（其中 *They*=You and I） （18.33）

另外，英语中的代词也受到格一致的约束，在主语位置（主格，例如 *he*，*she* 和 *they*）、宾语位置（宾格，例如 *him*，*her* 和 *them*）和所属格位置（属格，例如 *his Acura*，*her Acura* 和 *their Acura*）需要代词的不同形式。图 18.3 也给出了这种分类。

性的一致（Gender Agreement） 所指对象也必须满足所指语指定的性。英语中第三人称代词被区分为阳性、阴性和非人类。与一些语言不同，前两者只能用于有生命的实体。在图 18.4 中给出了一些例子。

阳性	阴性	非人类
he, him, his	she, her	it

图 18.4 英语代词系统中人称和格的一致

下面的例句用于说明性的一致的约束。

John has an Acura. He is attractive.(he=John, not the Acura) （18.34）

John has an Acura. It is attractive.(it=the Acura, not John) （18.35）

句法约束（syntactic constraint）　当一个所指语和一个可能的先行名词短语出现在同一个句子中时，所指关系也可能受到该所指语和先行名词短语之间句法关系的约束。例如，下面所有句子中的代词都服从括号中的约束。

John bought himself a new Acura. [himself=John]　　(18.36)

John bought him a new Acura. [him ≠ John]　　(18.37)

John said that Bill bought him a new Acura. [him ≠ Bill]　　(18.38)

John said that Bill bought himself a new Acura. [himself=Bill]　　(18.39)

He said that he bought John a new Acura. [He ≠ John; he ≠ John]　　(18.40)

英语中的代词，例如 *himself*，*herself* 和 *themselves* 被称为反身代词（reflexive）。大幅度地简化这种情形，可以说：反身代词可用于同指包含它的最紧邻从句的主语，见例句（18.36），而非反身代词不能用于同指该主语，见例句（18.37）。这个规则只能应用于例句（18.38）和例句（18.39）所示的最紧邻从句的主语，相反的所指模式出现在代词与较高一级句子的主语之间。另外，像 *John* 这样的完全的名词短语并不能同指最紧邻从句的主语，也不能同指较高一级句子的主语，见例句（18.40）。

尽管这些句法约束应用于所指语和特别的可能先行名词短语，然而这些约束实际上禁止了表示相同实体的任何其他先行词的任意两个词之间同指一个实体。例如，通常像 *him* 这样的非反身代词能够与前一个句子中的主语同指一个实体，如例句（18.41）所示，但是它在例句（18.42）中不能，因为它与第二个从句中同指代词 *he* 的句法关系。

John wanted a new car. Bill bought him a new Acura. [him=John]　　(18.41)

John wanted a new car. He bought him a new Acura. [he=John; him ≠ John]　　(18.42)

从许多方面看，这些规则都过分地简化了这种情形，并且不能覆盖许多例子。的确，对事实的进一步审视使这一问题变得更复杂了。实际上，只使用句法关系不太容易解释所有事实(Kuno, 1987)。例如，在例句（18.43）和例句（18.44）中，反身代词 *himself* 和非反身代词 *him* 都可以指向主语 *John*，即使它们出现于相同的句法结构中。

John set the pamphlets about Acuras next to himself. [himself=John]　　(18.43)

John set the pamphlets about Acuras next to him. [him=John]　　(18.44)

然而，在本章以后的算法讨论中，我们将假设句子内的同指的约束是由于句法的原因。

选择限制（selectional restriction）　动词对它的论元所施加的选择限制（见第 16 章）可以用于排除所指对象，如例句（18.45）所示。

John parked his Acura in the garage. He had driven it around for hours.　　(18.45)

其中，*it* 有两个可能的所指对象，Acura 和 garage。然而，动词 *drive* 要求它的直接宾语是某种能够驾驶的事物，例如小轿车（car）、卡车（truck）或公共汽车（bus），但是不能是车库（garage）。因此，代词作为 *drive* 的宾语这一事实限制了 Acura 的可能的所指对象集合。可以想到，一个实际的 NLP 系统应该包含关于系统词典中动词的选择限制的相当全面的集合。

选择限制在带有比喻的例子中可能被打破（见第 16 章）。例如，我们来研究下面的例句（18.46）。

John bought a new Acura. It drinks gasoline like you would not believe.　　(18.46)

尽管动词*drink*通常不能带有一个非生命的主语，但是这里它的比喻用法允许*it*指向*a new Acura*。

当然，也可以使用更概括的语义约束，但是以一种全面的方式将其引入是非常困难的。我们来研究例句（18.47）。

John parked his Acura in the garage. It is incredibly messy, with old bike and car parts lying around everywhere. （18.47）

这里几乎可以肯定*it*的所指对象是车库（garage），部分原因在于汽车可能太小而不能使自行车和各种汽车配件摆得到处都是。判定这个所指需要系统具有关于典型的汽车有多大，典型的车库有多大，以及其中有哪些典型的东西等的知识。而另一方面，一个人关于Beverly Hills的知识可能导致他认为例句（18.48）中*it*的所指对象确实是Acura。

John parked his Acura in downtown Beverly Hills. It is incredibly messy, with old bike and car parts lying around everywhere. （18.48）

最后，话语参与者共享的几乎所有知识对于判断代词所指都是必需的。但是，部分原因在于这类知识的浩瀚，实践算法通常并不完全依赖于这些知识。

18.1.3 代词解释中的优先关系

在前面的章节中，我们讨论了可以被算法用于确定所指语的可能所指对象的相对严格的约束。现在来讨论一些可用于算法实现的更容易被违反的优先关系。特别是这些优先关系已经被应用于代词解释。因为所指判断算法的大多数工作都重点关注代词解释，所以这里也将集中讨论这个问题。

新近性（recency） 大多数所指理论都引入了这样一个概念：新近的话段所引入的实体比那些先前较远的话段引入的实体具有较高的显著性。因此，在例句（18.49）中，代词*it*的所指对象相比Integra更可能是Legend。

John has an Integra. Bill has a Legend. Mary likes to drive it. （18.49）

语法角色（grammatical role） 许多理论都规定了实体的显著性的层级，即通过表示这些实体的表达的语法位置来进行排序。典型地，我们认为处于主语位置的实体的显著性高于处于宾语位置的实体，而处于宾语位置的实体的显著性又比后续位置的实体高。

例句（18.50）和例句（18.51）中就借用了这样的层级支持。尽管在每个例子中第一个句子的命题内容大致相同，但代词*him*的优先所指对象在每个例子中都因主语的不同而不同：在例句（18.50）中是John，而在例句（18.51）中是Bill。在例句（18.52）中，John和Bill的所指一起出现于主语位置。因为看上去它们具有相同的显著度，所以代词的所指就不清楚了。

John went to the Acura dealership with Bill. He bought an Integra. [he = John] （18.50）

Bill went to the Acura dealership with John. He bought an Integra. [he = Bill] （18.51）

John and Bill went to the Acura dealership. He bought an Integra. [he = ??] （18.52）

重复提及（repeated mention） 一些理论引入了这样的思想：在前面话语中已经作为焦点的实体，在其后的话语中更可能作为焦点，所以它们的所指也更可能被代词化。例如，在例句（18.51）中，Bill是优先的解释，而例句（18.53）的最后一个句子中的代词的所指更可能是John。

John needed a car to get to his new job. He decided that he wanted something sporty. Bill went to the Acura dealership with him. He bought an Integra. [he = John]　（18.53）

平行（parallelism）　平行效果会带来明显的优先关系，如例句（18.54）所示。

Mary went with Sue to the Acura dealership. Sally went with her to the Mazda dealership. [her = Sue]　（18.54）

根据前面所述的语法角色的层级思想，Mary 比 Sue 具有更高的显著性，因此应该作为 her 的优先所指对象。同时，也没有任何语义上的原因使得 Mary 不能作为所指对象。然而，*her* 实际上被理解为 Sue。

这说明我们可能需要一个启发式的规则，以说明非主语代词更喜欢非主语的所指对象。然而，这样一个启发式规则对于缺少例句（18.54）那样的结构平行的例子并不适用，例如例句（18.55）。在这个例子中代词的优先所指对象是 Mary，而不是 Sue。

Mary went with Sue to the Acura dealership. Sally told her not to buy anything. [her =Mary]（18.55）

动词语义（verb semantics）　有些动词的出现会对它们的其中一个论元的位置产生语义上的强调，这会造成对其后代词的理解偏差。下面比较例句（18.56）和例句（18.57）。

John telephoned Bill. He lost the pamphlet on Acuras.　（18.56）

John criticized Bill. He lost the pamphlet on Acuras.　（18.57）

这两个例子的不同仅在于第一个句子中所用的动词，而通常例句（18.56）的主语代词被判定为 John，而例句（18.57）的主语代词被判定为 Bill。有些研究者认为这种效果来自动词的“隐含因果关系”：criticizing 事件的这种隐含因果被认为是动词的宾语，而 telephoning 事件的这种隐含因果关系被认为是动词的主语。这使得在这个论元位置的实体具有较高的显著性，从而导致例句（18.56）和例句（18.57）具有不同的优先关系。

类似的优先关系可以根据可能的先行词所占的题元角色（见第 16 章）进行阐明。例如，大部分听话人都会判定例句（18.58）中的 *He* 为 John，而例句（18.59）中的 *He* 为 Bill。尽管这些所指对象是从不同的语法角色位置被唤起的，它们都满足相应的动词的目标（Goal）题元角色，而其他可能的所指对象满足源（Source）角色。同样地，如果激励（stimulus）角色的填充者比体验角色的填充者更优先，听话人通常会将例句（18.60）和例句（18.61）中的 *He* 分别判定为 John 和 the car dealer。

John seized the Acura pamphlet from Bill. He loves reading about cars.　（18.58）

(Goal=John, Source=Bill)

John passed the Acura pamphlet to Bill. He loves reading about cars.　（18.59）

(Goal= Bill, Source= John)

The car dealer admired John. He knows Acuras inside and out.　（18.60）

(Stimulus=John, Experiencer =the car dealer)

The car dealer impressed John. He knows Acuras inside and out.　（18.61）

(Stimulus= the car dealer, Experiencer = John)

18.1.4　代词判定算法

目前所提出的代词判定算法仍不能很好地解释所有这些优先关系，更不必谈成功地解决它们之间出现的冲突。但是，Lappin and Leass(1994)给出了考虑到这些优先关系的代词解释的直接算法。

该算法采用一个简单的加权方案，综合考虑了新近和基于句法的优先关系的影响；除了那些一致关系所施加的优先关系以外，并没有采用其他语义优先关系。这里描述的是稍加简化的应用于第三人称非反身代词的算法。

一般来说，该算法所执行的运算有两类：话语模型的更新和代词判定。首先，当遇到一个唤起新的实体的名词短语时，必须为它添加一个表示以及用于计算的显著度（我们称之为显著值）。显著值（salience value）是由一组显著因子（salience factor）所指派的权重的总和来计算的。图 18.5 给出的是该系统所采用的显著因子以及它们相应的权重。

Sentence recency	100
Subject emphasis	80
Existential emphasis	70
Accusative (direct object) emphasis	50
Indirect object and oblique complement emphasis	40
Non-adverbial emphasis	50
Head noun emphasis	80

图 18.5 Lappin 和 Leass 系统的显著因子

话语模型中的每个因子为实体所指派的权重每处理一个新句子就减一半。再加上句子新近权重所添加的影响（最初的权重是 100，每处理一个新句子就减一半），就可以捕捉到例句（18.49）所描述的新近优先关系，因为当前句子所提及的所指语比前面的句子倾向于具有较高的权重，而依次前面的句子又比那些更前面的句子具有较高的权重。

类似地，图18.5中的下面5个因子可以看成是利用下述的层级来表示语法角色优先关系方案的一种方式：

主语（subject）> 存在谓词性名词（existential predicate nominal）> 宾语（object）> 间接宾语或旁格（indirect object or oblique）> 可区分的状语介词短语（demarcated adverbial PP）

对这 5 种位置，我们通过例句（18.62）至例句（18.66）中的斜体短语的位置分别加以说明。

An Acura Integra is parked in the lot.(subject) （18.62）

There is *an Acura Integra* parked in the lot.(existential predicate nominal) （18.63）

John parked *an Acura Integra* in the lot.(object) （18.64）

John gave *his Acura Integra* a bath.(indirect object) （18.65）

Inside *his Acura Integra*, John showed Susan his new CD player.(demarcated adverbial PP) （18.66）

在可区分的状语介词短语中，即那些被类似例句（18.66）中的逗号等标点符号分开的短语中，所有其他位置的所指对象的优先关系都被表示为正值 50，在图 18.5 中以非状语强调（non-adverbial emphasis）列出。这是为了确保任何所指对象的权重总是正值，这是必须的，因为权重的减半效果总是使之减少。

中心语名词强调因子惩罚那些嵌在较大名词短语中的所指对象，同时也提升那些没有嵌在较大名词短语中的所指对象的权重。因此，从例句（18.62）到例句（18.66）中的短语 Acura Integra，都会因为是中心语名词而获得 80 点的权重，而例句（18.67）中的 Acura Integra 将不能获得，因为它被嵌在主语名词短语中。

The owner's manual for *an Acura Integra* is on John's desk. （18.67）

这些因子对一个所指对象的显著性的贡献都基于表示该所指对象的名词短语的属性。当然，很可能在前述的话语中几个名词短语是同一个所指对象，它们都被指派了不同层级的显著性，因此需要一种方法将其贡献融合在一起。Lappin 和 Leass 为每个所指对象添加一个等价类（equivalence class）以解决该问题，等价类包含所有已经确定指向该所指对象的名词短语。显著因子指派给所指对象的权重是它指派给该所指对象的等价类的所有成员的权重。因此，一个所指对象的显著性权重的计算是将每个因子相加。显著因子的范围是一个句子，因此，如果一个可能的所指对象既在当前句子中提及，也在前述句子中提及，句子的新近权重将分别计算（另外如果相同的所指对象在同一个句子中出现多次，该权重将只计算一次）。因此，一个所指对象在前述话语中多次提及将增加它的显著性，这是对在代词解释优先关系中提到的“重复提及”的具体化。

一旦我们打算用可能的新所指对象更新话语模型并重新计算它们的显著值，就不得不考虑位于新句子中所有代词的判定处理。我们采用两个显著性权重来处理该问题，一个用于奖励代词和可能的所指对象之间的语法角色平行，另一个用于惩罚提前指代（cataphoric reference）。在图 18.6 中给出了权重。与其他的优先关系不同，代词的这两个权重不能独立计算，因此也不能在话语模型的更新阶段进行计算。我们将用术语初始显著值（initial salience value）表示给定所指对象在没有应用这些因子时的权重，而用术语最终显著值（final salience value）表示应用这些因子后的权重。

角色平行	35
提前指代	-175

图 18.6　Lappin 和 Leass 系统中每个代词的显著权重

现在我们准备详细说明代词的判定算法。假定话语模型已经被更新，以反映出上述所指对象的初始显著值，则判定一个代词的步骤如下：

1. 收集可能的所指对象（可在前述四个句子中）。
2. 排除与代词在数和性上不一致的可能的所指对象。
3. 排除不能通过句内句法同指约束（见 18.1.2 节）的可能的所指对象。
4. 通过将在话语模型更新阶段计算出的显著值（即对图 18.5 中的所有可应用值求和）与图 18.6 中可应用的值相加，来计算所指对象的总的显著值。
5. 选择显著值最高的所指对象。在出现平分的情况下，根据字符串的位置（计算时不考虑方向），选择最靠近的所指对象。

我们通过例句（18.68）来演示该算法实施的每个步骤。

John saw a beautiful Acura Integra at the dealership. He showed it to Bob. He bought it.　（18.68）

首先处理第一个句子，收集所有可能的所指对象，并计算它们的初始显著值。下表给出的是每个显著因子对显著性的贡献。

	Rec	Subj	Exist	Obj	Ind-Obj	Non-Adv	Head N	Total
John	100	80				50	80	310
Integra	100			50		50	80	280
dealership	100					50	80	230

在这个句子中没有代词需要判定。我们继续处理下面的句子，对上表中的值通过除以因子2加以降低。短语栏给出了每个所指对象的所指语的等价类。

所指对象	短语	值
John	{ *John* }	155
Integra	{ *a beautiful Acura Integra* }	140
dealership	{ *the dealership* }	115

第二个句子中的首个名词短语是代词*he*。因为*he*所指的是男性，通过步骤2中的判定算法就可以将可能的所指对象集减少为只包括John，因此我们就可以停止，并选择John为*he*的所指对象。

现在话语模型必须加以更新。首先，代词*he*被加入John的等价类（为与其他可能提及的*he*区分，我们用he_1来表示它）。因为*he*出现在当前的句子中，而John出现在前面的句子中，因此它们两个的显著因子并不重叠。代词在当前句子中（recency=100），主语位置（=80），非状语（=50），非嵌入（=80），则总值310加入当前John的权重：

所指对象	短语	值
John	{ *John*, he_1 }	465
Integra	{ *a beautiful Acura Integra* }	140
dealership	{ *the dealership* }	115

第二个句子中的下一个名词短语是代词*it*，它可能是Integra或dealership。首先，我们通过在上面的初始显著值中加入图18.6中可应用的权重来计算最终显著值。两个所指对象都不会引起提前指代，所以并不使用该因子。从平行优先关系来看，*it*和*a beautiful Acura Integra*在各自的句子中都处于宾语位置（而the dealership不是），因此对这种选择加权35。则Integra的权重是175，而dealership的权重是115，我们选择Integra为所指对象。

现在话语模型必须再次被更新。因为*it*处于非嵌入宾语位置，它获得的权重为100+50+50+80=280，并加入Integra目前的权重。

所指对象	短语	值
John	{ *John*, he_1 }	465
Integra	{ *a beautiful Acura Integra*, it_1 }	420
dealership	{ *the dealership* }	115

第二个句子中的最后一个名词短语是*Bob*，它引入一个新的话语所指对象。因为它占据间接论元的位置，获得的权重为100+40+50+80=270。

所指对象	短语	值
John	{ *John*, he_1 }	465
Integra	{ *a beautiful Acura Integra*, it_1 }	420
Bob	{ *Bob* }	270
dealership	{ *the dealership* }	115

现在我们准备处理最后一个句子。我们再次通过减半来降低当前的权重。

所指对象	短语	值
John	{ *John*, he_1 }	232.5
Integra	{ *a beautiful Acura Integra*, it_1 }	210
Bob	{ *Bob* }	135
dealership	{ *the dealership* }	57.5

读者可以验证 *he* 的所指对象是 John，而 *it* 的所指对象是 Integra。

Lappin 和 Leass 所采用的权重是通过对关于计算机训练手册的语料所做的实验获得的。当该算法与这里没有讲述的几个过滤算法一起使用，并应用于相同体裁的未训练语料时，可以获得 86% 的精度。对于其他体裁的语料，这些具体的权重可能并不是最优的（对其他语言来说更是如此），因此对新的应用或语言，读者可以借助训练数据通过实验的方法来确定。

下面将简单介绍其他两种代词判定的方法。

树查询算法（Tree Search Algorithm）　Hobbs(1978)描述了一种代词判定的算法，以当前句子以前的几个句子（包含当前句子）的句法表示为输入，并在这些句法树中执行先行名词短语的查询。这里并没有明确地用到 Lappin 和 Leass 算法中的话语模型或优先关系的表示。但是，通过执行句法树查询的先后顺序可以近似地表现出某些优先关系。

查询剖析树的算法也必须指定语法，因为与句法树结构有关的假设将影响结果。在图 18.7 中给出了该算法所用的英语语法的片断。该算法的步骤如下：

1. 从紧邻的支配该代词的名词短语（*NP*）结点开始。
2. 沿剖析树向上到达所遇到的第一个 *NP* 或句子（*S*）结点。称该结点为 *X*，并称到达该结点的路径为 *p*。
3. 以从左到右、宽度优先的方式遍历路径 *p* 左侧低于结点 *X* 的所有分支。对于遇到的任何 *NP* 结点，如果在它与 *X* 之间存在 *NP* 或 *S* 结点，则提议作为先行词。
4. 如果结点 *X* 是句子中最高的 *S* 结点，则按照新近顺序（首先是最新近的），遍历文中前述句子的表层剖析树；每个剖析树用从左到右、宽度优先的方式遍历，当遇到一个 *NP* 结点时，它就被提议为先行词。如果 *X* 不是句子中最高的 *S* 结点，继续步骤 5。
5. 从结点 *X* 沿剖析树向上到达最先遇到的 *NP* 或 *S* 结点，称它为新的 *X* 结点，并称到达该结点的路径为 *p*。
6. 如果 *X* 是 *NP* 结点，并且如果到 X 的路径 *p* 没有穿过紧邻的支配 *X* 的名词性结点，则提议 *X* 为先行词。
7. 以从左到右、宽度优先的方式遍历路径 *p* 左侧低于结点 *X* 的所有分支。提议所遇到的任何 *NP* 结点为先行词。
8. 如果 *X* 是 *S* 结点，以从左到右、宽度优先的方式遍历路径 *p* 右侧低于结点 *X* 的所有分支，但是不要遍历低于任何遇到的 *NP* 或 *S* 结点的分支。提议所遇到的任何 *NP* 结点为先行词。
9. 回到步骤 4。

$$
\begin{aligned}
&S \rightarrow NP\ VP\\
&NP \rightarrow \left\{ \begin{array}{l} (Det)\quad Nominal\quad \left(\left\{ \begin{array}{l} PP \\ Rel \end{array} \right\} \right)^{*} \\ pronoun \end{array} \right\}\\
&Det \rightarrow \left\{ \begin{array}{l} determiner \\ NP\ 's \end{array} \right\}\\
&PP \rightarrow preposition\ NP\\
&Nominal \rightarrow noun\ (PP)^{*}\\
&Rel \rightarrow wh\text{-}word\ S\\
&VP \rightarrow verb\ NP\ (PP)^{*}
\end{aligned}
$$

图 18.7　树查询算法的语法片断

如前所述，该算法以完整并正确的句法结构为输入。Hobbs从3个不同的文本中各选出一百个例句手工评测了他的方法（分为剖析构建和算法实现两部分），报道的精度为88.3%（如果假定某些选择限制约束，则精度上升为91.7%）。Lappin和Leass在他们的系统中也实现了该算法，对他们的测试语料所报道的精度为82%。尽管这低于他们自己的算法精度（86%），但应该记住Lappin和Leass所用的测试语料的体裁与他们的训练集一致，但与Hobbs在研制算法时所用的体裁不同。

中心算法（Centering Algorithm） 正如前面的描述，Hobbs算法并没有明确采用话语模型表示。Lappin和Leass算法中的确采用了话语模型，但是这种算法将显著性作为优先关系的加权组合。中心理论(Grosz et al., 1995)（GJW）也明确采用了话语模型的表示，并引入了一个额外的主张：在话语中的任何给定点都有一个单独的实体被作为“中心”，该实体与被唤起的其他实体都有所不同。

在该话语模型中主要记录了两种表示。在以下的讲述中，以U_n和U_{n+1}表示相邻的话段。话段U_n的向后看中心（backward looking center），以$C_b(U_n)$表示，代表在U_n被解释后，话语中当前所关注的实体。话段U_n的向前看中心（forward looking center），以$C_f(U_n)$表示，形成一个包含U_n中提及的实体的有序列表，列表中的所有实体都可以作为后面话段的C_b。实际上，$C_b(U_{n+1})$被定义为U_{n+1}提及的列表$C_f(U_n)$中级别最高的元素。话语中的首个话段的C_b是未定义的。至于$C_f(U_n)$中实体排序的方式，出于简化的考虑，我们可以采用在Lappin和Leass算法中权重（的子集）所表示的语法角色层级，重述如下[①]。

subject > existential predicate nominal > object >
indirect object or oblique > demarcated adverbial PP

然而，与Lappin和Leass算法不同，对列表中的实体并没有附加一个权重值，仅仅是简单的相互之间的排序。作为简写，我们将最高级别的向前看中心称为C_p，即最优先的中心（preferred center）。

我们描述一个归功于Brennan et al.(1987)的基于中心的算法（今后称为BFP）。其他中心算法参见Walker et al.(1994); Kameyama(1986)和Strube and Hahn(1996)。在该算法中，代词的优先所指对象是通过相邻句子向前看中心和向后看中心之间的关系来计算的。话段对U_n和U_{n+1}之间的句间关系是通过$C_b(U_{n+1})$，$C_b(U_n)$和$C_p(U_{n+1})$之间的关系来定义的，如图18.8所示。

	$C_b(U_{n+1}) = C_b(U_n)$ or undefined $C_b(U_n)$	$C_b(U_{n+1}) \neq C_b(U_n)$
$C_b(U_{n+1}) = C_p(U_{n+1})$	Continue	Smooth-Shift
$C_b(U_{n+1}) \neq C_p(U_{n+1})$	Retain	Rough-Shift

图18.8 BFP算法中的转换

算法所用的规则如下：

- 规则1：如果$C_f(U_n)$中的所有元素都是由话段U_{n+1}中的代词构成的，则$C_b(U_{n+1})$也必须是一个代词。
- 规则2：转换状态是有优先顺序的。继续（Continue）优先于保持（Retain），保持优先于平滑转移（Smooth-Shift），平滑转移优先于粗糙转移（Rough-Shift）。

① 这是Brennan et al.(1987)所用层级的扩展形式，下面给出描述。

在定义了这些概念和规则以后，下面是算法的定义：

1. 为每个所指对象指派的可能集生成可能的 C_b-C_f 组合。
2. 通过约束，例如句法同指约束、选择限制、中心规则和约束等来过滤。
3. 通过转换的顺序给出排序。

如果没有违反规则 1 和其他的同指约束（性、数、句法、选择约束），那么获得指派的代词的所指对象是那些在规则 2 中产生最优先关系的所指对象。

让我们以例句（18.68）为例，重新编号为下面的例句（18.69），以说明算法的处理过程。

John saw a beautiful Acura Integra at the dealership.(U_1)　　（18.69）
He showed it to Bob.(U_2)
He bought it.(U_3)

利用语法角色的层级对 C_f 进行排序，对句子 U_1 可以得到：

$C_f(U_1)$: {John, Integra, dealership}
$C_p(U_1)$: John
$C_b(U_1)$: undefined

句子 U_2 包含两个代词：*he* 和 *it*，其中 *he* 与 John 一致，*it* 与 Integra 或 dealership 一致。这里 $C_b(U_2)$ 指 John，因为是在 U_2 中提及的 $C_f(U_1)$ 的顺序最高的成员（是 *he* 惟一可能的所指对象）。我们比较 *it* 的每个可能所指对象的转换结果。如果假设 *it* 所指的是 Integra，则指派将是：

$C_f(U_2)$: {John, Integra, Bob}
$C_p(U_2)$: John
$C_b(U_2)$: John
Result: Continue　($C_p(U_2)$=$C_b(U_2)$; $C_b(U_1)$ undefined)

如果假设 *it* 所指的是 dealership，则指派将是：

$C_f(U_2)$: {John, dealership, Bob}
$C_p(U_2)$: John
$C_b(U_2)$: John
Result: Continue　($C_p(U_2)$=$C_b(U_2)$; $C_b(U_1)$ undefined)

因为这两种可能性的结果都是继续（Continue）状态，算法不能确定应该接受哪种可能性。为了能够继续说明该算法，我们将假设根据在前面 C_f 中的顺序来打破这个平局。因此，我们将选择 Integra 而不是 dealership 作为 *it* 的所指对象，选择上面所列的第一种可能性为当前话语模型的表示。

在第三个句子 U_3 中，*he* 既与 John 又与 Bob 一致，而 *it* 与 Integra 一致。如果假设 *he* 指向 John，则 John 是 $C_b(U_3)$，并且指派将是：

$C_f(U_3)$: {John, Acura}
$C_p(U_3)$: John
$C_b(U_3)$: John
Result: Continue　($C_p(U_3)$=$C_b(U_3)$=$C_b(U_2)$)

如果假设 *he* 指向 Bob，则 Bob 是 $C_b(U_3)$，并且指派将是：

$C_f(U_3)$: {Bob, Acura}
$C_p(U_3)$: Bob
$C_b(U_3)$: Bob
Result: Smooth-Shift $(C_p(U_3)=C_b(U_3); C_b(U_3)\neq C_b(U_2))$

根据规则2，继续（Continue）优先于平滑转移（Smooth-Shift），则John被正确地选择为所指对象。

中心算法引入的主要显著因子包括：语法角色、新近性和重复提及等优先关系。但是与Lappin和Leass算法不同，语法层级对显著性影响的方式是间接的，因为确定最终所指对象指派的是作为结果的转换类型。特别是，如果低级语法角色的所指对象导致的转换是较高级别的，它将比高级角色的所指对象优先。因此，中心算法可能（常常不正确地，但不是总是）将其他算法认为是相对较低显著性的所指对象判定一个代词的所指对象(Lappin and Leass, 1994; Kehler, 1997a)。例如，在例句（18.70）中，

Bob opened up a new dealership last week. John took a look at the Acuras in his lot. He ended up buying one. （18.70）

中心算法将Bob指派为第三个句子中主语代词 *he* 的所指对象，因为Bob是$C_b(U_2)$，该指派导致的是继续关系，而指派John导致的是平滑转移关系。Hobbs和Lappin/Leass算法会把John指派为所指对象。

与Hobbs算法一样，中心算法也是基于这样的假设：以正确的句法结构作为输入。为了对自然产生的数据进行自动评测，中心算法将不得不引入详细说明，既包括句中所有名词短语在C_f列表中相互排序的方法（当前的方法仅包括某些语法角色的非嵌入填充者，只能生成部分顺序），也包括句中所有名词短语判定的方法，例如，回忆例句（18.68）的第二个句子中判定 *it* 时的不确定性。

而Walker(1989)对分布于三种体裁的281个例子的语料进行了中心算法的手工评测，并与Hobbs算法的性能进行了对比。评测假定以足够的句法表示、语法角色标注和选择约束信息为输入。而且，对那些中心算法不能惟一指定所指对象的例句，只将Hobbs算法能够正确识别的例句计入中心算法的错误。在这种附带条件下，Walker报道中心算法的精度为77.6%，而Hobbs算法为81.8%。几种中心算法和Hobbs算法的比较可以参阅Tetreault(1999)。

18.2 文本的连贯

前一节的大部分内容都是在讲述复指的性质和话语中代词的判定方法。复指语常常被称为衔接语（cohesive device）(Halliday and Hasan, 1976)，因为它们所建立的同指关系所起的作用是将话语的不同部分“结合”在一起，使话语连贯。尽管话语中常常包含衔接语，但是仅仅存在这类衔接语并不能满足话语连贯的要求。本节将讲述文本连贯的涵义和确定文本连贯的计算机制。

18.2.1 现象

假如随意收集一些良构并可独立理解的话段，例如从本书前面的每一章选择一个句子，那么你获得的是一个话语吗？几乎可以肯定这是不可能的。原因是当把这些话段并列在一起时并不能体现出连贯（coherence）。例如，研究下面例句（18.71）和例句（18.72）之间的不同。

John hid Bill's car keys. He was drunk. （18.71）
（John藏起了Bill的车钥匙。他喝醉了）

?? John hid Bill's car keys. He likes spinach.　　(18.72)

(John 藏起了 Bill 的车钥匙。他喜欢菠菜。)

大部分人都会发现例句(18.71)很平常，而例句(18.72)就有些奇怪。为什么呢？与例句(18.71)一样，例句(18.72)也是良构和容易理解的。将这些句子并列在一起似乎出现了一些错误。听话人也许会问，例如，藏起某人的车钥匙与喜欢菠菜有什么关系。通过这个问题，听话人是在质疑该段落的连贯性。

另外，听话人也可能构建一种解释使得该话语连贯，例如，推测也许有人给 John 菠菜，以交换 Bill 被藏起的车钥匙。事实上，如果我们在一个已知的上下文中考虑，就会发现该段落现在变得好理解了。为什么会如此呢？因为通过这个推测允许听话人将John喜欢菠菜作为他藏起了Bill车钥匙的原因，这就可以解释这两个句子为什么被连接。听话人尽可能识别出这种连接的事实表明：我们需要将确立连贯作为话语理解的一部分。

话语的话段之间的所有可能连接可以被称为连贯关系(coherence relation)集。下面给出的是 Hobbs(1979a)提出的一些连贯关系。符号 S_0 和 S_1 分别表示两个相关句子的意义。

结果(result)　推测 S_0 所声明的状态或事件导致或可能导致 S_1 所声明的状态或事件。

John bought an Acura. His father went ballistic.　　(18.73)

说明(explanation)　推测 S_1 所声明的状态或事件导致或可能导致 S_0 所声明的状态或事件。

John hid Bill's car keys. He was drunk.　　(18.74)

平行(parallel)　推测 S_0 所声明的 $p(a_1, a_2, ...)$ 和 S_1 所声明的 $p(b_1, b_2, ...)$，对所有 i，a_i 和 b_i 是类似的。

John bought an Acura. Bill leased a BMW.　　(18.75)

详述(Elaboration)　推测 S_0 和 S_1 所声明的是同一命题 P。

John bought an Acura this weekend. He purchased a beautiful new Integra for 20 thousand dollars at Bill's dealership on Saturday afternoon.　　(18.76)

时机(Occasion)　推测从 S_0 所声明的状态到 S_1 所声明的最终状态的状态变化，或推测从 S_1 所声明的状态到 S_0 所声明的最初状态的状态变化。

John bought an Acura. He drove to the ballgame.　　(18.77)

识别连贯的机制能够支持许多自然语言的应用，包括信息抽取和信息摘要。例如，依靠“详述”关系而连贯的话语的特征常常是一个概要句，紧接着一个或多个句子来增加细节，如例句(18.76)。在该段落中尽管有两个句子描述事件，但是从详述关系可以推测出它们描述的是同一事件。识别这个事实的机制能告诉信息抽取或摘要系统将句中的信息融合生成一个事件(而不是两个事件)的描述。

18.2.2　基于推理的判定算法

以上所述的每个连贯关系都是与一个或多个约束有关的，符合这些约束才能维持这种关系。那么，怎样才能应用这些约束呢？我们需要一个进行推理的方法。也许最熟悉的推理类型是演绎(deduction)；回想 14.3 节讲述过的，演绎的中心规则是取式推理(modus ponens)：

$$\frac{\alpha \Rightarrow \beta \quad \alpha}{\beta}$$

下面是取式推理的一个例子：

$$\frac{\text{All Acuras are fast.} \quad \text{John's car is an Acura.}}{\text{John's car is fast.}}$$

演绎是可靠推理（sound inference）的一种形式：如果前提为真，结论必为真。

然而，许多语言理解所依赖的推理是不可靠的。尽管不可靠推理具有推出大量推论的能力，但是它也导致一些有误的解释和错误的理解。这类推理的一种方法被称为溯因推理（abduction）(Pierce, 1995)。溯因推理的中心规则是:

$$\frac{\alpha \Rightarrow \beta \quad \beta}{\alpha}$$

尽管演绎是向前推出隐含的关系，而溯因推理是向后，从结果中找可能的原因。下面是溯因推理的一个例子：

$$\frac{\text{All Acuras are fast.} \quad \text{John's car is fast.}}{\text{John's car is an Acura.}}$$

显然，这可能是一个不正确的推理：John 的汽车可能是由其他制造商生产的，同时仍可能是速度快的。

一般来说，一个给定的结果β可能有许多潜在的原因α_i。我们从一个事实所要的并不仅仅是对它的一个可能的解释，通常需要最佳解释。为了实现这个目的，要比较可选择的溯因推理的品质。这里可以采用各种战术。一种可能是采用概率模型(Charniak and Goldman, 1988; Charniak and Shimony, 1990)，尽管在选择计算概率的正确空间和假如缺少事件语料库时获取这些概率的方法等方面会出现一些问题。另一种方法是利用纯粹的启发式战术(Charniak and McDermott, 1985，第10章），例如优先选择假设数目最少的解释，或选择采用最具体输入特征的解释。尽管这类启发式战术最容易实现，但通常被证实过于脆弱和有限。最后，也可以采用更全面的基于代价（cost-based）的战术，它结合了概率特征（既包括正值也包括负值）和启发式方法。在此描述的演绎解释方法归功于 Hobbs et al.(1993)，采用的就是这样的战术。然而，为了简化讨论，我们将几乎完全忽略系统中的代价部分，但记住它是必须的。

Hobbs et al.(1993)将他们的方法用于语言理解中的许多问题；这里将集中讨论它在确立话语连贯中的应用，世界知识和领域知识用于确定话段之间最合理的连贯关系。让我们一步步通过分析来确立例句（18.71）的连贯。首先，需要关于连贯关系本身的公理。公理（18.78）表明一个可能的连贯关系是解释关系，其他关系将有类似的公理。

$$\forall e_i, e_j \; Explanation(e_i, e_j) \Rightarrow CoherenceRel(e_i, e_j) \qquad (18.78)$$

变量 e_i 和 e_j 代表两个相关话段所表示的事件（或状态）。在本公理和以下将给出的公理中，量词总是覆盖它们右边的所有事物。本公理告诉我们，假如在两个事件之间需要确立一种连贯关系，一种可能性就是溯因推理假定该关系是说明（Explanation）关系。

说明关系要求第二个话段所表达的是第一个话段表达的结果的原因。我们通过下面的公理来陈述：

$$\forall e_i, e_j \; cause(e_j, e_i) \Rightarrow Explanation(e_i, e_j) \tag{18.79}$$

除了连贯关系的公理，我们也需要代表世界常识的公理。采用的第一个常识公理表明：如果某人喝醉了，我们就不让他开车，前者导致后者（为了简便，用 *diswant* 表示谓词“不让”）

$$\begin{aligned} &\forall x, y, e_i \; drunk(e_i, x) \Rightarrow \\ &\quad \exists e_j, e_k \; diswant(e_j, y, e_k) \wedge drive(e_k, x) \wedge cause(e_i, e_j) \end{aligned} \tag{18.80}$$

在继续讲述之前，我们依次补充两个关于该公理以及将要讲述的其他公理的注解。第一点，在公理（18.80）中采用全称量词来绑定几个变量，这从本质上说明了在所有情形下某人喝醉了，所有人都不会让他开车。尽管这是通常我们希望的情形，但是这个陈述还是过于绝对。在 Hobbs 等系统中，对这一点的处理是在这种公理的前提中引入了另外的关系，称为 *etc* 谓词。*etc* 谓词代表为了应用该公理而必须为真的所有其他属性，但是它太含糊，不能清晰地阐述。因此这些谓词不能被证实，而只能被假定为一个相应的代价。带有较高假定代价的规则的优先性低于较低代价的规则，则应用该规则的可能性可以根据这个代价来计算。因为我们已经选择通过忽略代价以简化讨论，因此将按类似方式忽略 *etc* 谓词的用法。

第二点，每个谓词在论元的第一个位置带有一个看起来好像“多余”的变量；例如，谓词 *drive* 有两个而不是一个变量。该变量被用于具体化由谓词表示的关系，以至于可以在其他谓词的论元位置指向该变量。例如，用变量 e_k 具体化谓词 *drive*，则允许我们通过指向 *diswant* 谓词的最后一个论元来表达不让某人开车的思想。

现在回到讲述的正题，我们采用的第二个有关世界常识的公理是：如果某人不想让其他人去开车，那么他们就不愿意让该人拥有他的车钥匙，因为车钥匙能够使人驾驶汽车。

$$\begin{aligned} &\forall x, y, e_j, e_k \; diswant(e_j, y, e_k) \wedge drive(e_k, x) \Rightarrow \\ &\quad \exists z, e_l, e_m \; diswant(e_l, y, e_m) \wedge have(e_m, x, z) \\ &\quad \wedge carkeys(z, x) \wedge cause(e_j, e_l) \end{aligned} \tag{18.81}$$

第三个公理是：如果某人不想让其他人拥有某件东西，那么他可以将它藏起来。

$$\begin{aligned} &\forall x, y, z, e_i, e_j \; diswant(e_l, y, e_m) \wedge have(e_m, x, z) \Rightarrow \\ &\quad \exists e_n \; hide(e_n, y, x, z) \wedge cause(e_l, e_n) \end{aligned} \tag{18.82}$$

最后一个公理很简单：原因是可传递的，也就是说，如果 e_i 导致 e_j，e_j 导致 e_k，则 e_i 导致 e_k。

$$\forall e_i, e_j, e_k \; cause(e_i, e_j) \wedge cause(e_j, e_k) \Rightarrow cause(e_i, e_k) \tag{18.83}$$

最后，我们引入话段本身的内容，即John藏起了Bill的汽车钥匙（John hid Bill's car keys (from Bill)），

$$hide(e_1, John, Bill, ck) \wedge carkeys(ck, Bill) \tag{18.84}$$

并且用代词 *he* 来描述某人喝醉了；我们将用自由变量 *he* 表示该代词，

$$drunk(e_2, he) \tag{18.85}$$

现在即可看到怎样通过话段的内容和前面提及的公理在解释关系下确立例句（18.71）的连贯。图 18.9 对该推导过程进行了总结；方框中所示的是句子的解释。我们从假定存在一个连贯关系开始，利用公理（18.78）推测该关系是说明关系，

$$Explanation(e_1, e_2) \tag{18.86}$$

通过公理（18.79），我们推测

$$cause(e_2, e_1) \tag{18.87}$$

成立。通过公理（18.83）可以推测这里有一个中间原因 e_3，

$$cause(e_2, e_3) \wedge cause(e_3, e_1) \tag{18.88}$$

再次重复该公理，将公理（18.88）的第一个因子扩展为含有中间原因 e_4。

$$cause(e_2, e_4) \wedge cause(e_4, e_3) \tag{18.89}$$

我们从公理（18.84）第一个句子的解释获得 *hide* 谓词，以及公理（18.88）的第二个 *cause* 谓词，并且，利用公理（18.82），可以推测 John 不让 Bill 拥有他的汽车钥匙：

$$diswant(e_3, John, e_5) \wedge have(e_5, Bill, ck) \tag{18.90}$$

根据上式以及公理（18.84）中的 *carkeys* 谓词，公理（18.89）中的第二个 *cause* 谓词，我们可以利用公理（18.81）推测出 John 不让 Bill 驾驶：

$$diswant(e_4, John, e_6) \wedge drive(e_6, Bill) \tag{18.91}$$

根据上式、公理（18.80）以及公理（18.89）中的第二个 *cause* 谓词，我们可以推测出 Bill 喝醉了。

$$drunk(e_2, Bill) \tag{18.92}$$

现在，我们发现如果简单地假设自由变量 *he* 绑定于 Bill，就可以从第二个句子的解释中“证实”该事实。因此，在识别句子的解释之间的推理链的过程中，我们确立了句子的连贯。本例中的推理链包括关于公理选择和代词指派的无法证实的假设，并生成了确立说明关系需要的 $cause(e_2, e_1)$。

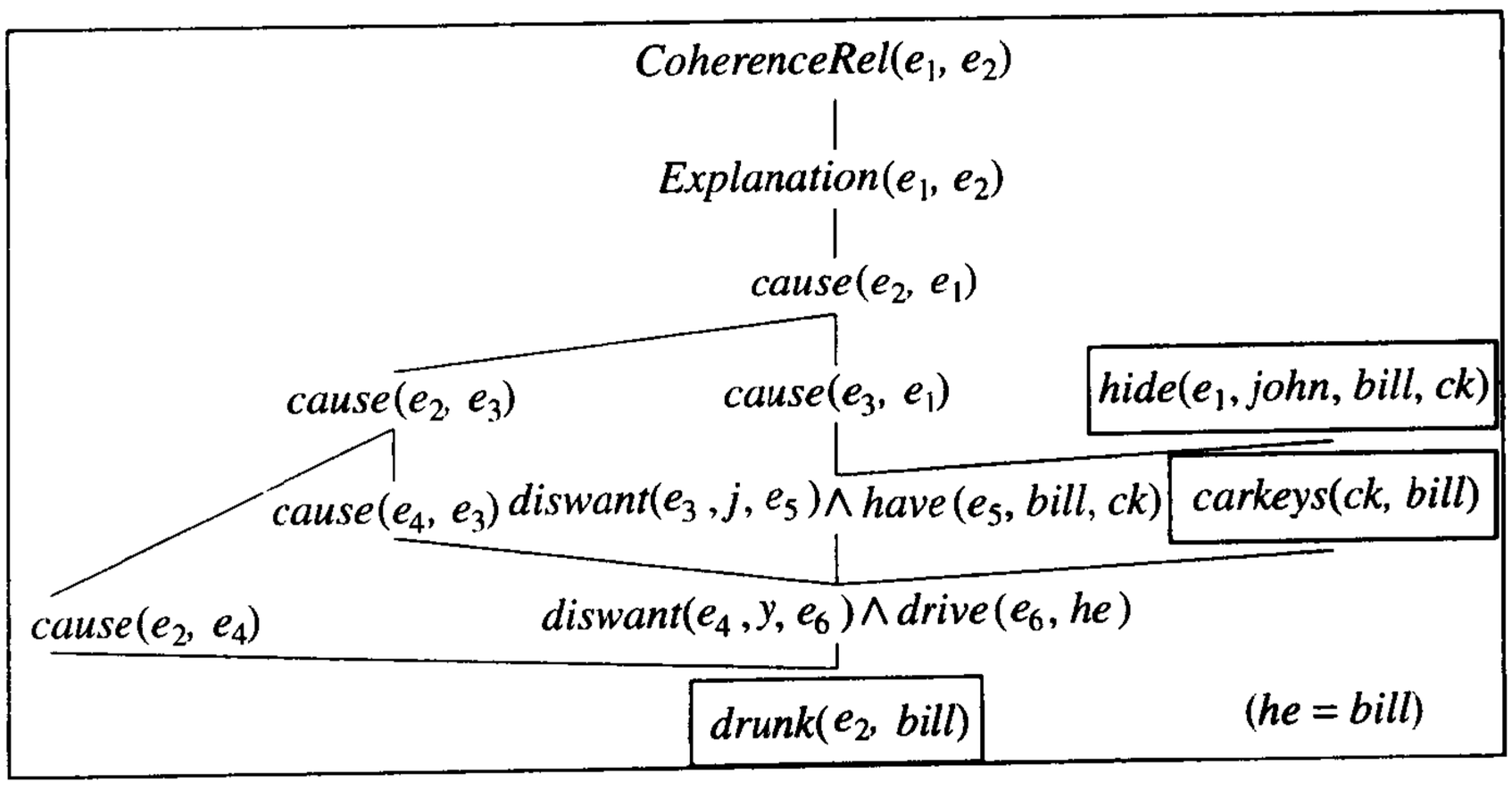


图 18.9 例句（18.71）的连贯的确立

这个推导过程以例子说明连贯确立具有强有力的特性，也就是说，它能够导致听话人推理出话语中说话人未说的信息。在这个例子中，推理所需的假设是：John 藏起了 Bill 的钥匙，因为他不想让他开车（大概是由于怕出事故，或被警察逮到），而不是因为其他原因，例如对他的恶作剧。这个原因在例句（18.71）的任何地方都没有提到，只是出现在确立连贯所需的推理过程中。从这个角度看，话语的意义大于它的每个部分意义的相加。也就是说，通常话语所传递的信息远大于组成该话语的单个句子的解释所包括的所有信息。

现在我们回到例句（18.72），将其重新编号为例句（18.94）。它的特别之处在于缺少例句（18.71）的连贯性，该例句现在被重新编号为例句（18.93）。

John hid Bill's car keys. He was drunk. （18.93）

?? John hid Bill's car keys. He likes spinach.　　(18.94)

现在来看看为什么会这样：它缺少类似的能够连接两个话段表示的推理链，特别是缺少类似于公理(18.80)的原因，公理能够说明喜欢菠菜可能导致某人不能驾驶。在缺乏能够支持推理链的额外信息的情况下(例如前面提及的情节，某人对 John 承诺用菠菜换取 Bill 被藏起的汽车钥匙)，就不能确立段落的连贯。

但是，溯因推理是非可靠推理的一种形式，必须能够在以后的处理中撤销溯因推理所得到的假设，即溯因推理是可废止的(defeasible)。例如，如果紧跟例句(18.93)的是例句(18.95)，

Bill's car isn't here anyway; John was just playing a practical joke on him.　　(18.95)

则系统将不得不撤销连接例句(18.93)中两个句子的原先的推理链，并用事实(藏钥匙事件是恶作剧的一部分)来替代。

对于为支持较大范围的推理而设计的更全面的知识库，需要使用比那些在确立例句(18.93)的连贯时所采用的更概括的公理。例如，研究公理(18.81)：如果你不想让某人驾驶汽车，就不想让他拥有车钥匙。该公理的一个更概括的形式是：如果你不想让某人进行某个行为，而某个物体能够让他进行该行为，则你就不想让他拥有该物体。所以，汽车钥匙能够让某人驾驶汽车的事实被分离出来，而实践中存在许多其他类似的事实。同样地，公理(18.80)称，如果某人喝醉了，则不让他驾驶汽车。我们可以用下面的公理来替代：如果某人不想让某件事发生，则他不愿意让可能导致该件事的原因发生。我们再一次将人们不让其他人卷入汽车事故的事实与酒后驾车导致事故的事实分离来。

尽管拥有能够阐明连贯确立问题的计算模型非常重要，但是该方法和其他类似方法很难用于覆盖范围广泛的应用。特别是，大量的公理需要对世界中所有必需事实进行编码，并且缺少利用这种大规模公理集进行约束推理的鲁棒的机制，这使得这些方法在实践中几乎无法实施。非正式地说，这个问题被称为 AI 完全问题(AI-complete)，引自计算机科学中的术语 *NP* 完全(*NP*-complete)问题。AI 完全问题是指本质上需要所有知识(人类拥有的)并能够利用这些知识的问题。

学者们也提出了一些分析话语连贯结构的其他方法。使用最广的方法称为修辞结构理论(Rhetorical Structure Theory，简称 RST)(Mann and Thompson, 1987)，它给出了由 23 个修辞关系(rhetorical relation)组成的集合，这些修辞关系能够保持在话语中不同跨度的文本之间。尽管 RST 是面向文本描述而不是文本解释的，但是已经证实对于开发自然语言生成系统，它是一个有用的工具。我们将在 20.4 节给出 RST 的更详细的介绍。

连贯(coherence)和同指(coreference)　读者可能已经注意到了，说明例句(18.93)是连贯的证据具有另一个有趣的特征。尽管代词 *he* 最初是一个自由变量，但是在推理过程中它被绑定于 Bill。其实，并不需要一个独立的判定代词的处理，连贯确立的过程中就可以作为边界效应生成。除了前述的树查询算法，Hobbs(1978)提出采用连贯确立机制作为代词解释的又一种方法。

这种方法对为什么例句(18.93)中代词的最自然理解是 Bill，而例句(18.96)中代词的最自然理解是 John，给出了说明。

John lost Bill's car keys. He was drunk.　　(18.96)

例句(18.96)在说明关系下确立的连贯需要这样一个公理：喝醉能够导致某人丢失某些东西。因为这样的公理规定了喝醉的人与丢东西的人必定是同一人，所以表示代词的自由变量就只能绑定

为 John。例句（18.96）和例句（18.93）之间具有的词汇－句法差异仅在于首句子中的动词。代词和可能的先行名词短语的语法位置在两个例子中都是相同的，因此基于句法的优先关系是无法区分它们的。

话语连接词（discourse connective） 有时，说话人会加入特别的线索，我们称其为连接词（connective），它用于约束两个或更多话段之间的连贯关系集。例如，例句（18.97）中的连接词 *because* 被用于清楚地表明这种说明（explanation）关系。

John hid Bill's car keys because he was drunk. （18.97）

because 的意义可以表示为 $cause(e_2, e_1)$，在证明中所扮演的角色类似于通过公理（18.79）溯因地引入的 *cause* 谓词。

但是，连接词并不能总是将所有的可能性约束为惟一的连贯关系。例如，在下面的例句（18.98）、例句（18.99）和例句（18.100）中，*and* 的意义分别与平行（parallel）、时机（occasion）和结果（result）关系一致。

John bought an Acura and Bill leased a BMW. （18.98）

John bought an Acura and drove to the ballgame. （18.99）

John bought an Acura and his father went ballistic. （18.100）

而 *and* 与说明（explanation）关系不一致；与例句（18.97）不同，例句（18.101）的意义不可能与例句（18.93）相同。

John hid Bill's car keys and he was drunk. （18.101）

尽管连贯判定处理可以使用连接词来约束连贯关系（可以从一对话段之间推理得出）的范围，但是它们本身并不能“造成”连贯。任何由连接词预示的连贯关系仍然必须被确立。因此，给例句（18.94）添加连接词并不能使它连贯。

?? John hid Bill's car keys because he likes spinach. （18.102）

这里，连贯确立失败的原因与例句（18.72）相同，即缺少能够将喜欢菠菜与导致某人藏起汽车钥匙联系在一起的因果知识。

18.3 话语结构

上一节讲述了如何确立一对句子的连贯。现在来研究对于较长的话语如何确立连贯。是不是只要简单地确立所有相邻句对的连贯关系就行了呢?

答案是否定的，这已经得到了证明。正如句子具有结构（即句法），话语也是有结构的。我们来研究例句（18.103）。

（18.103）

John went to the bank to deposit his paycheck.(S1)

He then took a train to Bill's car dealership.(S2)

He needed to buy a car.(S3)

The company he works for now isn't near any public transportation.(S4)

He also wanted to talk to Bill about their softball league.(S5)

直觉地，例句（18.103）的结构不是线形的。该话语似乎本质上是关于句子 τ1 和 τ2 中描述的事件的序列，与句子 S3 和 S5 最相关的是 S2，与句子 S4 最相关的是 S3。图 18.10 中给出的是这些句子之间的连贯关系所导致的话语结构。

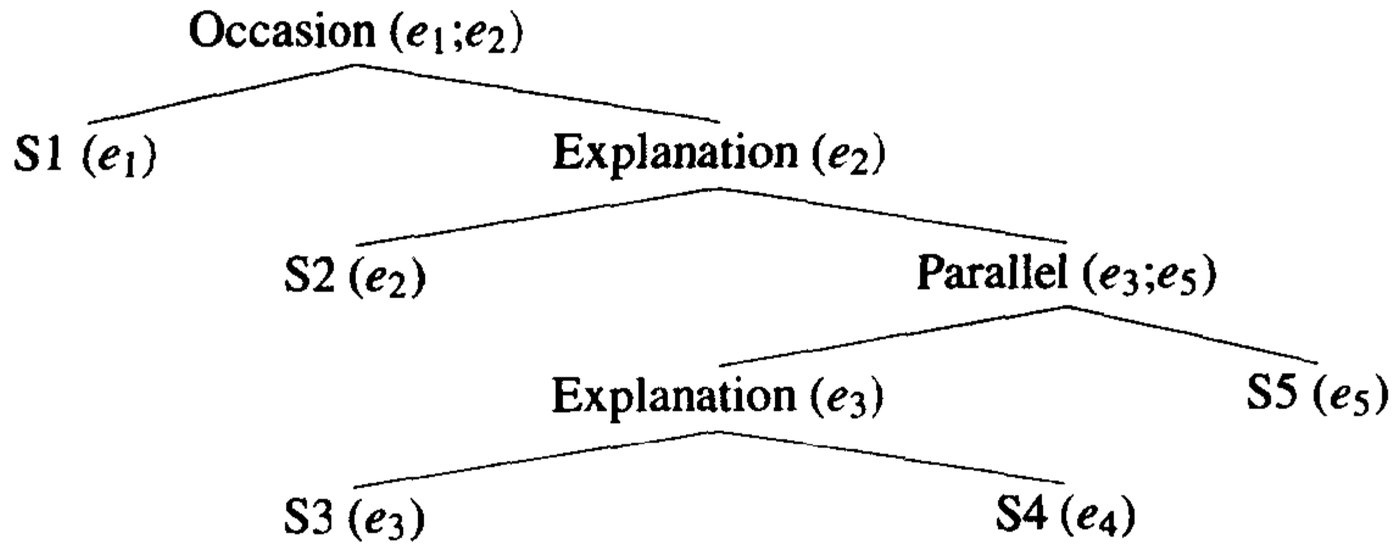


图 18.10　例句（18.103）的话语结构

在树中，代表一组局部连贯话段的结点称为话语片断（discourse segment）。粗略地说，话语中的话语片断就相当于句法中的成分。

我们可以通过扩展上一节中所采用的话语解释公理来确立像例句（18.103）那样较长且有层次的话语的连贯。话语片断和最终话语结构的识别是该处理的副产品。

首先，我们引入公理（18.104），它表明某个句子是一个话语片断，其中 w 是句中单词的字符串，e 是它所描述的事件。

$$\forall w, e \; sentence(w,e) \Rightarrow Segment(w,e) \tag{18.104}$$

然后，我们引入公理（18.105），它表明如果在两个较小的片断之间能够确立连贯关系，那么它们就可以组成一个较大的片断。

$$\begin{aligned} \forall w_1, w_2, e_1, e_2, e \; Segment(w_1, e_1) \wedge Segment(w_2, e_2) \\ \wedge CoherenceRel(e_1, e_2, e) \Rightarrow Segment(w_1 w_2, e) \end{aligned} \tag{18.105}$$

注意，将公理扩展用于较长的话语时，需要对 *CoherenceRel*(*e*)谓词增加第三个论元。该变量的值是 e_1和e_2所表达的信息的组合，它代表结果片断的主要声明。这里将假定从属关系（subordinating relation）（例如，说明）只与一个变量有关（在本例中指第一个句子，即结果），而并列关系（coordinating relation）（例如，平行和时机）与两个变量的组合有关。在图 18.10 中，这些变量出现在每个关系旁边的括号里。

现在来解释一段连贯的文本 W，就像例句（18.106）所表达的，我们必须简单地证明该文本是片断。

$$\exists e \; Segment(W,e) \tag{18.106}$$

对于一个话语，这些规则将导出任何可能的二元分支的片断结构，只要该结构能够被片断之间连贯关系的确立所支持。在此，句子句法结构（见第 9 章）和话语结构的计算之间有一个区别。通常句子层的语法是很复杂的，牵扯到许多关于不同成分（名词短语、动词短语等）怎样才能彼此修饰以及以何种次序修饰的句法事实。与之相反，上面所提的“话语语法”就简单得多，仅牵扯两个规则：一个片断可以改写为两个较小的片断，以及一个句子就是一个片断。实际指派那个可能的结构依赖于如何确立该段落的连贯。

为什么要计算话语结构呢？因为不少应用都可以从中获益。例如，利用话语结构，摘要系统可以只选择话语中的中心句，而摒除次要的信息。当应用于例句（18.103）时，生成简单摘要的系统

可能只会选择句子S1和S2，因为事件表示被传往顶层结点。生成更详细摘要的系统也可能会包括句子S3和S5。类似地，信息检索系统也可能对位于话语结构高层部分的句子所带有的信息给予比其他信息更大的权重，生成系统也需要话语结构知识，以生成连贯话语，有关内容将在第20章讲述。

话语结构对于自然语言的子任务也是十分有用的，例如代词判定。从18.1节已经知道代词表现出一种称为新近的优先关系，也就是它们更倾向于指向附近的所指对象。现在我们对新近有两种可能的定义：按照话语线性顺序的新近，或者根据话语层级结构的新近。实际上后一种定义已经被认为是正确的，尽管不可否认在所有例子中该事实并不都是完全清楚的。

本节简单讲述了获得话语结构的几种可能方法之一。另一种不同的方法常应用于对话，将在19.4节讲述。

18.4 所指和连贯的心理语言学研究

本章所描述的这些技术在多大程度上模拟出了人类话语理解的过程呢？针对这一问题进行了大量的心理语言学研究。

例如，许多研究工作是关于人们进行代词解释时在多大程度上使用了18.1节所描述的优先关系的，而研究结果常常是矛盾的。Clark and Sengal(1979)采用阅读时间实验（reading time experiment）研究了代词解释中句子新近性的影响。在收到并确认已阅读一段由三个句子组成的上下文后，包含代词的目标句被呈现给受试者。当受试者感到已经理解了这个目标句时，就按下按钮。Clark和Sengal发现：当代词的所指对象是从上下文中最近的从句唤起时，所用的阅读时间远远小于所指对象从前面两到三句唤起时的阅读时间。另外，所指对象是从向后二句还是向后三句被唤起，并没有太多的差别，因此他们认为："在工作记忆中，所处理的最后一个从句带给它所提及的实体一种特权地位"。

Crawley et al.(1990)比较了语法角色平行优先关系和语法角色优先关系，特别是从前面句子的主语位置唤起的所指对象优先于从宾语位置唤起的所指对象的关系。与前人混合这些优先关系，只考虑主语到主语（subject-to-subject）所指影响的研究不同，Crawley等研究宾语位置的代词，以确定它们是否更倾向于被指派为前一个句子的主语或宾语。他们对两种不同的任务环境：问题回答任务（question answering task）和所指对象指定任务（referent naming task）进行了研究。从问题回答任务可以看出受试者如何解释代词，而在所指对象指定任务中，受试者直接给出代词的所指对象。他们发现，受试者将代词判定为前面句子主语的情形远多于判定为宾语的情形。

然而，Smyth(1994)批评Crawley等评测平行的角色时的数据不具备充分性。采用满足评估平行的更严格要求的数据，Smyth发现在所指对象指定任务中，受试者在绝大多数情形下遵循了平行优先关系。该实验对主语所指对象优于宾语所指对象提供了微弱的支持，他将这作为提问句子不是十分平行时的默认策略。

Caramazza et al.(1977)研究了代词判定时动词的"隐含因果关系"（implicit causality）的影响。利用句子完成任务（sentence completion task），根据动词所具有的主语倾向性和宾语倾向性来划分动词。受试者所看到的是类似下面例句（18.107）的句子片断。

John telephoned Bill because he （18.107）

从受试者完成的句子中，实验设计者可以看出受试者对该代词所偏爱的所指对象。绝大部分受试者都表明具有语法主语或宾语优先关系的动词被划分为具有主语倾向性或宾语倾向性。接着对每个具有倾向性的动词构建一对例句：一个是"一致"例句，语义上支持由动词倾向性所预示的代词指派；

另一个是“不一致”例句，语义上支持相反预示。例如，对于具有主语倾向性的动词*telephone*，例句（18.108）是一致的，因为第二个从句的语义支持把主语*John*指派为*he*的先行词，而例句（18.109）是不一致的，因为语义支持的指派是宾语*Bill*。

John telephoned Bill because he wanted some information.　（18.108）

John telephoned Bill because he withheld some information.　（18.109）

在所指对象指定任务中，Caramazza等发现对一致例句的指定时间快于不一致例句。也许令人吃惊，这即使在第一个从句中所提及的两个人的性别不同的情形下（这使得所指对象并没有歧义）也是如此。

Garnham et al.(1996)对有关隐含因果关系可能影响代词判定方式的两个假设：焦点假设（focus hypothesis）和融合假设（integration hypothesis）进行了区分。焦点假设，也许是受到Caramazza等实验的启发，指出这种动词对特定的语法角色的填充者有促发效果（priming effect），因此贡献的信息能用于代词解释的那一点。融合假设中该信息只能用于从句被完全理解并与前面的话语融合以后。他们利用试探性任务（probing task）试图确定哪种假设是正确的。在给出一些句子组成的上下文后，以每次一个单词的方式呈现出包含代词的句子。在呈现的适当时机，其中一个可能的所指对象的名字被显示出来，向受试者提问迄今为止那个人是否已经被提及。Garnham等人发现通常隐含因果关系的信息倾向性在给出该代词后并不会立刻获得，而是稍后才被句子利用。

Matthews and Chodorow(1988)对句内的所指问题和基于句法的查询策略进行了分析。在问题回答任务中，他们发现受试者理解代词先行词占据早期的句法深层位置的句子时，需要的时间多于先行词占据新近的句法浅层位置的句子。这个结果与Hobbs的树查询算法所采用的查询处理一致。

也有一些心理语言学的研究是关于中心理论原则的测试的。通过一组阅读时间实验，Gordon et al.(1993)发现在当前的向后看中心被指向一个全部为名词的短语而不是代词时，阅读时间会变长，即使该代词是有歧义的，而固有名称没有歧义。这个效应被称为重复名称惩罚（repeated name penalty），仅出现于主语位置的所指对象，暗示C_b被优先实现为主语。Brennan(1995)分析了如何选择与中心原则联系起来的语言学形式。她设计了一组受试者看篮球比赛并将其描述给另一个人的实验。她发现，在后续代词化实体之前，受试者倾向于用主语位置的全部名词的短语来指向该实体，即使在宾语位置这个所指对象已经被引入。

心理语言学家也对人们确立话语连贯的过程进行了研究。其中有些研究集中于推理控制（inference control）问题，也就是说，在实际解释中从潜在的无限多可能推理中选择哪一个(Singer, 1994; Garrod and Sanford, 1994)。这可以分为必须推理和精致推理，必须推理对确立连贯是必需的，而精致推理是由文本暗示但对确立连贯不是必需的推理。在解释中只做出必需推理的见解被称为延迟推理理论（deferred inference theory）(Garnham, 1985)和最低限度立场（minimalist position）(McKoon and Ratcliff, 1992)。与代词解释一样，检验这些问题的研究成果充满潜在的矛盾。当然，每种情形下所得到的结果在很大程度上依赖于实验设计和范例(Keenan et al., 1990)。

例如，Johnson et al.(1973)采用识别判断任务（recognition judgement task）对这一问题进行检验。他们将类似段落（18.110）的段落传给受试者。

When the man entered the kitchen he slipped on a wet spot and dropped
the delicate glass pitcher on the floor. The pitcher was very expensive,
and everyone watched the event with horror.　（18.110）

随后传给受试者直接从段落中选择的一个句子，例如段落（18.110）中的第一句，或者像例句（18.111）那样以一种预期结果的方式引入精致推理的句子。

The man broke the delicate glass pitcher on the floor. (18.111)

然后询问受试者，所看到的句子是否与段落中的某个句子一字不差。这两种类型的句子所获得的识别率都在60%左右，而完全改变了句子意义的那些句子的识别率远低于通常的情况（约为22%）。通过一个类似但记录受试者反应时间的实验，Singer(1979)对这些推理发生的时间是在原始句子被理解时（这是真正的思考）还是在预期结果版本出现时，进行了研究。尽管Singer发现相同和预期结果两种版本的句子产生正面反应的效率是类似的，但判断预期结果版本的句子所需的时间比相同句子的时间多0.2~0.3 s，这暗示着推理并不是在理解时做出的。

Singer(1980)采用如例句（18.112）至例句（18.114）的句子来检验不同类型的推理发生的时间。

The dentist pulled the tooth painlessly. The patient liked the new method. (18.112)

The tooth was pulled painlessly. The dentist used a new method. (18.113)

The tooth was pulled painlessly. The patient liked the new method. (18.114)

每个句子都紧跟着测试句（18.115），呈现给受试者。

A dentist pulled the tooth. (18.115)

测试句（18.115）中所表达的信息在例句（18.112）中明显地提及过，它是例句（18.113）确立连贯所必需的，在例句（18.114）中是精致推理。Singer发现在前两个例子中受试者验证的时间几乎是相同的，但对于精致的例子要慢0.25 s，这是对延迟推理理论的支持。

Kintsch和他的同事提出并分析了话语理解的“构建－集成”（construction-integration）模型(Kintsch and van Dijk, 1978; van Dijk and Kintsch, 1983; Kintsch, 1988)。他们定义了一个文本宏观结构（text macrostructure）的概念，这是一个具有层级的命题构成的网络，它对文本全局内容给出了抽象的语义描述。Guindon and Kintsch(1984)采用词典启发技术（lexical priming）来评估构建宏观结构及伴随的理解处理时是否需要精致推理。首先受试者阅读一段话，然后问他们某个特定的单词对是否在所读文本中出现过。单词对分为三种类型：文中没有提及但与文本宏观结构相关的单词对，主题相关但与文本宏观结构无关的干扰词对，以及主题无关的干扰词对。受试者错误地认为该单词曾在文本中出现过，则称为“假警报”（false alarm）。文本宏观结构相关的单词对的假警报数远大于主题相关词对，而主题相关词对的假警报数大于主题无关的词对。在受试者正确判断出该单词没在文本中出现过的情形下，文本宏观结构相关的单词对的响应时间远长于主题相关词对，主题相关词对的响应时间长于主题无关词对。

Myers et al.(1987)对句子之间因果关系的程度对句子理解的时间和回忆精度的影响进行了研究。研究的目标句如例句（18.116）所示，

She found herself too frightened to move. (18.116)

他们设计了4个上下文句子，如例句（18.117）至例句（18.120）所示，它们与例句（18.116）形成的因果关系是一个从高到低的连续集。

Rose was attacked by a man in her apartment. (18.117)

Rose saw a shadow at the end of the hall. (18.118)

Rose entered her apartment to find a mess. (18.119)

Rose came back to her apartment after work. (18.120)

呈现给受试者的是由上下文句子和目标句组成的因果关系句对。Myers等发现越是因果关系强的句对所花费的阅读时间越少。在受试者阅读了许多此类句子以后，Myers 等开始进行线索回忆（cued recall）实验。在该实验中，从句对中挑选一个句子给受试者，然后问他有关另一个句子的尽可能详细的情况。他们发现，对于那些因果关系较强的句对，受试者所能回忆起来的内容也较多。

18.5　小结

本章探讨了自然语言处理系统面临的句子之间（即话语层）的运算。下面是本章主要内容的总结：

- 话语解释（discourse interpretation）需要对话语状态建立一种可演变的表示，即话语模型（discourse model）。话语模型包含对已提及的实体以及它们所承担的关系的表示。
- 自然语言提供了许多指向实体的方法。所指的每种形式都将有关它自己如何与话语模型和世界看法集一起被加工的信号传递给听话人。
- 代词所指可被用于话语模型中具有足够显著度的所指对象。各种词汇、句法、语义以及话语的因素似乎都会影响显著性。
- 根据 Lappin and Leass(1994)，在代词判定算法中，这些因子相互参照进行建模和设定权重，在某些情况下可获得 80% 左右的性能。
- 话语不是任意收集的一些句子，它们必须是连贯的（coherene）。把良构并可独立理解的几个句子放置在一起的结果常常是不连贯的话语。
- 应用一个或多个连贯关系（coherence relation）施加的约束而进行的确立连贯的处理，经常导致对说话人未提及的额外信息的推理。溯因逻辑推理的非可靠规则可以用于进行这种推理。
- 与句子一样，话语也具有层级结构。话语中间一组局部连贯的话段被称为话语片断（discourse segment）。话语结构识别可以看成是话语解释的副产品。

18.6　文献和历史说明

基于自然语言理解早期系统(Woods et al., 1972; Winograd, 1972b; Woods, 1978)所实现的基本集，话语计算方法中的许多基础性工作是在 20 世纪 70 年代晚期完成的。Webber(1978, 1983)的研究工作为话语模型中如何表示实体以及容许后续所指的方式提供了研究基础。她给出的许多例子目前仍在继续挑战所指理论。Grosz(1977a)对话语展开时谈话参与者所保持的注意力的焦点进行了研究。她定义了焦点的两个层次：与整个话语有关的实体称为全局焦点（global focus）；局部关注（大多数为一个特定话段的中心）的实体称为紧邻焦点（immediate focus）。Sidner(1979, 1983)描述了一种跟踪话语（紧邻）焦点的方法以及这些焦点在代词和指示名词短语的判定中的使用。她给出了当前话语焦点和可能焦点之间的区别，而它们分别是中心理论的向后看中心和向前看中心的前身。

中心方法源自 Joshi and Kuhn(1979)以及 Joshi and Weinstein(1981)，它给出了紧邻焦点和集成当前话段到话语模型所需的推理之间的关系。Grosz et al.(1983)将该研究与 Sidner 和 Grosz 早期的研究工作结合在了一起。这形成了中心方法的草稿，尽管从 1986 年以来它已经广为流传，但是它的发表是在论文 Grosz et al.(1995)中。其后，基于该草稿和论文发表了一系列关于中心方法的文章(Kameyama, 1986; Brennan et al., 1987; Di Eugenio, 1990; Walker et al., 1994; Di Eugenio, 1996; Strube and Hahn, 1996; Kehler, 1997a)。有关中心方法的最新的论文集可参阅 Walker et al.(1998)。

语言学界的研究者给出了关于话语模型中所指对象持有的信息状态（information status）的原因(Chafe, 1976; Prince, 1981; Ariel, 1990; Prince, 1992; Gundel et al., 1993; Lambrecht, 1994)。例如，

Prince(1992)根据交叉的二分状态(听话人状态和话语状态)来分析信息状态,并给出了这些状态与所指语的语法位置的关系。另外,Gundel et al.(1993)根据假定的单维度的6个状态,称为约定层级(givenness hierarchy),建立了它们与所指语的语言学形式之间的关系。

自Hobbs(1978)的树查询算法开始,研究者对于能够鲁棒地应用于自然出现文本的基于句法的识别所指方法进行了深入研究。基于Lappin and Leass(1994)的研究,Kennedy and Boguraev(1996)描述了一个类似地但不依赖于完全的句法剖析器的系统,它所依赖的是一种只识别名词短语并标注其语法角色的机制。这两种方法都采用了Alshawi(1987)的集成显著因子的方案。把该方案用于多通道(即语音和手语)人机交互界面的算法可参阅论文Huls et al.(1995)。对计算系统中所指的各种方法的讨论可参阅Mitkov and Boguraev(1997)。

近期,一些研究者提出了基于监督学习的代词判定方法(Connolly et al., 1994; Aone and Bennett, 1995; McCarthy and Lehnert, 1995; Kehler, 1997b; Ge et al., 1998)。在这些方法中,机器学习方法,例如贝叶斯模型归纳、决策树和最大熵模型等,用于从标注所指关系的语料中训练模型。对所指语料标注中可能出现问题的讨论可参阅Poesio and Vieira(1998)。

MUC-6信息抽取评测包括对同指的一般性评测(Sundheim, 1995a)。该任务包括专有名称、别名、定名词短语、裸名词(bare noun)和代词之间的共指,甚至包括一些由句法关系,例如谓词性名词(*The Integra is the world's nicest looking car*)和同位语(*the Integra, the world's nicest looking car*)所预示的共指。系统的性能是通过计算基于差异的召回率和精度来评测的,这里的差异指系统生成的共指描述的等价类与那些手工标注的答案之间的差异。参与评测的7个机构中的5个获得的召回率在51%~63%之间,精度在62%~72%之间。MUC-7也进行了类似的评测。

一些研究者对话语的话段之间能够持有的连贯关系进行了假定(Halliday and Hasan, 1976; Hobbs, 1979a; Longacre, 1983; Mann and Thompson, 1987; Polanyi, 1988; Hobbs, 1990; Sanders et al., 1992)。Hovy(1990)给出了曾在文献中提出的超过350个此类关系的概览。语言学话语模型(Polanyi, 1988; Scha and Polanyi, 1988)是一种更强调话语句法的理论框架,在该方法中,基于从句-从句建立了话语的剖析树,直接对应于基于成分-成分建立的句子的剖析树。最近的研究方向是把树邻接语法体系应用于话语剖析(Webber et al., 1999)及其参考文献。除了确定话语结构和意义,话语连贯理论也被用于话语层语言现象解释的算法,这些语言现象包括代词判定(Hobbs, 1979a; Kehler, 2000),动词短语省略与间隙(Prüst, 1992; Asher, 1993; Kehler, 1993, 1994a)以及时态解释(Lascarides and Asher, 1993; Kehler, 1994b, 2000)。对连贯关系和话语连接词之间关系的深入研究可参阅Knott and Dale (1994)。

第19章　对话与会话智能代理

C: I want you to tell me the names of the fellows on the St. Louis team.
A: I'm telling you. Who's on first, What's on second, I Don't Know is on third.
C: You know the fellows' names?
A: Yes.
C: Well, then, who's playing first?
A: Yes.
C: I mean the fellow's name on first.
A: Who.
C: The guy on first base.
A: Who is on first.
C: Well what are you askin' *me* for?
A: I'm not asking you – I'm telling you. Who is on first.①

Who's on First – Bud Abbott and Lou Costello's version of an old burlesque standard.

文艺作品中充满了对无生命的物体被神奇地赋予感觉和语言天赋的描写。从Ovid的皮格马利翁（Pygmalion）雕像到Mary Shelley的人形怪物（Frankenstein）以及曹雪芹的《红楼梦》里描述的日落时在庭院里等待晚霞映照在雪白镜面上的那块会发冷光的神奇怪石，在这些文学艺术作品中，深深打动我们的是创造出一个充满神奇的东西，然后与它聊天。传说Michelangelo（米开朗基罗）在完成他的雕塑摩西（Moses）以后，感到它是那么逼真以至于轻叩它的膝盖并要求它说话。也许这不应该让我们感到吃惊。语言本身长久以来就是人性和感知的标志，会话或对话是语言能够尽情展现的最基本且最具特权的舞台。当然它也是我们自孩提时代以来学习的第一种语言，对我们中的大多数人来说，无论我们是点午餐的咖喱粉、买邮票，还是参与商业会谈、与我们的家人聊天、预订机票或抱怨天气，它是我们沉湎其中的一种语言。

本章介绍会话智能代理中的基本结构和算法。会话智能代理是一种能够使用自然语言与用户进行交流的程序，通过它可以预订机票、回答问题或回复电子邮件。其中的许多问题也是与商业会议摘要（business meeting summarization）系统以及其他口语理解系统（它必须转录并摘要类似会议

---

① 在这段对话中，队员的名字分别为“谁”、“什么”和“不知道”，这使得对话十分滑稽。译文如下，供读者对照：
C: 我让你告诉我St. Louis队那些家伙的名字。
A: 我正要告诉你。谁在一垒，什么在二垒，我不知道在三垒。
C: 你知道那些家伙的名字?
A: 是。
C: 好吧，那么，谁第一个上场?
A: 是。
C: 我是说在一垒的那个家伙的名字。
A: 谁。
C: 在一垒的家伙。
A: 谁在一垒。
C: 唉，你想问我什么?
A: 我没有问你——我是在告诉你。谁在一垒。

的结构化会话）相关的。从19.1节开始引入一些导致会话区别于其他话语的问题，讲述话轮转换、会话基础和会话隐涵等概念。19.2节介绍言语行为或对话行为，19.3节给出自动言语行为解释的两个不同算法。19.4节描述对话中的结构和连贯与话语中的结构和连贯（见第18章）的差别。最后，19.5节介绍在为对话管理（会话智能代理的一部分）选择架构时应该如何面对这些问题。

19.1 什么使对话出现差别

对话的大部分特征与其他类型的话语，例如在第18章中讲述的文本独白，都是相似的。对话也表现出了指代、话语结构和连贯等特点，尽管这与对白有稍许差别。例如，在对话中判定所指时，很重要的一点是看看其他说话人说了什么。在图19.1（下面将详细讨论）的航空旅行的对话片断中，如果要确定话段 *C* 中代词 *they* 的所指是 *non-stop flights*，就需要看看前面的话段 *A*。

A_4: Right. There's three non-stops today.

C_5: What are they?

然而，从更深层次来看，对话确实与书面的独白不同。下面几个小节将重点讲述其中的一些差别。

19.1.1 话轮和话段

独白和对话之间的一个差别是对话具有话轮转换（turn-taking）的特征。说话人A说一些事情，然后说话人B，然后说话人A，不断变化。图19.1给出的是添加话轮标记以后的一段对话的样例。我们之所以选择这段对话，是因为它属于旅游计划领域，而该领域是近期许多人机对话研究所关注的领域。

C_1:	. . . I need to travel in May.
A_1:	And, what day in May did you want to travel?
C_2:	OK uh I need to be there for a meeting that's from the 12th to the 15th.
A_2:	And you're flying into what city?
C_3:	Seattle.
A_3:	And what time would you like to leave Pittsburgh?
C_4:	Uh hmm I don't think there's many options for non-stop.
A_4:	Right. There's three non-stops today.
C_5:	What are they?
A_5:	The first one departs PGH at 10:00am arrives Seattle at 12:05 their time. The second flight departs PGH at 5:55pm, arrives Seattle at 8pm. And the last flight departs PGH at 8:15pm arrives Seattle at 10:28pm.
C_6:	OK I'll take the 5ish flight on the night before on the 11th.
A_6:	On the 11th? OK. Departing at 5:55pm arrives Seattle at 8pm, U.S. Air flight 115.
C_7:	OK.

图19.1 客户（C）和旅行代理人（A）之间的电话对话的片断

说话人怎样知道什么时间轮到他说话呢？我们来研究类似图 19.1 的会话中话段的时间安排。首先，这个对话中没有明显的重叠。也就是说，每个说话人开始说话的时间接着前面说话人说话结束的时间（重叠话语的周围将用符号 # 表示）。在美国英语会话中，实际出现重叠的话语数似乎很少；Levinson(1983)认为总体上少于 5%，而在类似图 19.1 的面向任务的对话中可能更少。如果说话人的谈话不重叠，那么他们也许在其他说话人说完之后会等一会再说。这种现象也是不多见的。话轮之间的时间非常短暂，即使在多方话语中通常也少于几百毫秒。实际上，下一个说话人开动脑筋准备生成话段的时间就可能大于几百毫秒，这意味着在前一个说话人结束之前，说话人已经开始准备下面的话段。而如果真是这样，自然会话就必须以这样一种方式建立，即（大部分时间）人们能够快速地领悟到谁将是下一个说话人，以及确切的说话时间。这种话轮转换的行为通常是会话分析（Conversation Analysis，简称 CA）领域的研究内容。在一篇重要的会话分析文章Sacks et al.(1974)中，认为话轮转换的行为，至少在美国英语中，是由一组话轮转换规则制约的。这些规则被应用于合适转换位置（transition-relevance place，简称 TRP）；这些位置的语言结构容许转换说话人。下面是话轮转换规则的简化版本，归纳为一个由三部分组成的规则，完整的规则可参阅 Sacks et al.(1974)。

话轮转换规则（Turn-taking Rule）　每个话轮的每个 TRP：　（19.1）

a. 如果在该话轮中，目前的说话人已经选择 A 为下一个说话人，则下一个讲话的必须是 A。
b. 如果目前的说话人没有选择下一个说话人，其他说话人可以在下一轮说话。
c. 如果没有其他人参加下一个话轮，目前的说话人可以接着参加下一个话轮。

规则（19.1）蕴涵对话模型的许多重要结论。首先，子规则（19.1a）暗示通过一些话段，说话人特意选定了下一个说话人。最明显的是问句，说话人通过问句选择另一个说话人来回答该问题。类似问答（QUESTION-ANSWER）这样的两部分结构称为毗邻对（adjacency pair）(Schegloff, 1968)；其他毗邻对包括："问候"接"问候"（GREETING followed by GREETING），"称赞"接"自谦"（COMPLIMENT followed by DOWNPLAYER），"请求"接"准许"（REQUEST followed by GRANT）等。这些毗邻对和由它们建立的对话预期在对话模型中扮演着举足轻重的角色。

子规则（19.1a）也对沉默（silence）的解释给出了暗示。尽管在任何话轮之后都可能出现沉默，但是那些紧跟着毗邻对第一部分而出现的沉默是有意义的沉默（significant silence）。例如，Levinson(1983)记录了下面来自 Atkinson and Drew(1979)的例子，在括号中（以秒计算）给出停顿的时间：

A: Is there something bothering you or not?　（19.2）
(1.0)
A: Yes or no?
(1.5)
A: Eh?
B: No.

因为 A 刚才已经问了 B 一个问题，这时的沉默可以理解为拒绝，或者可能是不喜欢回应（例如，对一个指责的请求说"不"）。相反地，在其他地方的沉默，例如说话人结束一个话轮之后的停顿，通常就不能这样理解。这些事实与口语对话系统的用户界面设计有关；由于语音识别器的速度慢而导致用户受到了对话系统中这些停顿的干扰(Yankelovich et al., 1995)。

子规则（19.1）的另一个暗示是：说话人之间转换发生的地点不是任意的。合适转换位置通常出现在话段的边界。这就带来了口语对话和文本独白（当然话语可以是书面的，独白也是可以说的；但是最新的对话应用涉及口语）的如下差别：口语的话段相当于书面文本的句子。回忆第9章中讲过的，话段与书面句子有许多方面的差别。话段常常较短，更可能是单一从句，主语常常是代词而不是完全的词典名词短语，并且它们中充斥着停顿、修正和重说等。

一个很重要但在第9章中没有提及的差别是：尽管书面句子和段落之间相对来说比较容易自动切分，但是话段和话轮的切分非常复杂。话段的边界识别很重要，因为许多计算对话模型都是基于所抽取的话段作为一个基本单元的。切分问题很困难，因为一个单一的话段可能被扩展为几个话轮，或者一个单一的话轮可能包括几个话段。例如，在下面的旅行代理人和客户之间的对话片断中，旅行代理人的话段展开为三个以上的话轮：

A: Yeah yeah the um let me see here we've got you on American flight time thirty eight. （19.3）
C: Yep.
A: leaving on the twentieth of June out of Orange County John Wayne Airport at seven thirty p.m.
C: Seven thirty.
A: and into uh San Francisco at eight fifty seven.

相反，在下面的例子中，三个话段在一个话轮里：

A: Three two three and seven five on. OK and then does he know there is a nonstop that goes from Dulles to San Francisco? Instead of connection through St. Louis. （19.4）

话段切分的算法是基于许多边界线索的，例如：

- **线索词** 线索词，例如 *well*，*and* 和 *so* 等，倾向于出现在话段的首尾处(Reichman, 1985; Hirschberg and Litman, 1993)。
- ***N*元词或词性序列** 特定单词或词性序列常常预示着边界。标注了特定话段边界标记的训练集可用于训练*N*元语法，然后通过解码算法给标注的测试集找到最可能的话段边界(Mast et al., 1996; Meteer and Iyer, 1996; Stolcke and Shriberg, 1996a; Heeman and Allen, 1999)。
- **韵律** 正如第4章的介绍，尽管话段和韵律单位，例如语调单位(Du Bois et al., 1983)或语调短语(Pierrehumbert, 1980; Beckman and Pierehumbert, 1986)之间的关系很复杂(Ladd, 1996; Ford and Thompson, 1996; Ford et al., 1996)，韵律的特征，例如音高、重音、短语最后延长和停顿音延，在话段话轮的切分中扮演着重要角色。

在人机对话中，话轮和话段之间的关系出现一对一的情形似乎多于上面讨论的人-人对话。这可能是因为当前系统的简单性导致人们采用了较简单的话段和话轮。因此，尽管像会议摘要这样的任务需要解决非常困难的切分问题，但是在会话智能代理中还算比较容易。

19.1.2 对话的共同基础

对话区别于独白的另一个重要特征是：它是说话人和听话人共同的行为。这意味着，与独白不同，说话人和听话人必须不断地建立共同基础（common ground）(Stalnaker, 1978)，即对话双方都认可的事物的集合。需要获得共同基础，意味着听话人必须依靠（ground）或确认（acknowledge）说话人的话段，或者阐明达到共同基础还存在问题。例如，研究单词 *mm-hmm* 在下面的旅行代理人和客户的对话片断中的角色。

A: ...returning on U.S. flight one one one eight.

C: Mm hmm

其中，单词 *mm-hmm* 是一个接续（continuer），也常常称为反输（backchannel）或确认标记（acknowledgement token）。接续是一个短的话段，它确认前面话段的某些方面，常常提示其他说话人来继续该对话(Jefferson, 1984; Schegloff, 1982; Yngve, 1970)。通过让说话人知道该话段已经“到达”听话人，则接续/反输帮助说话人和听话人获得共同基础。接续只是听话人表明自己确信已理解了说话人的用意的许多方法中的一种。Clark and Schaefer(1989)讨论了 5 种主要类型的方法，按从弱到强的顺序排列：

1. **继续关注**（continued attention）　B 表明她准备继续关注，因此对 A 的陈述保持满意。
2. **相关邻接贡献**（relevant next contribution）　B 开始邻接相关的贡献。
3. **确认**（acknowledgement）　B 点头或说接续，如 *uh-huh*，*yeah*，或给出一个评价，如很好（*that's great*）。
4. **表明**（demonstration）　B 表明她已经理解 A 的所有或部分陈述，例如释义或重组 A 的话段，或协作地完成 A 的话段。
5. **展示**（display）　B 逐字展示 A 的所有或部分陈述。

从会话样例摘录的如下例子说明：通过对 A 的 *on the 11th* 的重述展示了理解。

C_6: OK I'll take the 5ish flight on the night before on the 11th.

A_6: On the 11th?

这种重述或重组常常出现于 A_6 问题形式中，我们将在 19.3 节再讨论这个问题。

对基于会话智能代理的电话系统来说，并不是所有的 Clark 和 Schaefer 的方法都可以使用。例如，如果没有出现凝视（关注的视觉表征），那么继续关注就不是一种选择。实际上 Stifelman et al.(1993)和 Yankelovich et al.(1995)指出，在处理基于语音界面的用户话段之后，如果系统没能给用户一个明显的确认信号，用户常常会感到困惑。

除了这些确认行为，听话人可以表明对前面话段的理解存在问题，例如发出一个修复请求（request for repair），如下面的 Switchboard 例子所示：

A: Why is that?

B: Huh?

A: Why is that?

19.1.3　会话隐涵

会话的最后一个重要特性是对话段的理解不仅仅基于句子的字面意义。研究上面例子中客户的反应 C_2，重述如下：

A_1: And, what day in May did you want to travel?

C_2: OK uh I need to be there for a meeting that's from the 12th to the 15th.

注意，客户实际上并没有回答该问题。客户仅仅说在那段时间他要参加一个会议。通过语义解释器生成的该句的语义只会简单地提及这次会议。旅行代理人怎样才能推理出客户提及这次会议是为了告知旅行的时间？

再看看样例会话中的另一个话段，旅行代理人说：

A_4: ...There's three non-stops today.

即使今天有7个直飞航班，这个陈述仍然可以为真，因为即使有7个，通过界定也可能只有3个。这里，旅行代理人表示的是今天有3个并且不多于3个的直飞航班。客户如何才能推理出旅行代理人要表达的意思是只有3个（only three）直飞航班？

这两个例子都有一个共同点：在这两个例子中，说话人似乎都期望听话人能推理出某个结论。换句话说，说话人交流的信息大于话段单词表面给出的信息。Grice(1975, 1978)给出这类例子以支持他的会话隐涵（conversational implicature）理论。隐涵意味着对允许的推理需要进行特殊分类。Grice提出，听话人之所以能够推出这些结论，是因为会话都需要遵循普遍准则集，这些通用的启发式准则在会话话段的解释中起着指导作用。他提出了下面的4个普遍准则：

- **数量准则**（maxim of quantity） 与需求正好一致的信息：
  1. 贡献与需求一致的信息（满足当前交流的目的）。
  2. 一定不要贡献多于需求的信息。
- **质量准则**（maxim of quality） 尽可能贡献真实的信息：
  1. 不要说你认为是虚假的信息。
  2. 不要说对你而言缺乏足够证据的信息。
- **相关准则**（maxim of relevance） 贡献切题的信息。
- **方式准则**（maxim of manner） 贡献清楚的信息：
  1. 避免模糊的表达。
  2. 避免歧义。
  3. 简短（避免不必要的啰嗦）。
  4. 有序。

正是数量准则（特别是数量准则1）使听话人明白3个直飞航班与7个直飞航班不同。这是因为听话人假定说话人遵循该准则，因此如果说话人想表示7个直飞航班，她将说7个直飞航班（“与需求一致的信息”）。相关准则使旅行代理人明白客户想在12日旅行。旅行代理人假定客户遵循该准则，因此如果在对话的这一点上会议是切题的，则客户只需提及该会议。而使该会议相关的最自然的推理就是客户想让旅行代理人明白他的出发时间早于会议时间。

会话的这三个特性（话轮转换、共同基础和隐涵）在下面几节关于对话行为、对话结构和对话管理的讨论中将发挥重要的作用。

19.2 对话行为

Austin(1962)洞察到会话的一个重要特征，即对话中的话段是一种由说话人实施的行为。这在类似下面的施行句（performative sentence）中体现得尤为突出：

I name this ship the *Titanic*.（我把这艘船命名为Titanic。） (19.5)

I second that motion.（我赞成该运动。） (19.6)

I bet you five dollars it will snow tomorrow.（我与你赌5美元，明天将下雪。） (19.7)

例如，当由一个高尚的权威人士说出例句（19.5）时，与其他任何能够改变世界的行为一样，它具有改变世界状态的影响（导致这艘船具有 Titanic 的名字）。像 *name* 或 *second* 这样的能够实施这些行为的动词被称为施行动词(performative verb), Austin把这类行为称做言语行为(speech act)。导致Austin的研究有广泛影响的原因是言语行为并不仅仅局限于这一小类施行动词。Austin认为真实言语情形中发出的任何句子不外乎这三类行为：

- **发话言行**(locutionary act)　发出一个带有特殊意义的句子。
- **示意言行**(illocutionary act)　在发出一个句子时带有询问、回答、承诺等行为。
- **取效言行**（ perlocutionary act ）　发出的一个句子对听话人的感情、信念或行为产生一种特定效果（常常是有意地）。

例如，Austin解释发出类似例句（19.8）这样的句子具有抗议的示意语力（ illocutionary force ），并且具有使听话人停止做某事或惹恼听话人的取效影响。

You can't do that.　　（19.8）

术语“言语行为”常常用于描述示意言行而不是其他两个层次。Searle(1975b)对 Austin 的分类进行了修改，提出将所有言语行为划分为 5 类：

- **断言语**(assertives)　说话人对某事是某种情形的表态（建议、提出、宣誓、自夸、推断等）。
- **指令语**（ directives ）　说话人的目的是使听话人做某事（询问、命令、要求、邀请、建议、乞求等）。
- **承诺语**(commissives)　说话人对将来的行为做出承诺（承诺、计划、发誓、打赌、反对等）。
- **表情语**（ expressives ）　表达说话人对一些事情的心理状态（感谢、道歉、欢迎、悲痛等）。
- **宣告语**（ declarations ）　由说话人说出而使外在世界产生新情景（包括上面的许多施行例句；我辞职，你被开除了等）。

尽管言语行为给出了其中一类语用语力（ pragmatic force ）的有用特征，但是最近的研究工作，特别是对话系统的建立，已经在很大程度上扩展了这个核心概念，它们是建立在话段所能扮演的更多类型的会话功能之上的。这种被丰富化了的行为称为对话行为（ dialogue act ）(Bunt, 1994)或会话行动（ conversational moves ）(Power, 1979; Carletta et al., 1997)。最近正在努力实施的一个对话行为标注方案称为“对话系统多层标记语言”（ Dialogue Act Markup in Several Layers，简称 DAMSL ）体系(Allen and Core, 1997; Walker et al., 1996; Carletta et al., 1997; Core et al., 1999)，其中编录了话段的各种层次的对话信息。其中的两层，向前功能（ forward looking function ）和向后功能（ backward looking function ），是对言语行为的扩展，扩展了像前面提及的毗邻对这样的对话结构概念以及对话共同基础和修复的概念。例如，话段的向前功能对应于 Searle/Austin 言语行为的概念，尽管 DAMSL标记集是有层级的，并且集中于对面向任务的对话中容易发生的那些类型的对话行为的处理：

STATEMENT	a claim made by the speaker （说话人的声明）
INFO-REQUEST	a question by the speaker （说话人的提问）
CHECK	a question for confirming information (see below) （确认信息的提问）
INFLUENCE-ON-ADDRESSEE	(=Searle's directives) （ = Searle 的指令语）

OPEN-OPTION	a weak suggestion or listing of options （不充分的建议或列出选项）
ACTION-DIRECTIVE	an actual command （行动的命令）
INFLUENCE-ON-SPEAKER	(=Austin's commissives) （ = Austin 的承诺语）
OFFER	speaker offers to do something, (subject to confirmation) （说话人提议做某事，易被确认）
COMMIT	speaker is committed to doing something （说话人全力以赴做某事）
CONVENTIONAL	other （其他）
OPENING	greetings （欢迎）
CLOSING	farewells （再见）
THANKING	thanking and responding to thanks （致谢和对致谢进行回应）

DAMSL的向后功能集中于表示由其他说话人发出的前述话段的话段关系，包括接受或拒绝的建议(因为DAMSL集中于对面向任务的对话的研究),以及前面提及的对话的共同基础和修复行为。

AGREEMENT	speaker's response to previous proposal （说话人对前面的提议的回应）
ACCEPT	accepting the proposal （接受提议）
ACCEPT-PART	accepting some part of the proposal （接受部分提议）
MAYBE	neither accepting nor rejecting the proposal （既不接受也不反对提议）
REJECT-PART	rejecting some part of the proposal （反对部分提议）
REJECT	rejecting the proposal （反对提议）
HOLD	putting off response , usually via subdialogue （推迟反应，常常通过子对话给出）
ANSWER	answering a question （回答提问）
UNDERSTANDING	whether speaker understood previous （说话人是否理解前面的部分）
SIGNAL-NON-UNDER.	speaker didn't understand （说话人不理解）
SIGNAL-UNDER.	speaker did understand （说话人已理解）
ACK	demonstrated via continuer or assessment （通过接续或评价表明）
REPEAT-REPHRASE	demonstrated via repetition or reformulation （通过重复或重组表明）
COMPLETION	demonstrated via collaborative completion （通过协同完成表明 ）

图 19.2 给出的是采用 DAMSL 向前和向后标记标注后的样例会话。

[assert]	C_1:	. . . I need to travel in May.
[info-req,ack]	A_1:	And, what day in May did you want to travel?
[assert, answer]	C_2:	OK uh I need to be there for a meeting that's from the 12th to the 15th.
[info-req,ack]	A_2:	And you're flying into what city?
[assert,answer]	C_3:	Seattle.
[info-req,ack]	A_3:	And what time would you like to leave Pittsburgh?
[check,hold]	C_4:	Uh hmm I don't think there's many options for non-stop.
[accept,ack]	A_4:	Right.
[assert]		There's three non-stops today.
[info-req]	C_5:	What are they?
[assert, open-option]	A_5:	The first one departs PGH at 10:00am arrives Seattle at 12:05 their time. The second flight departs PGH at 5:55pm, arrives Seattle at 8pm. And the last flight departs PGH at 8:15pm arrives Seattle at 10:28pm.
[accept,ack]	C_6:	OK I'll take the 5ish flight on the night before on the 11th.
[check,ack]	A_6:	On the 11th?
[assert,ack]		OK. Departing at 5:55pm arrives Seattle at 8pm, U.S. Air flight 115.
[ack]	C_7:	OK.

图 19.2　图 19.1 中样例会话的可能的 DAMSL 标注

19.3　对话行为的自动解释

上一节介绍了话段能够实施的对话行为以及其他行为。本节将转向对这些行为的识别或解释。也就是说，如何确定一个给定的输入是QUESTION（疑问），STATEMENT（陈述），SUGGEST（建议，指令语）或 ACKNOWLEDGEMENT（承认）？

乍一看，这个问题似乎很简单。我们从第 9 章知道，英语中的 yes-no 疑问具有助词倒置（aux-inversion），陈述具有直陈式句式（没有助词倒置），命令具有祈使句式（没有句法上的主语）的特点，如例句（19.9）所示：

YES-NO-QUESTION	Will breakfast be served on USAir 1557?	(19.9)
STATEMENT	I don't care about lunch	
COMMAND	Show me flights from Milwaukee to Orlando on Thursday night.	

从例句（19.9）来看，似乎输入的表面句法就应该可以告诉我们它是示意言行。遗憾的是，从本章开头 Abbott 和 Costello 的著名的 *Who's on First* 也可以看出，事实往往不那么简单。表面句法形式和示意言行之间的对应关系并不明显，甚至不是一对一的。

例如，下面客户对 ATIS 系统说的话看起来像一个 yes-no 疑问句，表示类似 *Are you capable of giving me a list of ...*?的意思：

Can you give me a list of the flights from Atlanta to Boston? (19.10)

然而，事实上这个客户感兴趣的并不是该系统能否给出这样一个列表（list），而是一个指令语或 REQUEST（要求）的礼貌表达，表示的意思更接近于 *Please give me a list of ...*因此，表面上看起来像一个 QUESTION，而实际上是一个 REQUEST。

类似的情况还有表面上看起来像一个 STATEMENT，而实际上是一个 QUESTION。有种常见的疑问句，称为 CHECK（核对）疑问句(Carletta et al., 1997; Labov and Fanshel, 1977)，用于要求另一个说话人确认某些只有他知道的事情。这种 CHECK 是疑问句，但具有指令句的表面形式，例如下面从旅行代理人会话中摘录的用黑体表示的话段：

A OPEN-OPTION	I was wanting to make some arrangements for a trip that I'm going to be taking uh to LA uh beginning of the week after next.
B HOLD	OK uh let me pull up your profile and I'll be right with you here.[pause]
B CHECK	**And you said you wanted to travel next week?**
A ACCEPT	Uh yes.

采用陈述句形式提出疑问，或采用疑问形式发出请求，都被称为间接的言语行为（indirect speech act）。怎样才能使 *Can you give me a list of the flights from Atlanta to Boston?* 这样的形式上的 yes-no 疑问句对应于正确的示意言行 REQUEST？解决该问题的方法是习语的连续集。该连续集的一端是习语方法，它假定像 *Can you give me a list?*或 *Can you pass the salt?*这样的句子具有歧义，一个是作为 YES-NO-QUESTION 的字面意义，另一个是惯用的作为请求的意义。英语的语法可以简单地把 REQUEST 作为 *Can you X* 的一个意义。这种方法存在的一个问题是，实施一个间接请求的方式可能有很多种，而每个方法的表面语法结构可能会有细微的差别。这种语法就不得不把 REQUEST 的意义添加给许多不同的表达。而且，习语方法没有利用这样一个事实：存在确定间接请求合法的语义概括性。

该连续集的另一端是推理方法，由 Gordon and Lakoff(1971)首次提出，由 Searle(1975a)继承发展。像 *Can you give me a list of flights from Atlanta?*这样的句子是没有歧义的，表示的是 *Do you have the ability to give me a list of flights from Atlanta?*的意思。指令的言语行为 *Please give me a list of flights from Atlanta* 是由听话人推理得出的。

在下面的两个小节中将介绍对话行为解释的两种模型：一种是称为计划推理（plan inference）的推理模型，另一种是称为提示模型（cue model）的基于习语的模型。

19.3.1 对话行为的计划推理解释

对话行为解释的计划推理方法是由 Gordon and Lakoff(1971)以及 Searle(1975a)首次提出的。这些文献指出，存在一种对应于事件类型的结构，说话人通过这些事件类型可以发出间接请求。特别是，说话人可以提及或询问所期望行为的各种非常特定的属性，以发出间接请求，这里是从 ATIS 语料库摘录的部分例子：

1. 说话人可以对听话人实施该行为的能力提出疑问：
   - Can you give me a list of the flights from Atlanta to Boston?
   - Could you tell me if Delta has a hub in Boston?
   - Would you be able to, uh, put me on a flight with Delta?

2. 说话人可以提及说话人对该行为的希望或期望：
   - I want to fly from Boston to San Francisco.
   - I would like to stop somewhere else in between.
   - I'm looking for one way flights from Tampa to Saint Louis.
   - I need that for Tuesday.
   - I wonder if there are any flights from Boston to Dallas.
3. 说话人可以提及听话人来做该行为：
   - Would you please repeat that information?
   - Will you tell me the departure time and arrival time on this American flight?
4. 说话人可以对其是否被容许接受该行为的结果提出疑问：
   - May I get a lunch on flight U A two one instead of breakfast?
   - Could I have a listing of flights leaving Boston?

基于这个认识，Searle(1975a，p.73)提出了听话人在听到句子*Can you give me a list of the flights from Atlanta to Boston*?时的推理链，它类似于下面的描述（为 ATIS 例子做了修改）：

1. X 问我一个问题，是关于我是否具有给出航班列表的能力的问题。
2. 我认为在会话中 X 是协作的，因此他的话段具有一定的目的。
3. X 知道我具备给出这样一个列表的能力，并不存在 X 对我的列表能力产生纯理论的兴趣的其他原因。
4. 因此 X 的话段可能有一些隐藏的示意点。它可能是什么呢?
5. 指令语的准备条件是听话人具备实施该行为的能力。
6. X 问我的问题是：我对给他一个航班列表的行为是否有准备。
7. 而且，X 和我在会话的情形下，给出航班列表是一个普通和预期的行为。
8. 因此，在缺少任何其他合理的示意言行时，X 可能是请求我给他一个航班列表。

这种推理方法具有许多优点。首先，它解释了为什么*Can you give me a list of flights from Boston*?是给出间接请求的合理方式，而 *Boston is in New England* 不是。前者提及期望行为的一个先决条件，并且从该先决条件到该期望的行为本身有一个合理的推理链。关于推理方法的建模问题，Allen，Cohen和Perrault以及他们的同事发表了许多有影响的论文，所提模型被称为BDI（belief, desire, and intention）模型(Allen, 1995)。早期的论文，例如 Cohen and Perrault(1979)，给出了言语行为如何生成的 AI 计划模型。一个试图发现信息的行为人，能够利用标准的计划技术找到让听话人告诉说话人该信息的计划。Perrault and Allen(1980)以及 Allen and Perrault(1980)也将 BDI 方法应用于理解，特别是间接言语语效的理解，本质上是对 Searle(1975)的书面记录给出了可计算的形式体系。

我们以谓词演算中 Perrault 和 Allen 对信念（belief）和期望（desire）的形式化定义作为开始。把“*S* 相信命题 *P*”表示为二元谓词 $B(S, P)$。关于信念的推理是通过许多受 Hintikka(1969)启发的公理方案来实现的，例如 $B(A,P) \wedge B(A,Q) \Rightarrow B(A,P \wedge Q)$;，详见 Perrault and Allen(1980)。知识（knowledge）被定义为“真的信念”（true belief）；*S knows that P* 被表示为 $KNOW(S, P)$，定义如下：

$$\text{KNOW}(S,P) \equiv P \wedge B(S,P)$$

除了 *knowing that*，还需要定义 *knowing whether*。如果 *S* KNOWs that *P* 或 *S* KNOWs that $\neg P$，那么 *S* knows whether (KNOWIF)命题 *P* 为真：

$$\text{KNOWIF}(S,P) \equiv \text{KNOW}(S,P) \vee \text{KNOW}(S,\neg P)$$

期望（desire）的理论基于谓词 WANT。如果行为人 S 希望 P 为真，我们用 $WANT(S, P)$或 $W(S, P)$表示。P 可以是一些行为的状态或实施，因此如果ACT是一个行为的名称，则 $W(S, \text{ACT}(H))$ 表示 S 想让 H 来做ACT。与信念（belief）的逻辑一样，WANT的逻辑是基于它本身的公理方案的。

BDI模型也需要行为和计划的公理化；最简单的是基于行为方案（action schema）集，类似于AI计划模型 STRIPS(Fikes and Nilsson, 1971)。每个行为方案都有一个参数集，用于约束每个变量的类型和下面的三个部分：

- **前提**（precondition） 为成功实施该行为必须已经为真的那些条件。
- **效果**（effect） 作为成功实施该行为结果的变为真的那些条件。
- **体**（body） 在实施该行为中必须达到的部分有序的目标集。

例如，在旅行领域，行为人 A 为客户 C 预订航班（booking flight）$F1$ 的行为，可以简化为下面的定义：

BOOK-FLIGHT(A,C,F):

Constraints: Agent(A) ∧ Flight(F) ∧ Client(C)

Precondition: Know(A,departure-date(F)) ∧ Know(A,departure-time(F)) ∧ Know(A,origin-city(F)) ∧ Know(A,destination-city(F)) ∧ Know(A,flight-type(F)) ∧ Has-Seats(F) ∧ W(C,(BOOK(A,C,F))) ∧ ...

Effect: Flight-Booked(A,C,F)

Body: Make-Reservation(A,F,C)

Cohen and Perrault(1979)和Perrault and Allen(1980)对言语行为采用了这种类型的详细说明。例如，下面是 Perrault 和 Allen 对三种与间接请求有关的言语行为所做的定义。INFORM 是通知听话人一些命题（Austin/Searle 断言语，或 DAMSL STATEMENT）的言语行为。INFORM 的定义基于 Grice(1957)的思想：说话人通知（inform）听话人某些事情，是通过让听话人相信说话人想让他们知道（know）一些事情来实现的：

INFORM(S,H,P):

Constraints: Speaker(S) ∧ Hearer(H) ∧ Proposition(P)

Precondition: Know(S,P) ∧ W(S, INFORM(S, H, P))

Effect: Know(H,P)

Body: B(H,W(S,Know(H,P)))

INFORMIF 是用于通知听话人命题是否为真的行为；与 INFORM 类似，说话人 INFORMIF 听话人，是通过让听话人相信说话人想让他们 KNOWIF 一些事情来实现的：

INFORMIF(S,H,P):

Constraints: Speaker(S) ∧ Hearer(H) ∧ Proposition(P)

Precondition: KnowIf(S, P) ∧ W(S, INFORMIF(S, H, P))

Effect: KnowIf(H, P)

Body: B(H, W(S, KnowIf(H, P)))

REQUEST 是请求听话人实施某些行为的指令言语行为：

REQUEST(S,H,ACT):

Constraints: Speaker(S) ∧ Hearer(H) ∧ ACT(A) ∧ H is agent of ACT

Precondition: W(S,ACT(H))

Effect:　　W(H,ACT(H))
Body:　　B(H,W(S,ACT(H)))

Perrault 和 Allen 的理论也需要被称为“表面层的行为”（surface-level act）。这对应于命令式、疑问式和陈述式结构的“字面意义”。例如“表面层”行为 S.REQUEST 生成命令式的话段：

S.REQUEST (S, H, ACT):
Effect: B(H, W(S,ACT(H)))

S.REQUEST 的效果（effect）与常规的 REQUEST 的体（body）一致，因为这是发出请求的默认或标准方式（但不是惟一的方式）。这个“默认”或“字面”的意义是听话人推理链的起点。一个输入将传给听话人，表示说话人请求听话人通知说话人，听话人是否能够给说话人一个列表：

S.REQUEST(S,H,InformIf(H,S,CanDo(H,Give(H,S,LIST))))

听话人必须领会到说话人实际上是发出一个请求：

REQUEST(H,S,Give(H,S,LIST))

从 request-to-inform-if-cando 到 request-to-give 的推理链基于似乎合理的推理（plausible inference）链，基于被称为计划推理（plan inference，简称 PI）规则的启发式规则。我们将采用由 Perrault and Allen(1980)提出的如下规则的子集：

- **(PI.AE)行为 – 效果规则（Action-Effect Rule）**　对所有行为人 S 和 H，如果 Y 是行为 X 的效果，并且如果 H 相信 S 想实施 X，则 H 相信 S 想获得 Y 是合理的。
- **(PI.PA)前提 – 行为规则（Precondition-Action Rule）**　对所有行为人 S 和 H，如果 X 是行为 Y 的前提，并且如果 H 相信 S 想获得 X，则 H 相信 S 想实施 Y 是合理的。
- **(PI.BA)体 – 行为规则（Body-Action Rule）**　对所有行为人 S 和 H，如果 X 是 Y 的体的一部分，并且如果 H 相信 S 想实施 X，则 H 相信 S 想实施 Y 是合理的。
- **(PI.KP)知道–期望规则（Know-Desire Rule）**　对所有行为人S和H，如果H相信S想KNOWIF(P)，则 H 相信 S 想让 P 为真。

$$B(H,W(S,\text{KNOWIF}(S,P))) \overset{\text{plausible}}{\Longrightarrow} B(H,W(S,P))$$

- **(EI.1)扩展推理规则（Extended Inference Rule）**　如果$B(H,W(S,X)) \overset{\text{plausible}}{\Longrightarrow} B(H,W(S,Y))$是一个 PI 规则，则：

$$B(H,W(S,B(H,(W(S,X))))) \overset{\text{plausible}}{\Longrightarrow} B(H,W(S,B(H,W(S,Y))))$$

也是一个 PI 规则，也就是说可以把 $B(H, W(S))$放置于任何计划推理规则之前。

我们来看看如何应用这些规则来解释例句 *Can you give me a list of flights from Atlanta*?的间接言语行为。下表中 Step 0 的位置给出了说话人最初的言语行为，听话人最初把它解释为一个提问。然后，Step 1 利用计划推理的行为 – 效果规则，表示如果说话人请求某件事（本例为信息），则可能想获得它。Step 2 再次利用行为 – 效果规则，表示如果说话人想要 INFORMIF，并且 KNOWIF 是 INFORMIF 的效果，则说话人可能也想要 KNOWIF。

Step 3 添加了一个重要的推理：人们通常不询问他们不感兴趣的事，因此如果说话人询问某件事（本例为 CanDo）是否为真，则说话人可能想让这件事（CanDo）为真。Step 4 利用事实 CanDo(ACT)是(ACT)的前提，做出了这样的推理：如果说话人想获得一个行为（Give）的前提（CanDo），

则说话人也可能想要该行为（Give）。最后，Step 5基于REQUEST的定义来表示如果说话人想让某人知道，说话人想让他们做某事，则说话人可能REQUEST他们做某事。

Rule	Step	Result
	0	S.REQUEST(S, H, InformIf(H, S, CanDo(H, Give(H, S, LIST))))
PI.AE	1	B(H, W(S, InformIf(H, S, CanDo(H, Give(H, S, LIST)))))
PI.AE/EI	2	B(H, W(S, KnowIf(H, S, CanDo(H, Give(H, S, LIST)))))
PI.AE/EI	3	B(H, W(S, CanDo(H, Give(H, S, LIST))))
PI.PA/EI	4	B(H, W(S, Give(H, S, LIST)))
PI.BA	5	REQUEST(H, S, Give(H, S, LIST))

在给出间接言语理解的计划推理方法的这个概要时，我们忽略了许多细节，包括许多必须的公理，以及确定采用哪个推理规则的机制等。有兴趣的读者可参阅 Perrault and Allen(1980)和本章结尾处提及的其他文献。

19.3.2 对话行为的基于提示的解释

对话行为理解的计划推理方法是特别强有力的；通过采用丰富的知识结构和强大的计划技术，该算法甚至被用于对话行为的微妙的间接应用。计划推理方法的缺点是非常耗时，无论是从计划推理的启发式规则所花费的人工成本来看，还是从这些启发式规则在系统中运行的时间来看，都是如此。实际上，由于容许所有可能类型的非语言推理在对话处理中发挥作用，该方法的完整应用是一个AI完全问题。AI完全问题是指那些在没有解决生成完全的人工智能的整个问题之前，不可能真正解决的问题。

因此，对于许多应用来说，较简单但更有效的数据驱动方法可能就足够满足需要了。这类方法中的一种是前述的习语方法的变种。我们知道，在习语方法中，像 *Can you give me a list of flights from Atlanta*?这样的句子具有两种字面意义，一种是提问而另一种是请求。这可以通过对像Can you X这样的句子结构添加两种意义而在语法中加以实现。本节讲述的对话行为理解的基于提示的方法就基于习语的这个直觉。

在对话行为的理解中，许多研究者都采用了基于提示的方法，尽管可能使用了其他名称。基于提示的模型之间的不同在于辨别对话行为时所采用的知识（提示）源的不同，例如词汇、搭配、句法、韵律或对话结构等提示源。我们将描述的方法采用的是（监督）机器学习的算法，利用对每个话段都添加了手工标注的对话行为的语料加以训练。采用哪些提示则取决于每个系统。许多系统都基于这样一个事实：单个的对话行为常常具有Goodwin(1996)所称的微语法（microgrammar），特定的词汇、搭配和韵律特征是它们的特点。这些系统也依赖于会话结构。例如，Jurafsky et al.(1997)的对话行为解释系统就依赖于下面的三个信息源：

1. **单词和搭配（Words and Collocation）** *Please* 或 *would you* 是REQUEST（请求）的有效提示，而 *are you* 是 YES-NO-QUESTION 的有效提示。
2. **韵律（Prosody）** 上升音高是 YES-NO-QUESTION 的有效提示。响度或重音可以帮助区分 *yeah* 是 AGREEMENT（同意）还是 BACKCHANNEL（反输）。
3. **会话结构（Conversational Structure）** 建议之后的*yeah*可能是AGREEMENT，而INFORM（通知）之后的 *yeah* 可能是 BACKCHANNEL。

上一节重点讨论了如何通过基于计划的方法得出一个表面的提问具有REQUEST（请求）的示意语力。本节将研究一种不同类型的间接请求：CHECK（核对），研究 Jurafsky et al.(1997)的系统为解决这个对话行为的识别问题而采用的特别提示。回想一下，CHECK 是请求对话者确认某些信息的提问的子类型；这些信息既可能是在前面对话中明显提及的信息（参见下面的例子），也可能是从对话者所说的内容推理出来的信息：

A OPEN-OPTION	I was wanting to make some arrangements for a trip that I'm going to be taking uh to LA uh beginning of the week after next.
B HOLD	OK uh let me pull up your profile and I'll be right with you here. [pause]
B CHECK	**And you said you wanted to travel next week?**
A ACCEPT	Uh yes.

英语中 CHECK 的可能的实际例子包括：

1. 作为附加疑问句：

取自训练语料(Allen and Core, 1997)　　（19.11）

U **and it's gonna take us also an hour to load boxcars right?**

S right

2. 作为陈述疑问句，通常采用升调(Quirk et al., 1985，p.814)

取自 Switchboard 语料(Godfrey et al., 1992)　　（19.12）

A and we have a powerful computer down at work.

B Oh(laughter)

B **so, you don't need a personal one (laughter)?**

A No

3. 作为片断疑问句（构成句子的单元、词、名词短语、从句等）(Weber, 1993)

取自 Map Task 语料(Carletta et al., 1997)　　（19.13）

G Ehm, curve round slightly to your right.

F **To my right?**

G Yes.

如上面的例句所示，对核对（CHECK）的研究表明它们常常是陈述句结构（即没有助词倒置），它们倾向于采用升调(Shriberg et al., 1998)，并且常常带有疑问附加，如例句（19.11）中的*right*(Quirk et al., 1985，810–814)。它们也常常被实现为带有升调的“片断”（构成句子的单词或短语）。在 Switchboard 语料中，CHECK 的 REFORMULATION（重组）子类具有非常特殊的微语法：陈述句的词序，常以 *you* 为主语（31%），常以 *so*（20%）或 *oh* 为句子的开始，有时则会以 *then* 结束。一些例句如下所示：

Oh so you're from the Midwest too.

So you can steady it.

You really rough it then.

许多文献，首先是Nagata and Morimoto(1994)，都意识到这些微语法中的许多结构都可以通过为每个对话行为训练一个单独的单词-*N*元语法来简单地捕捉(Suhm and Waibel, 1994; Mast et al., 1996;

Jurafsky et al., 1997; Warnke et al., 1997; Reithinger and Klesen, 1997; Taylor et al., 1998)。这些系统通过实现相同对话行为的所有话段生成一个独立的小型语料库，然后对每个此类小型语料库进行训练，以生成独立的单词-N元语言模型。假设有一个由单词W的序列组成的话段u，则选择对话行为d，因为它的N元语法为W指派的相似性最高：

$$d^* = \underset{d}{\operatorname{argmax}} P(d|W) = \underset{d}{\operatorname{argmax}} P(d)P(W|d) \tag{19.14}$$

这个简单的N元方法确实捕捉到了许多微语法。例如，细查Switchboard语料中REFORMULATIONS的最高频率的二元对，我们发现最常见的二元组包括对REFORMULATIONS的最有效的提示，例如*so you*，*sound like*，*so you're*，*oh so*，*you mean*，*so they*和*so it's*等。

对话行为微语法的韵律模型基于音位特征，例如音高或重音，或基于第4章和第7章讨论过的它们的声学的关联，例如F0（基频）、音延和音能等。例如，许多研究已经表明对YES-NO-QUESTIONS句尾音高上升的捕捉对增强词典提示是一个十分有效的提示信息(Sag and Liberman, 1975; Pierrehumbert, 1980; Waibel, 1988; Daly and Zue, 1992; Kompe et al., 1993; Taylor et al., 1998)。Pierrehumbert(1980)也指出陈述话段（如STATEMENTS）具有最后下降（final lowering）的特点：在句尾F0的下降。借助这些研究成果，Shriberg et al.(1998)采用CART风格的决策树对简单的基于声学的韵律特征进行了训练，这些韵律特征包括在话段结尾处F0的倾斜，话段不同阶段的平均音能以及不同的音延系数等。他们发现这些特征十分有用，例如对于区分4种对话行为: STATEMENT(S), YES-NO-QUESTION(QY)以及类似CHECKS(QD)和WH-QUESTIONS(QW)的DECLARATIVE-QUESTIONS。图19.3中的决策树给出了假定声学特征F时，对话行为d的后验概率$P(d|f)$。树中每个结点给出了4个概率，按顺序S，QY，QW和QD对应于各个对话行为，以其中最可能的对话行为作为该结点的标记。通过贝叶斯规则，这个概率可以用于计算给定对话行为$P(f|d)$时声学特征的可能性。

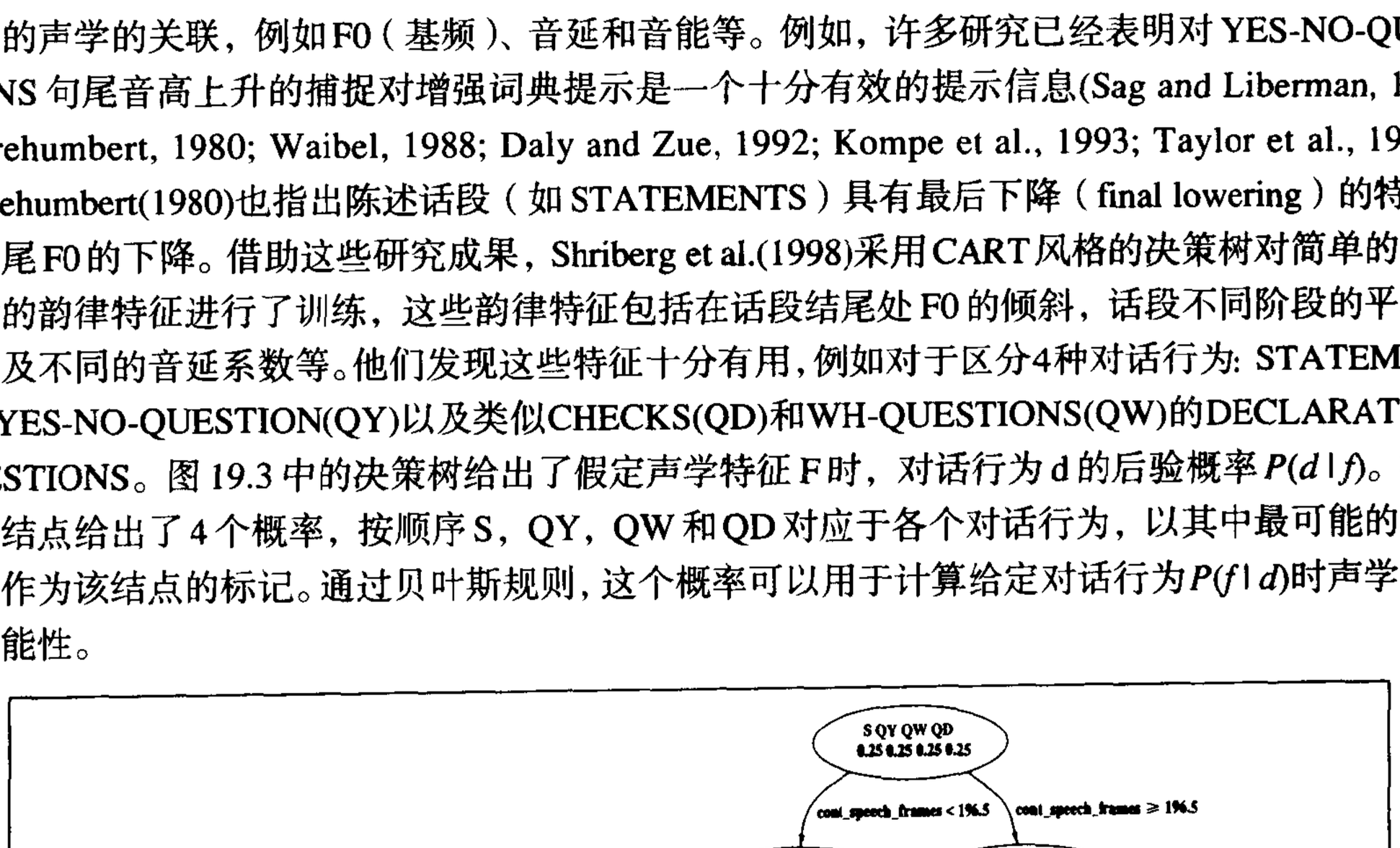


图19.3 用于STATEMENT(S)，YES-NO-QUESTIONS(QY)，WH-QUESTIONS(QW)和DECLARATIVE-QUESTIONS分类的决策树，模仿自Shriberg et al.(1998)。注意该树右边S和QY之间的差别基于特征norm_f0_diff（结尾区域和倒数第二个音节区域的平均F0归一化后的差别），而底部左边的QW和QD之间的差别基于utt_grad，表示F0在整个话段中的斜率

对话行为解释中的最后一个重要提示是会话结构。会话结构由前述的毗邻对(Schegloff, 1968; Sacks et al., 1974)的思想引申而来，对它建模的最简单方法是把它看成对话行为的概率序列。前面对话行为的识别有助于后续对话行为的预测。许多研究已经采用对话行为N元为对话行为序列建模(Nagata and Morimoto, 1994; Suhm and Waibel, 1994; Warnke et al., 1997; Chu-Carroll, 1998; Stolcke et al., 1998; Taylor et al., 1998)，常常作为对话行为的HMM系统(Reithinger et al., 1996; Kita et al., 1996; Woszczyna and Waibel, 1994)的一部分。例如，图 19.4 中所示的是 Woszczyna and Waibel(1994)给出的针对 Verbmobil-like 约会安排任务的对话隐马尔可夫模型（HMM）。

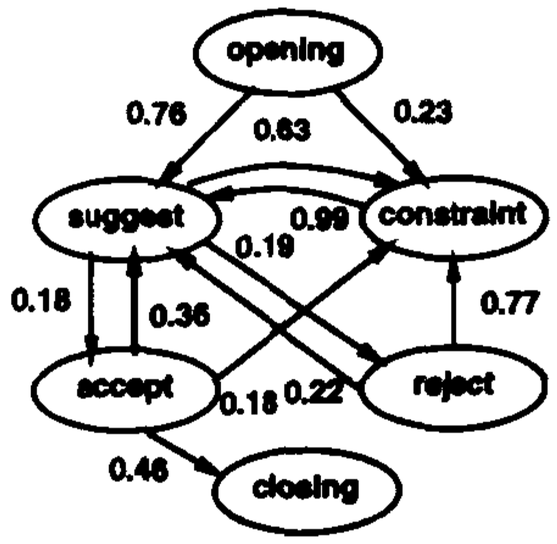


图 19.4　对话行为隐马尔可夫模型，模仿自 Woszczyna and Waibel(1994)

对话行为的解释器如何组合这些不同的提示，以找到在给定一个会话时正确对话行为的最可能正确序列？ Stolcke et al.(1998)和 Taylor et al.(1998)采用 Woszczyna and Waibel(1994)的 HMM 直觉，把对话行为发现的过程当成是HMM剖析来处理。假定E表示会话的所有可获得的证据，目的是寻找对话行为序列 $D = \{d_1, d_2, \cdots, d_N\}$，在给定证据的条件下它具有最高的后验概率 $P(D|E)$（这里用大写字母表示序列）。应用贝叶斯规则，可以得到：

$$\begin{aligned} D^* &= \operatorname*{argmax}_D P(D|E) \\ &= \operatorname*{argmax}_D \frac{P(D)P(E|D)}{P(E)} \\ &= \operatorname*{argmax}_D P(D)P(E|D) \end{aligned} \tag{19.15}$$

其中，$P(D)$表示对话行为 D 序列的先验概率。这个概率可以通过Nagata and Morimoto(1994)引入的对话行为 N 元语法来计算。似然概率 $P(E|D)$可以通过其他两个证据源，即微句法模型（例如每个对话行为的不同的单词 -N 元语法）和微韵律模型（例如每个对话行为的韵律特征的决策树）来进行计算。每个对话行为的单词 -N 元模型可以用于估计 $P(W|D)$，即单词 W 的序列的概率。微韵律模型可以用于估计 $P(F|D)$，即韵律特征 F 的序列的概率。

为了简化问题，如果假定韵律和单词相互独立（当然这是不正确的），我们就可以用下面的公式来估计一个对话 D 序列的证据似然概率：

$$P(E|D) = P(F|D)P(W|D) \tag{19.16}$$

我们可以通过将方程（19.16）带入方程（19.15）来计算对话行为 D^* 的最可能的序列，即选择三种知识源（会话结构、韵律、词汇 / 句法知识）的乘积为最大的对话行为序列：

$$D^* = \operatorname*{argmax}_D P(D)P(F|D)P(W|D)$$

则标准的HMM剖析技术（例如Viterbi）可以用于给定输入话段序列时寻找对话行为的最可能序列。

HMM方法只是解决数据驱动对话行为识别问题的一种方法。与HMM标注的结合暗示也可以使用其他方法，把对话行为看成一种标记，就可以应用其他词性标注方法。例如，Samuel et al. (1998b)将基于转换的学习用于对话行为标注。

19.3.3 要点

进行对话行为解释的两种方法（通过推理和通过提示）各有优缺点。基于提示的方法可能更适合于需要相对浅层对话结构（可以通过大规模语料训练）的系统。如果需要语义解释，则基于提示的方法还需要采用语义解释来加以扩充。完全的推理方法可能更适合需要复杂推理的情形。

19.4 对话结构与连贯性

18.2节描述了一个通过连贯关系集以确定连贯的方法。为了确定所具有的连贯关系，系统必须思考这些关系施加于话段信息（information）的约束。我们把这种观点称为连贯的信息方法（informational approach）。从历史上看，信息方法主要被应用于独白。

话段解释的BDI方法导致连贯的另一种观点，我们称之为意图方法（intentional approach）。根据这个方法，话段被理解为行为，需要听话人推测基于计划的说话人的意图，这是确立连贯的基础。与信息方法不同，意图方法主要被应用于对话。

这里所描述的意图方法归功于Grosz and Sidner(1986)，该文献认为话语可以被表示为一个由相互关联的三部分，即语言结构（linguistic structure）、关注状态（attentional state）和意图结构（intentional structure）组成的混合体。语言结构包括话语中的话段，分成话语片断的层级结构（回想第18章对话语片断的描述）。关注状态是话语在每个点具有显著性的对象、属性和关系的动态变化的模型。这与上一章介绍的话语模型的概念十分接近。中心确定（Centering）（见第18章）被认为是可以应用于该方法中的关注状态的理论。

这里将重点介绍该方法中的第三个部分，即意图结构。意图结构是基于上一节描述的对话行为解释的BDI模型。基本思想是把话语与该话语的发起人所持有的潜在目的联系在一起，这个目的被称为话语目的（discourse purpose，简称DP）。同样，话语中的每个话语片断也有一个相应的目的，被称为话语片断目的（discourse segment purpose，简称DSP）。在实现话语的DP时，每个DSP都有一个与该话语片断相对应的角色。下面所列的是Grosz和Sidner给出的一些可能的DP/DSP：

1. 某些行为人企图实施某些实际任务的意图。
2. 某些行为人相信某些事实的意图。
3. 某些行为人相信一个事实支持另一个事实的意图。
4. 某些行为人企图识别一个对象（现有物理对象、虚构对象、计划、事件和事件序列）的意图。
5. 某些行为人知道一个对象的某些属性的意图。

与在信息方法中为确立连贯而采用的大量连贯关系集相反，Grosz和Sidner只提出了两种关系：支配（dominance）和满足－优先（satisfaction-precedence）。如果满足DSP_2的目的是为DSP_1的满足提供部分基础，则称DSP_1支配DSP_2。如果DSP_1必须在DSP_2之前被满足，则称DSP_1满足－优先于DSP_2。

作为示例，我们来研究前面见到的客户C和旅行代理人A之间的对话，这里重述为图19.5。

C_1:	I need to travel in May.
A_1:	And, what day in May did you want to travel?
C_2:	OK uh I need to be there for a meeting that's from the 12th to the 15th.
A_2:	And you're flying into what city?
C_3:	Seattle.
A_3:	And what time would you like to leave Pittsburgh?
C_4:	Uh hmm I don't think there's many options for non-stop.
A_4:	Right. There's three non-stops today.
C_5:	What are they?
A_5:	The first one departs PGH at 10:00am arrives Seattle at 12:05 their time. The second flight departs PGH at 5:55pm, arrives Seattle at 8pm. And the last flight departs PGH at 8:15pm arrives Seattle at 10:28pm.
C_6:	OK I'll take the 5ish flight on the night before on the 11th.
A_6:	On the 11th? OK. Departing at 5:55pm arrives Seattle at 8pm, U.S. Air flight 115.
C_7:	OK.

图 19.5　客户 C 和旅行代理人 A 之间的电话对话片断（重述自图 19.1）

打电话者和代理人共同协作才会成功地找出打电话者所需的航班。实现这个共同目标需要满足顶层的话语意图，用下面的 I1 表示，为 I1 满足提供基础的其他几个中间意图，用 I2 至 I5 列出：

I1: (Intend C (Intend A (A find a flight for C)))
I2: (Intend A (Intend C (C Tell A departure date)))
I3: (Intend A (Intend C (C Tell A destination city)))
I4: (Intend A (Intend C (C Tell A departure time)))
I5: (Intend C (Intend A (A find a nonstop flight for C)))

意图 I2~I5 都从属于意图 I1，因为它们都是实现意图 I1 的前提。可以表示为下面的支配关系：

I1 支配 I2
I1 支配 I3
I1 支配 I4
I1 支配 I5

而且，意图 I2 和 I3 必须在意图 I5 之前被满足，因为为了列出直飞航班，代理人需要知道出发时间和目的地。这可以表示为下面的满足 – 优先关系：

I2 满足 – 优先于 I5
I3 满足 – 优先于 I5

从支配关系可以导出图 19.6 所示的话语结构。图中将每个话语片断与它所对应的意图 DP/DSP 联系起来。

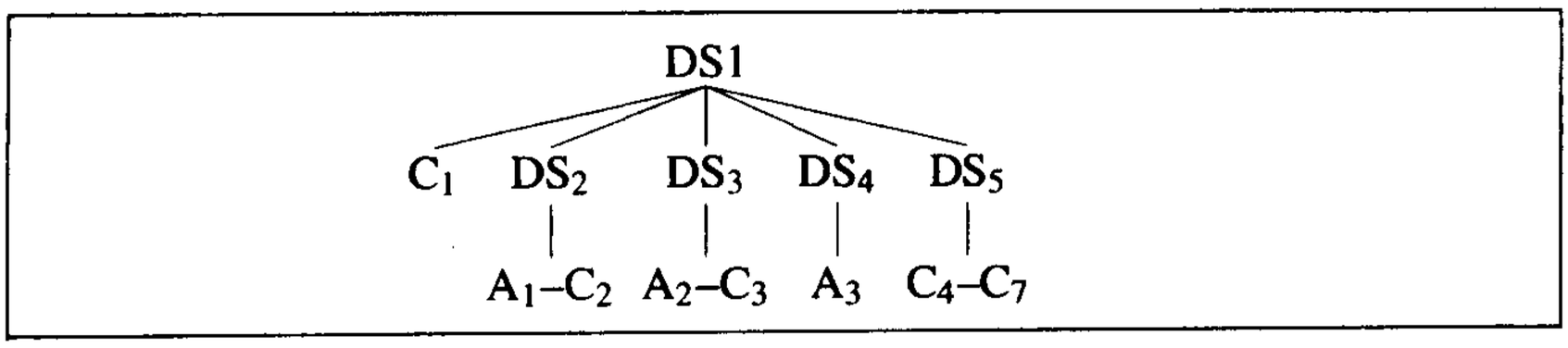


图 19.6 航班预订对话的话语结构

这个意图集和它们之间的关系基于什么才会产生连贯的话语？答案是它们在被推理出的打电话者的整个计划中的角色。该计划的表示方法有许多种，这里将采用上一节介绍过的简单STRIPS模型。我们采用了两个简单的行为方案，第一个是预订航班，重述自 19.3.1 节。

BOOK-FLIGHT(A,C,F):
Constraints: Agent(A) ∧ Flight(F) ∧ Client(C)
Precondition: Know(A,departure-date(F)) ∧ Know(A,departure-time(F)) ∧ Know(A,origin-city(F)) ∧ Know(A,destination-city(F)) ∧ Know(A,flight-type(F)) ∧ Has-Seats(F) ∧ W(C,(BOOK(A,C,F))) ∧ ...

Effect: Flight-Booked(A,C,F)
Body: Make-Reservation(A,F,C)

可以看出，预订航班需要代理人知道许多有关航班的参数信息，包括出发日期、时间、出发和目的城市等。打电话者发起的这段对话的话段包括出发城市和出发时间的部分信息。代理人不得不请求其他信息，我们采用的第二个行为方案是对该请求的简化表示[有关更深入的wh-疑问计划的讨论，参见 Cohen and Perrault(1979)]。

REQUEST-INFO(A,C,I):
Constraints: Agent(A) ∧ Client(C)
Precondition: Know(C,I)
Effect: Know(A,I)
Body: B(C,W(A,Know(A,I)))

由于REQUEST-INFO的效果（effect）匹配于BOOK-FLIGHT的每个前提（precondition），因此前者可以用于满足后者的需要。话语片断DS2和DS3是成功实施REQUEST-INFO，以分别识别出参数出发日期和目的城市的实例。片断DS4也是一个对参数值（出发时间）的请求，但是没有成功，因为打电话者没有直接回答，而是（含蓄地）问直飞航班。片断DS5通过打电话者从代理人给出的简短列表中选择直飞航班而满足顶层的DP。

像DS2和DS3这样的辅助话语片断又称为子对话（sub-dialogue）。DS2和DS3示例的这类辅助对话通常称为知识前提（knowledge precondition）子对话(Lochbaum et al., 1990; Lochbaum, 1998)，因为它们被行动者发起以帮助满足实现更高层目标的前提（在本例中指满足客户在5月旅行的请求）。它们又称为信息共享子对话（information-sharing subdialogue）(Chu-Carroll and Carberry, 1998)。

图 19.5 没有给出的会话是子对话的另一种类型：修正子对话（correction subdialogue）(Litman, 1985; Litman and Allen, 1987)或称磋商子对话（negotiation subdialogue）(Chu-Carroll and Carberry, 1998)。话段 C_{20} 到 C_{23a} 构成了对前述 5 月 15 日返回计划的修正：

A_{17}: And you said returning on May 15th?

C_{18}: Uh, yeah, at the end of the day.

A_{19}: OK. There's #two non-stops...#

C_{20}: #Act...actually#, what day of the week is the 15th?

A_{21}: It's a Friday.

C_{22}: Uh hmm. I would consider staying there an extra day til Sunday.

A_{23a}: OK....OK.

A_{23b}: On Sunday I have...

最后，子任务（subtask）子对话(Grosz, 1974)也许是子对话在文献中出现的最早类型，用于处理面向任务对话中整个任务的子任务。

确定意图结构　对话（以及口语独白）中意图结构的推理算法类似于对话行为的推理算法。许多算法应用的是 BDI 模型，例如(Litman, 1985; Grosz and Sidner, 1986; Litman and Allen, 1987; Carberry, 1990; Passonneau and Litman, 1993; Chu-Carroll and Carberry, 1998)的变体。其他算法依赖于与 19.1.1 节所描述的话段和话轮切分线索类似的提示，包括线索词和短语(Reichman, 1985; Grosz and Sidner, 1986; Hirschberg and Litman, 1993)，韵律(Grosz and Hirschberg, 1992; Hirschberg and Pierrehumbert, 1986; Hirschberg and Nakatani, 1996)以及其他提示。例如，Pierrehumbert and Hirschberg(1990)认为某些边界声调（boundary tone）也许可以用于暗示两个声调短语之间的支配关系。

信息连贯与意图连贯　正如我们刚刚看到的，意图连贯取决于对话参与者识别彼此的意图并使其适合于计划的能力。而另一方面，正如在第 18 章中所看到的，信息连贯取决于确立话段之间内容所承担的某些类型的关系的能力。因此，也许有人会问它们之间是什么关系，是彼此排斥还是可以一起为我们所用？

Moore and Pollack(1992)以及其他一些学者都认为事实上这两个层次的分析必须共存。假设例子中的代理人和打电话者在找到合适的航班之后，代理人说出下面的一段话，即例句（19.17）。

You'll want to book your reservations before the end of the day.　（19.17）

Proposition 143 goes into effect tomorrow.

这段话既可以从信息角度也可以从意图角度进行分析。从意图角度看，代理人试图说服打电话者在今天预订航班。实现这个目的的一种方式就是为该行为提供动机，这就是第二个句子所起到的作用。从信息角度看，这两个句子满足第 18 章描述的说明（explanation）关系，因为第二个句子提供了在今天预订航班的原因。

基于打电话者的知识，在信息层的识别可能导致对说话人计划的识别，反之亦然。例如，打电话者知道条款 143（Proposition 143）对机票添加了新税，但是不知道代理人说第二个句子的动机。由于激励某行为的一种方式是提供使该行为发生的原因，这个打电话者就可以推测出代理人正在推动第一个句子所描述的行为。换句话说，即使打电话者不知道条款 143 的内容，也可能从该话语情景推测出该意图。已知一个因果关系和激励某事之间的关系，可能导致打电话者假定一个说明关系，这需要他推理出该条款对机票的购买者来说是不利的（即加税）。因此，至少在某些情形下这两种层次的分析都是需要的。

19.5 会话智能代理中的对话管理

会话智能代理的思想令人神往，像ELIZA，PARRY或SHRDLU这样的会话智能代理系统已经成为自然语言技术最具知名度的实例。目前会话智能代理的应用实例包括航空旅行信息系统、基于语音的饭店向导以及电子邮件或日程表的电话界面。对话管理是这类会话智能代理的重要部件之一，它的功能包括控制对话流、在一个较高层次确定智能代理这边的会话应该如何进行，询问或陈述什么问题，以及什么时间给出这些问题。

本节将简短概括对话管理设计中的一些问题，讨论一些基于有限状态自动机（FSA）和生成规则的简单系统，以及一些基于BDI风格推理和计划技术的更复杂的系统。

最简单的对话管理系统是基于有限状态自动机的。例如，一个平凡的旅行航线系统的任务就是询问用户的出发城市、目的城市、时间和任何优先航线。图19.7给出了这样一个系统的简单对话管理。FSA的状态对应于对话管理系统询问用户的问题，弧线对应于将采取的行为，这依赖于用户的回答。

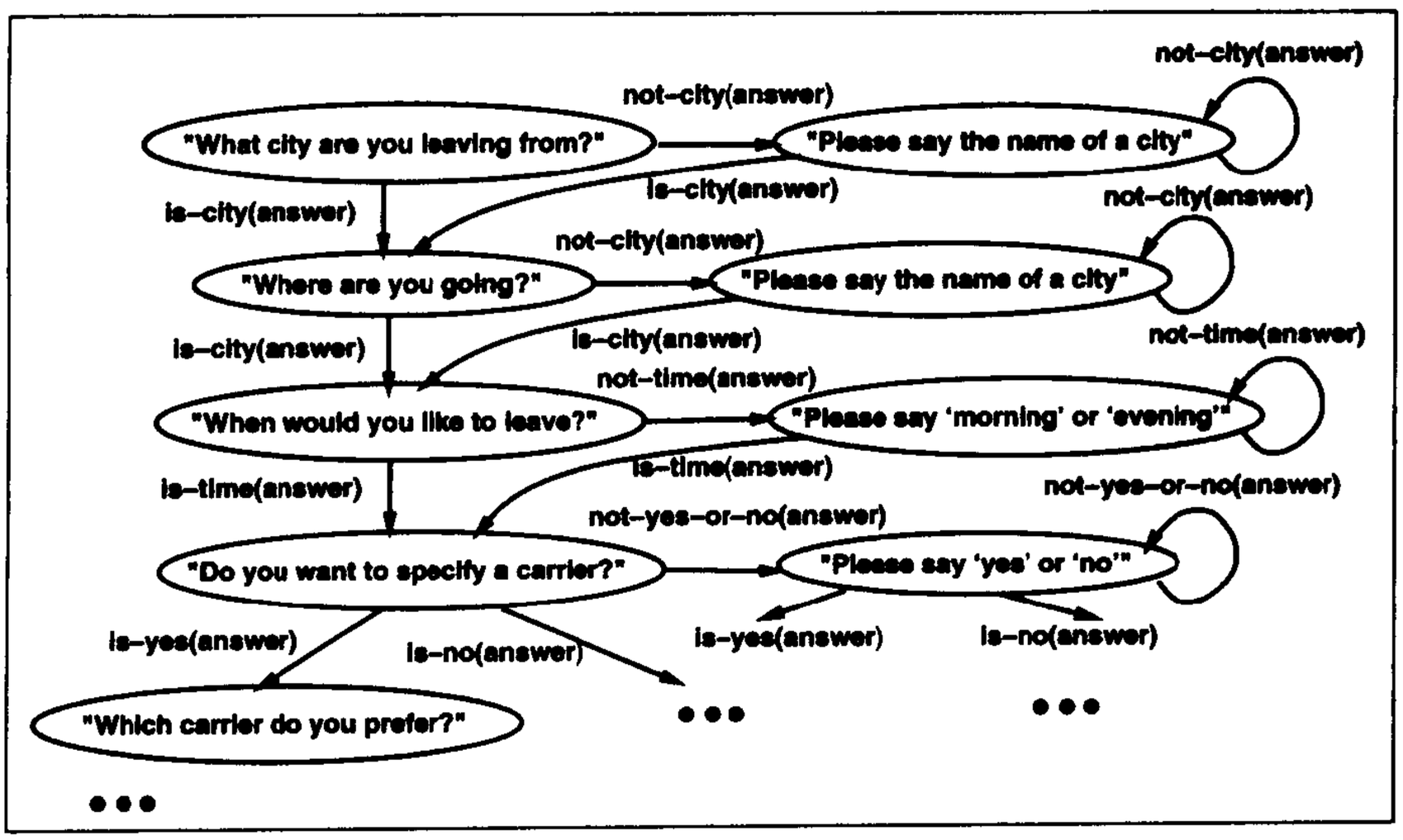


图 19.7 对话管理系统的简单有限状态自动机架构

通过这种方式完全控制会话的系统被称为单一主动（single initiative）或系统主动（system initiative）系统。尽管这个简单对话管理架构对于某些任务（例如实现电话自动应答系统或简单的地理测验的语音界面）是足够的，但是对于基于语音的旅行代理系统来说，它可能存在过多的限制，参见McTear(1998)中的讨论。其中一个原因是用户更喜欢采用能够一次回答多个问题的更复杂的句子，例如下面的ATIS中的例子：

I want a flight from Milwaukee to Orlando one way leaving after five p.m. on Wednesday.

许多基于语音的问答系统都以用于旅行航线计划的具有广泛影响的GUS系统(Bobrow et al., 1977)为基础，并引入基于框架（frame）或模板（template）的较新的ATIS系统以及其他旅行和饭店向导系统。例如，一个简单的航线系统可能具有帮助用户发现合适航班的目标。这可以表示为一个带有各种用户需要指定的信息槽的框架或模板。下面是一些信息槽以及事先指定的询问用户的问题：

Slot	Optional Question
From_Airport	"From what city are you leaving?"
To_Airport	"Where are you going?"
Dep_time	"When would you like to leave?"
Arr_time	"when do you want to arrive?"
Fare_class	
Airline	
Oneway	

这类简单对话管理系统可能仅仅是向用户提问，将答案填入模板，直到拥有足够的信息来进行数据库查询，然后把结果返回给用户。并不是每个槽都有一个相关的问题，因为对话设计者可能并不想让用户面对一连串的问题。然而，如果用户碰巧指定了该信息，则系统必须能够将它们填入对应的信息槽。

即使这种简单的应用领域也需要采用比单一模板复杂的架构。例如，符合用户约束的航班常常不止一个。这意味着在显示屏上给用户列出一个选择列表，或对于完全的对话界面逐条读出该列表。因此，基于模板的系统需要另外的带有能够识别所列航班的信息槽的模板（*How much is the first one*? 或者 *Is the second one non-stop*?）。其他模板可能带有一些常见的路线信息（例如 *Which airlines fly from Boston to San Francisco*?），飞机票价实行的一些信息（例如 *Do I have to stay a specific number of days to get a decent airfare*?）或汽车和饭店预订的信息。因为用户可能在模板之间转换，也因为用户可能会回答预期的问题而不是系统当前所提的问题，所以系统必须能够对给定的输入填入应该对哪个模板的哪个槽进行排歧，然后将对话控制转换到该模板。因此，基于模板的系统本质上是一个生成规则的系统。不同类型的输入激发不同的生成规则，每个生成能够灵活地填入不同的模板。可见生成规则能够基于一些因素进行转换控制，这些因素包括用户的输入和一些简单的对话历史（例如系统所问的最后一个问题）。

模板或生成规则对话管理架构常常用于用户可能采取的行为相对有限，而用户只希望在这些行为中进行少许转换的场合。

基于模板和基于 FSA 的对话管理系统的局限性是非常明显的。研究图 19.5 的对话样例中的客户话段 C_4：

A_3: And what time would you like to leave Pittsburgh?

C_4: Uh hmm I don't think there's many options for non-stop.

A_4: Right. There's three non-stops today.

C_5: what are they?

A_5: The first one departs PGH at 10:00a.m...

通过 C_4，客户接管了对话的控制或主动权。C_4 是一个间接请求，让智能代理系统查询直飞航班。这时系统就不能只把 WANTS NON-STOP 域放入一个模板，而再次询问用户出发的时间。系统需要领悟到用户表示直飞航班优先于其他航班，系统以后应该以此为重点。

会话智能代理系统也需要使用 19.1.2 节所介绍的共同的基础行为。例如，当用户选择航班时，很重要的一点是代理系统向客户表明它已经理解了他的选择。下面给出的是摘自样例的共同基础：

C_6: OK I'll take the 5ish flight on the night before on the 11th.

A_6: On the 11th? OK.

对一个计算对话智能代理来说，利用请求进行修正也是十分重要的，因为考虑到语音识别或理解中可能出现的一些错误，智能代理系统常常会对用户的请求感到迷惑或无法理解。

为了解决这些问题以及其他一些问题，可以建立更复杂的基于BDI（belief，desire和intention，详见19.3.1节）架构的对话管理系统。这类系统常常与基于逻辑的计划模型集成在一起，并将会话当成计划行为的序列。

让我们研究一下在Allen et al.(1995)中描述过的TRAINS-93系统，它的对话管理部分参见Traum and Allen(1994)。TRAINS系统是一个口语会话计划编制智能代理系统，它的任务是管理一个小范围内的铁路运输系统。例如，用户和系统协作编制将一车厢橙子从一个城市运往另一个城市的计划。TRAINS的对话管理系统必须保持会话的流程，并表达会话的目的（例如，为实现橙子成功转运的目的，提出一个可行的计划）。为此，会话管理必须针对对话状态、对话本身的意图以及用户的请求、目标和信念来建立模型。会话管理系统利用会话行为解释器对用户的话段进行语义分析，领域计划器和实施器解决实际领域的问题，生成器为用户生成句子。图19.8给出了TRAINS-93对话管理算法的大纲。

```
DIALOGUE_MANAGER

  while conversation is not finished
    if user has completed a turn
    then interpret user's utterance
    if system has obligations
    then address obligations
    else if system has turn
    then if system has intended conversation acts
       then call generator to produce NL utterances
       else if some material is ungrounded
       then address grounding situation
       else if high-level goals are unsatisfied
       then address goals
       else release turn or attempt to end conversation
    else if no one has turn
    then take turn
    else if long pause
    then take turn
```

图19.8 对话管理算法，源自Traum and Allen(1994)，已稍加修改

该算法将它需要生成的话语行为排成一个队列，基于共同基础（grounding）、对话约束（dialogue obligation）或智能代理的目的（goal）。下面逐一进行讨论。共同基础的行为已经在19.1.2节中介绍过，前面的话段可以通过明显的反输（例如，*uh-huh*，*yeah*，或某些情形下的*ok*）或重述该话段的后面部分来达成共同基础。话段也可以通过“继续”该话段而使其具有共同基础，例如采用一种能表明话段已被清楚理解的方式继续会话，例如问答一个问题。

在TRAINS系统中使用了约束，以使系统能够正确地生成毗邻对的第二部分。也就是说，当用户REQUEST（请求）系统的某些东西，例如REQUEST(Give(List)，或REQUEST(InformIf(NonStop(FLIGHT-201)))，REQUEST为系统建立一种约束，或者解释它然后实施它（给出列表或者提供201航班是否为直飞航班的信息），或者拒绝它。

最后，TRAINS对话系统必须能够推理它自身的目的。对于旅行代理领域，对话管理的目的可能就是找出客户的旅行目的，然后生成正确的计划。让我们假定图19.5中旅行社的会话是由一个系统发出的，然后来研究TRAINS风格的对话管理是什么状态才能产生正确的行为。让我们从样例会话第一个话段之后的对话管理［格式源自 Traum and Allen(1994)］的状态开始（重述如下）：

C_1: I want to go to Pittsburgh in May.

方法论探讨：设计对话系统

对话系统的开发者应该如何选择对话的策略、架构、提示以及错误信息？Gould and Lewis(1985)提出的三个设计原则可以概括为：

关键概念 #8　以用户为中心的设计（User-Centered Design）：研究用户和任务，建立模拟和原型，并针对用户和固定任务进行反复测试。

1. **关注用户和任务**　通过与用户面谈和调研类似的系统，理解潜在用户以及任务的性质。对相关的人与人之间对话的研究也是有益的，尽管人与机器之间的对话通常比人与人之间的对话简单，例如人机对话中很少出现代词，并且即使出现也是能够局部绑定的(Guindon, 1988)。
2. **构建原型**　在儿童读物《Oz 巫师》（The Wizard of Oz）(Baum, 1900)中，最终发现巫师只是由一个躲在幕后的男人模拟的。在 Wizard-of-Oz（WOZ）或 PNAMBIC（Pay No Attention to the Man BehInd the Curtain）系统中，用户与他们认为是软件的系统交流，但实际上是一个人工操作员（“巫师”）躲在一些伪装的界面软件后面，例如(Gould et al., 1983; Good et al., 1984; Fraser and Gilbert, 1991)。一个 WOZ 系统可用于测试还未完全实现的架构，只要有界面软件和数据库。这种系统很难用于精确模拟系统的错误、局限性或真实系统的时间限制，因此 WOZ 研究的结果常常过于理想化。
3. **反复设计**　一个嵌入用户测试的反复设计循环对系统设计是至关重要的(Nielsen, 1992; Cole et al., 1994, 1997; Yankelovich et al., 1995; Landauer, 1995)。例如，Stifelman et al.(1993)和 Yankelovich et al.(1995)发现，语音系统的用户一贯试图干扰（barge in）系统，因此建议对系统中重叠语音的识别部分进行重新设计。Kamm(1994)和 Cole et al.(1993)发现指令性的提示（directive prompt）（例如，“如果你接受这个称呼，就说是，否则说不行”）或采用约束形式(Oviatt et al., 1993)产生的结果比那些开放的提示（例如，“你愿意接受这个称呼吗？”）产生的结果更好。

客户/用户刚刚采用INFORM言语行为完成一个话轮。系统具有找出用户旅行目的的话语目的（例如 *Wanting to go to Pittsburgh on May 15 and returning...*），并为实现该目的生成一个旅行计划。下表给出的是系统状态的5个参数：话语约束列表、预期的传给生成器的言语行为列表、仍需要被承认的言语行为列表、话语目的列表以及该话轮的主动权是由系统还是由用户掌握的：

Discourse obligations:	NONE
Turn holder:	system
Intended speech acts:	NONE
Unacknowledged speech acts:	INFORM-1
Discourse goals:	get-travel-goal, create-travel-plan

在该话段之后，对话管理系统决定为队列添加两个话语行为。第一，承认用户的 INFORM 行为（通过“说明共同基础的状况”），第二，询问用户的下一个问题（通过“说明目的”）。

这可以通过系统的STRIPS风格的计划器推理出来。给定目的get-travel-goal，REQUEST-INFO行为方案告诉系统：询问用户某事是找出该事的一种方式。添加这两个会话行为的结果是：

Intended speech acts: REQUEST-INFORM-1, ACKNOWLEDGE-1

这可以通过一个非常聪明的生成器集成为下面的单个话段：

A_2: And, what day in May did you want to travel?

注意，共同基础的功能是通过句子开头的话语标记*and*以及重述月份的名字*May*来实现的。信息请求是通过 wh 疑问句实现的。

让我们跳到客户的话段 C_4。回忆一下，C_4 是一个间接请求，让智能代理核对直飞航班。

A_3: And what time would you like to leave Pittsburgh?

C_4: Uh hmm I don't think there's many options for non-stop.

假设对话行为解释器能够正确地将 C_4 解释为 REQUEST-INFORM-3。则客户话段 C_4 之后，智能代理的状态为：

Discourse obligations:	address(REQUEST-INFORM-3)
Turn holder:	system
Intended speech acts:	NONE
Unacknowledged speech acts:	REQUEST-INFORM-3
Discourse goals:	get-travel-goal, create-travel-plan

对话管理将首先通过调用计划器找出所有直飞航班，以处理回应用户请求的话语约束。现在系统必须回答问题，但是也必须与用户的话段达成共同基础。对于直接请求，回应本身就意味着达成了共同基础，而对于间接请求，一个明显的承认是一种选择，因为本例中的间接请求是一个否定形式的核对疑问，承认的形式可以使用单词*right*（*no*也适用于承认一个否定形式），则这两个行为将离开队列并传给生成器：

A_4: Right. There's three non-stops today.

对话管理也需要处理 19.4 节讨论过的那种对话结构，识别用户开始一个子对话的时间，并明确自身发出子对话的时间。

19.6 小结

对话是一种特殊类型的话语，特别是它与诸如会话智能代理和自动会议摘要等语音处理任务密切相关。

- 对话与其他话语类型的区别表现在话轮转换（turn-taking）、对话的共同基础（grounding）以及对话的隐涵（implicature）。
- 对话行为（dialogue act）解释是对话模型的重要组成部分。我们介绍了对话行为解释的基于计划（plan-based）和基于提示（cue-based）的算法。
- 除了包括支配关系（dominance）、满足－优先关系（satisfaction-precedence）等类型的关系的信息结构（informational structure）之外，对话也表现出意图结构（intentional structure）。
- 用于会话智能代理的对话管理既包括简单的模板或框架的生成系统（production system），也包括完整的 BDI（belief-desire-intention）模型。

方法论探讨：对话系统的评测

目前人们已经提出了许多用于评测对话系统的标准，这些标准可以分为下面三种类型：

1. **用户满意度**（User Satisfaction）　常常通过与用户面谈(Stifelman et al., 1993; Yankelovich et al., 1995)或让用户填写问卷的方式来进行评测，例如询问如下的问题(Shriberg et al., 1992; Polifroni et al., 1992)：
   - 系统给出问题的速度足够快吗?
   - 系统是否理解你的第一次请求?
   - 你认为一个熟悉计算机的人能够很容易地使用该系统吗?
2. **任务完成的耗费度**（Task Completion Cost）:
   - 完成时间以话轮数或秒为单位(Polifroni et al., 1992)。
   - 疑问数(Polifroni et al., 1992)。
   - 系统没有回复的问句的数目或“话轮修正率”：只用于修正错误的系统或用户话轮的数目除以总的话轮数(Danieli and Gerbino, 1995; Hirschman and Pao, 1993)。
   - 系统给出的问题、回答和错误信息的适当性（冗余或歧义）(Zue et al., 1989)。
3. **任务完成的成功度**（Task Completion Success）:
   - 完成的子任务的百分比(Polifroni et al., 1992)。
   - 每个问题、回答和错误信息的正确性(或部分正确性)(Zue et al., 1989; Polifroni et al., 1992)。
   - 总的解决方案的正确性(Polifroni et al., 1992)。

如何才能将这些衡量标准进行组合与加权？PARADISE（PARAdigm for Dialogue System Evaluation）算法(Walker et al., 1997)采用多次回归来解决该问题。该算法首先采用问卷调查为每个对话指派一个用户满意率。然后，与上面类似的那些耗费和成功因子集合被看成是相互独立的因子集合，并采用多次回归为每个因子训练一个权重（系数），该系数用于衡量该因子在用户满意度中的重要性。作为结果的衡量标准可用于比较不同的对话策略。

19.7　文献和历史说明

口语和语言处理的早期研究很少以对话研究为核心。ELIZA是最早的会话系统之一，它仅有一个极为普通的对话管理生成系统。如果人工用户的句子满足可能回复的正则表达式的前提，则ELIZA可以简单地生成该回复(Weizenbaum, 1966)。模拟妄想狂的对话管理系统PARRY(Colby et al., 1971)更复杂一点。与ELIZA相同，它也是基于生成系统的，但是ELIZA的规则只是基于前面用户给出句子中的单词，而PARRY的规则还基于表示情感状态的全局变量。而且，PARRY的输出有时会采用类似脚本的陈述序列。当会谈转向它的幻想时，例如，如果PARRY的愤怒（anger）变量较高，它就会从“敌对”集中选择输出句。如果输入句提及它所幻想的主题，它将提高害怕（fear）变量并开始表达与它的幻想有关的句子。

更复杂的对话管理系统的出现需要在人们对人－人对话有更好的理解之后。在20世纪70年代至80年代，有关人－人对话特性的研究开始增加。会话分析学界(Sacks et al., 1974; Jefferson, 1984; Schegloff, 1982)开始研究会话的交互属性。Grosz(1977b)对于对话的计算研究有较大的影响，引入了对话（子对话）中的子结构的研究，特别是发现“面向任务的对话的结构十分类似于所实施任务的结构”(p.27)。Cohen and Perrault(1979)首次通过BDI模型将早期的AI计划研究(Fikes and

Nilsson, 1971)与口语行为理论(Austin, 1962; Gordon and Lakoff, 1971; Searle, 1975a)结合在一起，提出了口语行为的生成方法，Perrault and Allen(1980)和 Allen and Perrault(1980)也将该方法用于语言行为解释。

对话行为解释的基于提示的模型受到Hinkelman and Allen(1989)的启发，其研究工作表明词汇和短语提示可以集成于BDI模型，也受到会话分析文献，例如(Goodwin, 1996)的微语法的启发。20世纪90年代晚期世界各地的许多主要语音识别实验室都实现了该模型，例如(Nagata and Morimoto, 1994; Suhm and Waibel, 1994; Mast et al., 1996; Jurafsky et al., 1997; Warnke et al., 1997; Reithinger and Klesen, 1997; Taylor et al., 1998)。

20世纪80年代到90年代晚期对话模型作为协同行为的思想被引入研究工作，包括所指作为协同处理(Clark and Willkes-Gibbs, 1986)的思想，接合意图(joint intention)的模型(Levesque et al., 1990)以及分享计划（ shared plan)(Grosz and Sidner, 1980)。与该领域相关的是对话中主动性（ initiative ）的研究，以及参与者之间对话控制转移的研究(Walker and Whittaker, 1990; Smith and Gordon, 1997)。

第20章　自然语言生成[①]

"hello, world"
Kernighan & Ritchie, *The C Programming Language*

"... you, MR KEITH V LINDEN, will be a millionaire January 31!"
From a junk mailing

从某种意义上说，语言生成是语言处理的最早的分支领域。在计算机只能理解最不自然的指令语言时，它就能“吐出”自然的文本。例如，最早也是最著名的C程序，即hello, world程序，就是一个生成程序。它生成了精彩而优美的英文文本。遗憾的是，这些文字所拥有的微妙而卓越的交际语力其实不是由程序本身而是由该程序的作者创作的。这种生成方法，被称为罐装文本（canned text），虽然易于实现，但是不经过程序员的干涉就不可能适应新的情形。

语言生成也是语言处理最具普遍性的分支领域。几乎每个人都收到过在正确位置嵌着自己的名字的信件，它以动人的文字请求我们做这样或那样的事。这类程序也是易于实现的，大部分人都不会认为这种信件是由人写成的。当我们的名字被打乱（就好比上面的垃圾邮件中的地雷）时，或产生一些明显的其他错误时[②]，这种机制的不灵活性就会陡然显现。这种方法被称为模板填充（template filling），它比罐装文本的方法灵活，也已经用于各种应用，但是仍很有限。例如，Weizenbaum在ELIZA中使用的模板，在有些情形下性能良好，而其他时候则只是胡言乱语[③]。

简单生成机制的成功似乎说明：语言的生成比语言的理解更容易一些。一个语言理解系统通常不可能控制它所接收的输入语言结构的复杂性，而一个生成系统可以限制它所输出的语言结构的复杂性。正因为如此，语言处理的研究最初集中于语言理解，认为所需的任何生成都可以很容易地通过罐装文本或模板填充机制得以实现。遗憾的是，这些简单机制的灵活性不足以处理被表达信息，以及它的上下文中带有实际变化的应用。即使在“hello，world”程序这样的有限领域中应用的生成系统，也可以被更灵活地应用。该程序生成下面的文字也许会更恰当：

Congratulations, you've just compiled and run a simple C program　　(20.1)
which means that your environment is configured properly.

这段文字比原先的文字更复杂，而且我们可以看到许多可能的变化。如果读者是一个资深软件工程师，我们也许不应该祝贺他们编译了一个程序。这样做可能会侮辱他们。另外，对于问题中的program，这里存在许多其他表达方式，包括the program，your first C program，it以及that lame excuse for code。在给定的情形下，它们中的每一个都可能是，也可能不是恰当的。注意到不必联合compiled

---

① 本章的作者是Keith Vander Linden。

② 最近出现的一个例子是，报考大学的学生的家长收到的录取信中的内容：“Congratulations on 987-65-4321's admission ... as a parent you will be a partner with the university in encouraging 987-65-4321 to succeed."（摘自联合通讯社，1999年3月26日）。

③ 让ELIZA胡言乱语并不困难，比如下面与ELIZA 4.3的对话：HUMAN: "Let's talk just about you – not me. Can you think? " ELIZA: " You like to think I – not you – don't you" ... HUMAN: "I think you are an extremely stupid program." ELIZA: "Perhaps you would like to be an extremely stupid program" (Copeland, 1993, p.40)。

and run，我们可以表达结论（your environment is configured properly），然后是证据（you've compiled and run a simple C program）。这些变化并不一定是同义的。语言的发展产生了表达事物的各种方式，所有这些方式对于在特定上下文中特定事物的表达都是十分有用的。因此，为了能够用于更复杂的环境，一个生成系统必须能够：(1) 给出形式的正确种类；(2) 基于预期的意义和上下文在这些形式中进行选择。本章将忽略罐装文本和基于模板的机制，而研究用于解决这些问题的基本语言生成技术。本章也将不再讨论已经在第 4 章介绍过的文本 - 语音合成。

20.1 语言生成导引

"Language understanding is somewhat like counting from one to infinity; language generation is like counting from infinity to one." ①
Yorick Wilks, quoted in Dale et al. (1998a, p. 352)

"Generation **from what?!"** ②
attributed to Christopher Longuet-Higgins

自然语言生成（Natural Language Generation，简称 NLG）是指从非语言输入构造自然语言输出的处理。该处理的目的可以看成与自然语言理解（Natural Language Understanding，简称 NLU）的目的相反，NLG 是从意义到文本的映射，而 NLU 是从文本到意义的映射。在实现这个映射时，生成采用的许多语言学问题与前几章的讨论相同，但方向相反，使得它的方法与 NLU 的那些方法有两个重要的不同。

首先，用于生成处理的输入的性质根据应用的不同会有很大变化。尽管用于 NLU 系统的语言学输入可能会因文本类型的不同而有所变化，但是所有这些文本都是在一种相对统一的语法规则支配下的。生成系统则不是这样。每个生成系统面对的是带有不同输入规范的不同应用。一个系统可能是数值表的复杂集的解释，而其他系统可能是用面向对象的软件工程模型的结构组成的。因此，生成系统必须能够抽取驱动生成处理所需的信息。

其次，尽管 NLU 和 NLG 都必须能够表示一系列应用领域所需的词汇和语法形式，但是它们对这些表示的使用方式是不同的。NLU 以假设管理（hypothesis management）处理为特征，在语言学输入被顺序扫描的同时，系统考虑可供选择的解释。在 NLU 中最受关注的是歧义的、不确定的以及非良构输入，而这些不是生成研究所关注的，因为它们通常并不出现。NLG 系统的非语言学的表示输入往往是较少歧义的、指定的和良构的输入。相反，在 NLG 中最为关注的是选择（choice）。生成系统必须做出下面的选择：

- **内容选择** 基于特定交际目的的系统决策，系统必须从完全指定的可能输入中选择正确的内容。例如，我们注意到例句（20.1）包括的一些内容并不适合于所有读者。如果目的是表明该环境已建立，而读者是一名软件工程师，则我们可能只选用最后一个句子来表达。
- **词汇选择** 系统必须选择能最恰当地表达特定概念的词汇项。例如，在例句（20.1）中，必须在单词 configured 和其他可能形式（包括 set up）之间进行选择。
- **句子结构**
  - **集结** 系统必须将选择的内容分配给短语、从句和句子长度的词块。在例句（20.1）中，将 compiling 和 running 合并为一个短语。

① 译文：在某种意义上语言理解就好比从 1 数到无穷大；而语言生成就好比从无穷大数到 1。
② 译文：到底从什么生成？！

- **所指表达**　系统必须确定如何指向所讨论的客体。在例句（20.1）中如何指向 program 的决策不是无关紧要的。

● **话语结构**　NLG 系统经常处理多个句子的话语，它必须具有一个连贯且可分辨的结构。例句（20.1）就包含两个命题，显然其中一个是另一个的根据。

这些选择问题，与将单词的线性序列组成文本的实际问题一起，构成了 NLG 领域的核心。尽管这是一个相对年轻的领域，但已经有许多针对该核心的研究工作。本章将介绍这些成果，首先将给出一个简单的 NLG 系统的体系结构，然后将讨论组成该体系结构的主要组件的常用技术。

20.2　生成的体系结构

适合完成上一节所列出任务的体系结构到底应该具有什么样的特征，引起了极大的争论。然而，从实际考虑出发，所采用的体系结构常常如图20.1所示。这个体系结构包含由流水线连接的两个组件：

● **话语规划器（discourse planner）**　该组件由交际目的开始，实施上一节讨论的所有选择。从知识库中选择内容并使这些内容具有正确的结构。作为结果的话语规划将为可能跨越多个句子并包括其他注解（比如超文本、图表等）的整个交际指定所有的选择。

● **表层实现器（surface realizer）**　该组件接受完全指定的话语规划，并生成由句子词汇和语法知识源约束的单个句子。这些知识源决定了实现器的可能输出的种类。如果规划指定了多个句子的输出，则表层实现器就需要被多次调用。

这绝不是人们提出的NLG系统的惟一体系结构。其他可能机制包括AI风格和黑板体系结构。这也不是一个尽善尽美的体系结构。例如，这种简单的流水线设计就不容许在表层实现阶段对规划器的决策重新考虑。而且，规划与实现之间的精确分界线并不总是很清晰。我们采用它只是为了能够更好地组织本章的内容。首先讨论这两个组件中得到较充分发展的表层实现器，然后讨论话语规划器。

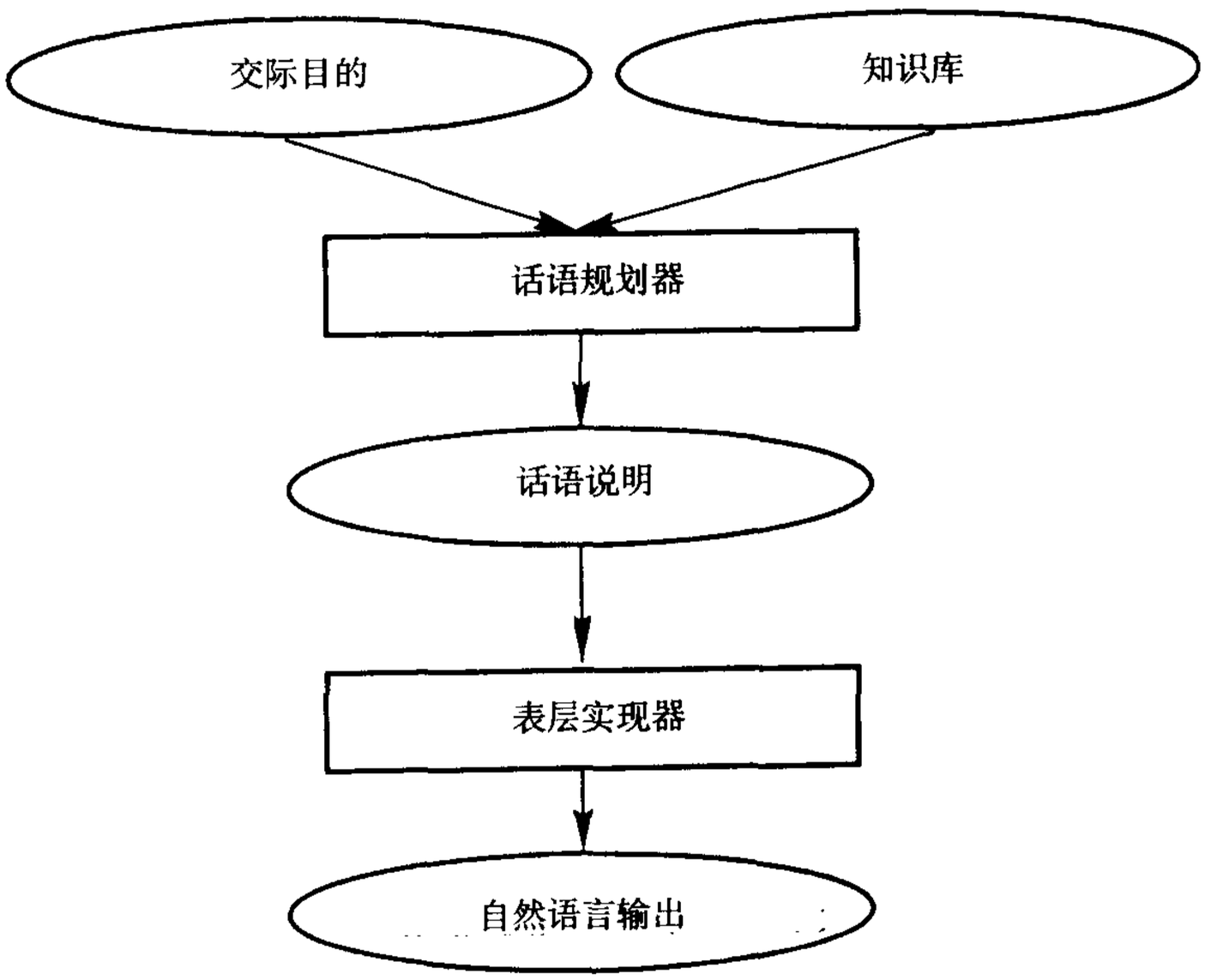


图 20.1　供参考的 NLG 系统的体系结构

20.3 表层实现

表层实现组件生成了由词汇和语法约束的具有顺序的单词序列。它所采用的输入是话语说明的句子长度的词块。本节将介绍用于该任务的最有影响的两种方法：系统语法（systemic grammar）和功能合一语法（functional unification grammar）。这两种方法都将用于生成下面的例句：

The system will save the document. (20.2)

对于表层实现器的输入应该被指定为哪一层，目前还没有达成共识。有些方法只指定命题的内容，拿例句（20.2）来说，话语规划需要指定一个由 system 实体对 document 实体的 saving 行为，而其他方法中也有对语法形式（本例中的将来时态的声明）和词汇项（本例中的 *save*，*system* 和 *document*）的指定。

我们将看到，采用本节所讨论的这两种方法的系统，所选取的输入隶属于不同的层次。当然，它们也有一个共同点：选取的都是功能性地指定的输入，而不是句法性地指定的输入。这一事实是生成系统的典型特征，也倾向于排除对本书较早讨论过的句法形式体系的使用。生成系统由意义和上下文开始，因此根据功能（function）而不是形式（form）是最自然的指定预期输出的方式。比如，例句（20.2）既可以采用主动语态也可以采用被动语态的形式。话语规划器倾向于不从这些句法的角度进行实施，而是更倾向于遵循话语的焦点或局部主题，因此更自然的方式是根据焦点来指定这种差别。在本例中，如果 document 是话语的局部主题，它就会被标记为能够触发采用被动语态的焦点。这里讨论的两种方法都隶属于功能语法的范畴。

20.3.1 系统语法

系统语法是系统功能语言学（Systemic-Functional linguistics）的一部分，是一个把语言看成是在上下文中表示意义的资源的语言学分支(Halliday, 1985)。系统语法把句子表示为功能集，并维护着这些功能与外在语法形式之间的映射的一些规则。该方法非常适用于生成，因此在 NLG 中已经具有了广泛影响。本节将从系统句子分析的一个例子开始，然后讨论简单的系统语法并将它用于连续的例子。

系统语法的句子分析要组织表示为多层结构的这些功能，如例句（20.2）的分析所示：

	The system	***will***	***save***	***the document***
Mood	subject	finite	predicator	object
Transitivity	actor	process		goal
Theme	theme	rheme		

这里，语态（Mood）层表示是一个带有主语、定式（助动词）、谓词（动词）和宾语的简单陈述句结构。及物（Transitivity）层表示 *system* 是 *saving* 处理的行为者或实施者，而目标或行为的宾语是 *document*[①]。主题（Theme）层表示 *system* 是句子的主位或关注的焦点。主位（theme）或述位（rheme）的概念是由 Prague 语言学学派提出的(Firbas, 1966)。注意，这三层处理的是不同的功能集。这三个功能集被称为元功能或纯理功能（meta-function），表示生成所涉及的三种基本功能：

① 我们在第 16 章中已经给出了对这些题元角色的讨论。

- 人际元功能（interpersonal meta-function）是由建立和维护作者与读者之间的交互关系的那些功能组成的。这里由语态层表示，确定了作者是在命令、倾诉或询问。
- 概念元功能（ideational meta-function）与通常被称为表达的“命题内容”有关。这里，及物层确定了正在表达的处理以及必须被表示的各种格角色的性质。注意，这个元功能涵盖了许多我们以前称之为“语义学”的内容。
- 语篇元功能（textual meta-function）与适合于当前话语的表达方式有关。这包括主题化（thematization）以及所指的问题。在我们的例子中，由主题层表示，它清楚地表明the system是句子的主题。

人际和语篇的论点与传统语义学之间具有明确的关系，是系统语言学吸引NLG的另一个特征。生成系统需要做出的许多决策都依赖于交际的背景，而这是通过人际和语篇元功能来形式化的。

系统语法能够构建一个句子的结构，比如刚刚给出的例句的结构。用有向、非循环、and/or图表示的语法称为系统网络（system network）。图20.2以举例的方式给出了一个简单的系统网络。这里，最大的波形括号表示 and（也就是并列的）系统，而笔直的垂直线表示 or（也就是不相交的）系统。因此，每个从句可以同时具有语态、及物和主题的特征集，但是不可能既是indicative（指示语）又是 imperative（祈使语），只可能是其中之一。尽管系统网络的形式化不需要利用系统理论，但是我们仍将宽松地把这个简单语法归类于系统范畴。依照这个语法，例句（20.2）是一个用未标记主题表示主动物质加工（material process）的指示性陈述句。

系统语法采用实现语句（realization statement）以建立语法指定的特征（比如指示语、祈使语）与句法形式之间的映射。网络中的每个特征都具有一个实现语句集，并通过该语句集指定了对最终表达形式的约束。每个特征的实现语句在图（20.2）中以特征下面的斜体语句集来表示。当系统网络被遍历时，实现语句容许将语法用于约束表达形式的结构。所采用的简单运算集如下所示：

$+X$	插入功能 X。例如，图 20.2 的语法指定所有从句都需要一个谓词。
$X = Y$	合并功能X和Y。这意味着容许语法通过对表达式的相同部分指派不同的功能而建立一个层次化功能结构。例如，主动语态的从句将行为人与主语合并，而被动语态的从句将目的与主语合并。
$X > Y$	将功能 X 置于功能 Y 之前的某一位置。例如，表示句子的主语必须在谓词之前。
X / A	将功能 X 与词汇或语法特征 A 划为一类。这种分类发出了在较下层递归调用该语法的信号。该语法应该包括其他网络，类似于可以应用于短语、词汇项和词形学的从句网络。比如，指示语特征插入了一个主语功能，则该主语功能必须是名词短语。这个短语将通过下一次调用该语法而被进一步指定。
$X ! L$	词典项 L 赋值给功能 X。在图 20.2 中，被动语态的定式被指派为词汇项 be。

在给定一个完全指定的系统网络的情形下，生成的处理程序是：

1. 从左到右遍历网络，选择正确的特征并收集相关的实现语句；
2. 建立中间表示，该中间表示满足遍历期间收集的实现语句所施加的约束；
3. 对于任何没有完全指定的功能，通过在较下层递归调用该语法而加以指定。

为了说明这个过程，我们将采用样例的语法来生成例句（20.2）（*The system will save the document*）。我们将以下面的说明作为输入[①]：

① 这个输入说明大体上是基于生成系统PENMAN(Mann, 1983)的spl构造器的界面。句子规划语言（Sentence Planning Language，简称 SPL）的讨论将在 20.7 节中给出。

```
(
  :process    save-1
  :actor      system-1
  :goal       document-1
  :speechact assertion
  :tense      future
  )
```

这里，知识库实例 save-1 被识别为该预期表达的处理。我们将假定所有的知识库客体都是 KLONE 风格化的实例(Brachman, 1979)，因为存在有关它们的正确的词典选项。行为人和目的类似地被分别指定为 system-1 和 document-1。这个输入也指定该表达应该为具有将来时态的断言句。

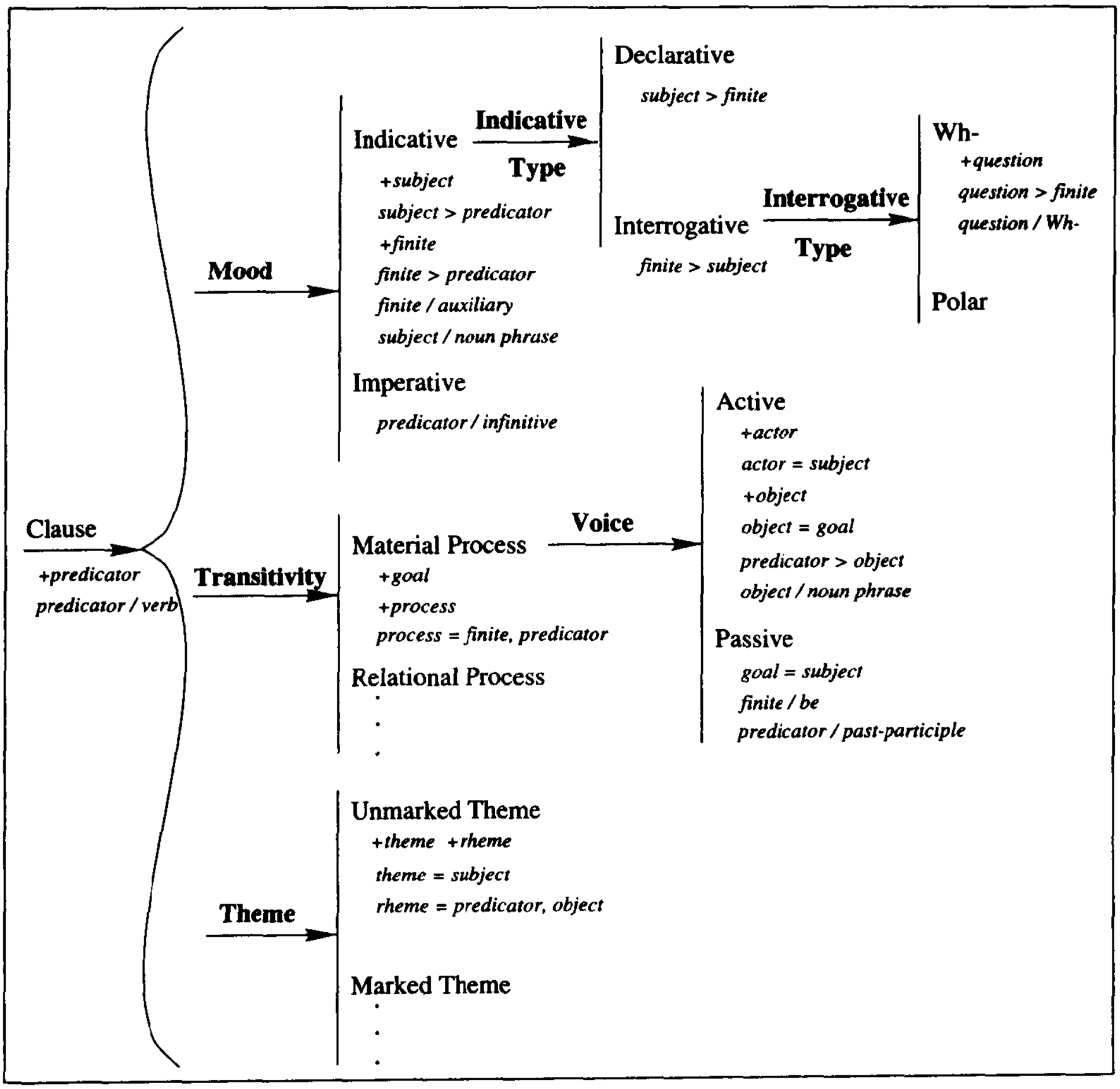


图 20.2 一个简单的系统语法

生成处理从图 20.2 中的从句特征开始，插入一个谓词并将它分类为动词。然后，继续进行至语态（mood）系统。一个系统的正确选项是通过与系统有关的简单查询或决策网来选择的。查询或决策网是基于从输入说明以及知识库获取的相关信息来进行决策的。在本例中，语态系统选择指示语和宣告语的特征，因为输入说明指定这是一个断言句。与指示语和宣告语特征有关的实现语句

将插入主语和定式功能，并将它们排列为主语（subject）、定式（finite）、谓词（predicator）的顺序。所得到的功能结构如下：

Mood	subject	finite	predicator

我们将假定 save-1 行为被标记为知识库中的物质加工，这导致及物系统选择物质加工（material process）的特征，插入目的（goal）和处理（process）功能，并合并处理定式/谓词对。这里，无论是输入还是知识库都没有指定采用被动语态，因此系统选择主动语态的特征，(1) 插入行为人（actor）并将它与主语合并；(2) 插入宾语，将它与目的合并，并将它列于谓词之后。结果如下：

Mood	subject	finite	predicator	object
Transitivity	actor	process		goal

最后，因为输入中没有主题说明，主题网络选择未标记主题（unmarked theme），插入主位（theme）和述位（rheme），把主位与主语合并，把述位与定式/谓词/宾语组合并。所得到的结果就是上面讨论过的完整的功能结构（重述如下）：

Mood	subject	finite	predicator	object
Transitivity	actor	process		goal
Theme	theme	rheme		

此时，生成处理递归地多次进入语法的较低层次，以完全指定短语、词典项以及词形。名词短语网络将采用与这里给出的过程类似的处理来生成 *the system* 和 *the document*。辅助动词网络将为系统插入词汇项 will。词汇项 *system*，*document* 和 *save* 的选择可以通过多种方式来处理，其中最典型的是从相关知识库实例中检索词汇项。

20.3.2　功能合一语法

功能合一语法（Functional Unification Grammar）采用合一运算（第11章曾讨论过）来操作和推理特征结构(Kay, 1979)。该技术只需经过小的修改就可以应用于NLG。它的基本思想是将生成语法建立为带有可能的交替列表的特征结构，然后将该语法与采用输入说明建立的相同类型的特征结构进行合一。这个合一处理将选取在输入中指定的特征，并且将它们与那些在语法中的特征进行一致化，生成全部的特征结构，在被线性化后形成句子的输出。

本节将通过再次生成例句（20.2）来说明这个机制。我们将采用图20.3所示的简单功能合一语法。该语法被表示为属性值矩阵（见第11章），支持现在或将来时态的简单及物句子，并坚持主谓数的一致。现在我们将跳过结构而解释其特征。

在它的最高层，该语法为句子（cat s）、名词短语（cat np）和动词短语（cat vp）提供了可供替换的对象。这个可选方法是由最左边的 alt 特征指定的。我们用波形括号表示，它所包括的这三个替换对象中的任何一个都是容许的。这一层也指定了一个模式，该模式表明了在该层指定的特征的顺序，本例中的顺序为：行为人（actor）、处理（process）和目的（goal）。

在句子层，该语法支持的 *actor*，*process* 和 *goal* 特征分别被预先指定为 *NP*，*VP* 和 *NP*。主谓数的一致通过采用 process 特征内的 number（数）特征得以实施。其中，process 的 number 必须与

路径{actor number}统一。路径是指定从根结点到一个特定特征的路径的特征列表。在本例中，process 的数必须与 actor 的数统一。这个路径是直接给定的，但是我们也可以采用相对路径，比如*NP*的中心语（head）特征的特征数。这里的路径{↑↑number}表示名词短语的中心语的数必须与高于两个层次的特征的数相统一。在下面的例子中，我们将看到这是非常有用的。

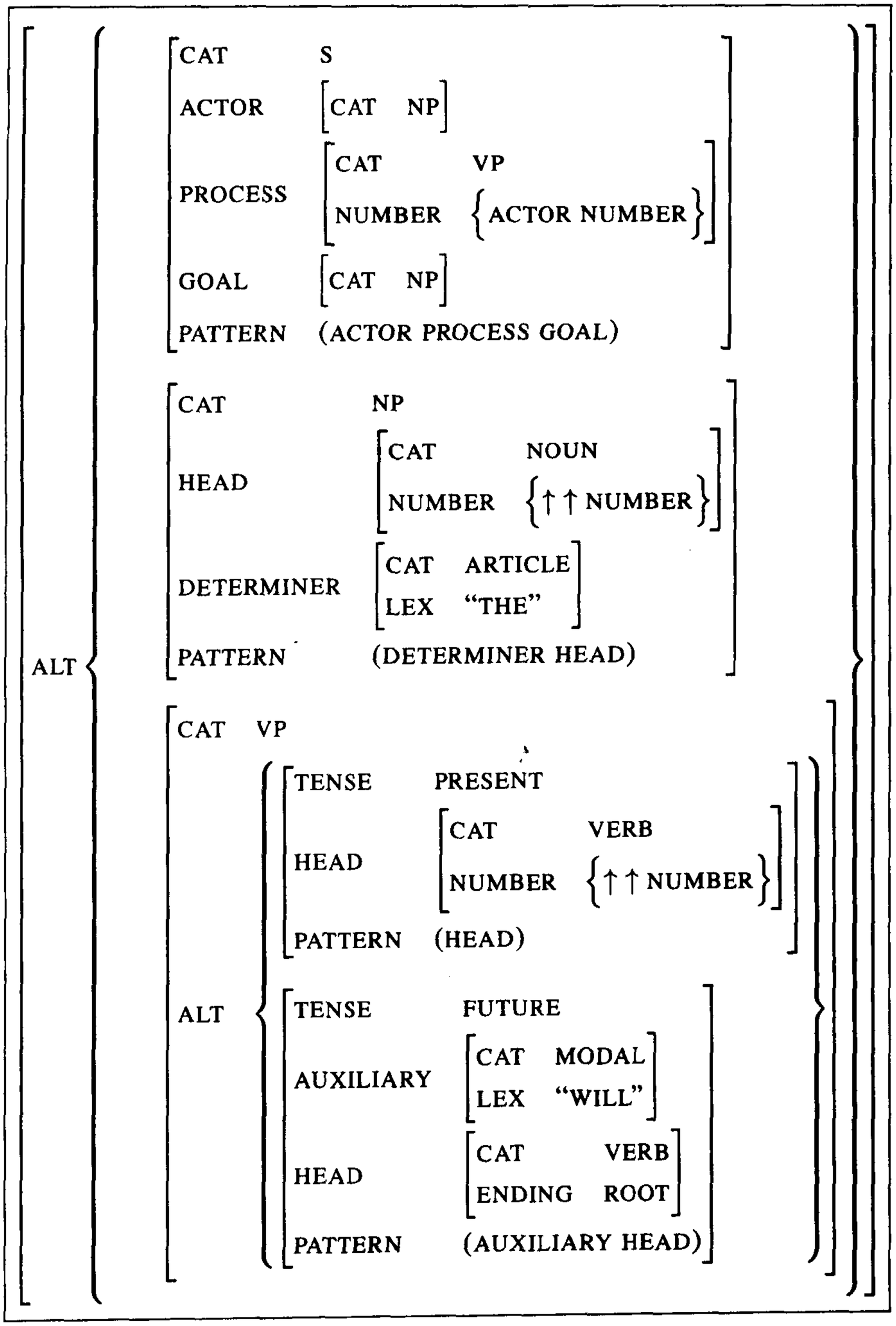


图 20.3 一个简单FUF语法。FUF，即Functional Unification Formalism，是由 Elhadad(1993, 1992)开发的对功能合一语法的一种实现

除了 *VP* 层独有的在现在和将来时态之间的替换，*VP* 层在性质上类似于NP层。给定时态（我们将看到这是在输入特征结构中指定的），合一运算将选择匹配的替换对象，并继续处理相关特征的统一。例如，如果时态是现在时，则中心语为单个动词；如果时态是将来时，则需要在中心动词之前插入辅助动词 will。

该语法与上一节介绍的系统语法的相似之处在于它也支持在生成处理时递归地进入多个层次。现在我们来看看输入特征结构，它指定了我们希望生成的特定句子的细节。该输入结构称为功能描述（functional description，简称 FD），是一种与该语法相似的特征结构。例句（20.2）的 FD 如下所示：

$$\begin{bmatrix} \text{CAT} & \text{S} \\ \text{ACTOR} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{LEX} & \text{SYSTEM} \end{bmatrix} \end{bmatrix} \\ \text{PROCESS} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{LEX} & \text{SAVE} \end{bmatrix} \\ \text{TENSE} & \text{FUTURE} \end{bmatrix} \\ \text{GOAL} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{LEX} & \text{DOCUMENT} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

这里我们看到了对一个带有特定行为人（actor）system 和特定目标（goal）document 的句子的说明。该处理（process）是指通过系统（system）在将来某个时候保存（saving）文件（document）。输入结构指定了被采用的特定动词和特定名词以及时态。这与系统语法的输入不同。在系统语法中，词汇项是从知识库里与行为人和目的相关的实体中检索出来的。时态，没有包含在作为样例的系统语法中，是由决策网络通过区分与表达内容有关的相对时间点而确定的。因此，合一语法需要通过话语规划组件做出更多的决策。

为了产生该输出，将这个输入与图 20.3 所示的语法进行合一。这需要多次调用该语法。初始的合一运算将输入FD与语法的S层进行合一（也就是说，顶层的第一个替换对象）。该处理导致的结果如下：

$$\begin{bmatrix} \text{CAT} & \text{S} \\ \text{ACTOR} & \begin{bmatrix} \text{CAT} & \text{NP} \\ \text{HEAD} & \begin{bmatrix} \text{LEX} & \text{SYSTEM} \end{bmatrix} \end{bmatrix} \\ \text{PROCESS} & \begin{bmatrix} \text{CAT} & \text{VP} \\ \text{NUMBER} & \{\text{ACTOR NUMBER}\} \\ \text{HEAD} & \begin{bmatrix} \text{LEX} & \text{SAVE} \end{bmatrix} \\ \text{TENSE} & \text{FUTURE} \end{bmatrix} \\ \text{GOAL} & \begin{bmatrix} \text{CAT} & \text{NP} \\ \text{HEAD} & \begin{bmatrix} \text{LEX} & \text{DOCUMENT} \end{bmatrix} \end{bmatrix} \\ \text{PATTERN} & (\text{ACTOR PROCESS GOAL}) \end{bmatrix}$$

这里我们看到了输入结构中指定的特征已经与语法顶层的特征进行了融合与统一。例如，与actor有关的特征包括来自输入*FD*的词汇项system以及来自语法的类别*NP*。类似地，process特征融合了来自输入*FD*的词汇项和时态以及来自语法的类别和数的特征。

现在，生成机制为每个子成分递归地调用语法。它两次进入*NP*层，一次为actor，一次为goal，并且为process进入*VP*层一次。由此所产生的FD结果如图20.4所示。我们看到每个内在复杂成分的特征都有一个模式（pattern）说明，而每个简单成分的特征都有一个词汇说明。现在系统就可以利用模式说明使该输入线性化，从而生成句子*The system will save the document.*。

```
[CAT      S
 ACTOR    [CAT      NP
           HEAD     [CAT      NOUN
                     LEX      SYSTEM
                     NUMBER   {↑↑NUMBER}]
           DETERMINER [CAT     ARTICLE
                       LEX     "THE"]
           PATTERN    (DETERMINER HEAD)]
 PROCESS  [CAT      VP
           NUMBER   {ACTOR NUMBER}
           HEAD     [CAT      VERB
                     LEX      SAVE
                     ENDING   ROOT]
           AUXILIARY [CAT     MODAL
                      LEX     "WILL"]
           TENSE    FUTURE
           PATTERN  (AUXILIARY ROOT)]
 GOAL     [CAT      NP
           HEAD     [CAT      NOUN
                     LEX      DOCUMENT
                     NUMBER   {↑↑NUMBER}]
           DETERMINER [CAT     ARTICLE
                       LEX     "THE"]
           PATTERN  (DETERMINER HEAD)]
 PATTERN  (ACTOR PROCESS GOAL)]
```

图20.4　完全合一的FD

在这个特殊的例子中并没有指定 actor 为复数。这一点可以通过对输入 FD 的 actor 结构添加特征值对 number plural 来实现。然后，主谓一致将通过合一处理来支持。该语法需要 *NP* 和 *VP* 的中心语的数与输入 FD 中指定的 actor 的数相一致。

20.3.3　要点

我们以本节看到的两种表层生成语法为例，说明了用于生成的计算语法的特性。两者都隶属于功能范畴。也许有人想知道对于生成和理解这两者而言，采用单一的语法是否可能。这种语法称为双向语法（bidirectional grammar），目前仍在研究中，还没有被广泛应用于 NLG（见第 21 章）。这在很大程度上是因为需要额外的语义和背景信息作为生成器的输入。

20.4　话语规划

上一节讨论的表层实现组件借助指定的输入来生成单个句子。因此，它对句子驻留的话语结构或句子本身的内容都几乎不可能进行控制，而这些是由话语规划器来控制的。本节将介绍建立话语结构的两种主要机制：文本说明图（text schemata）和修辞关系（rhetorical relation）。

NLG 的许多研究都具有一个重要特征，即关注的焦点是话语而不只是句子。许多应用需要系统产生多个句子或多个话段的输出。这可以通过简单地对预期意义的每个部分生成一个句子来实现，但是在以正确的方式进行意义选择和构造时常常需要加倍小心。例如，研究下面例句，它是我们在引言中讨论过的“hello, world”输出的可替代版本：

You've just compiled a simple C program. You've just run a simple C program. Your environment is configured properly.　　(20.3)

这些句子单个来看都很完美，但是与例句（20.1）相比就显得较为松散，也可能较难理解。尽管句子的顺序会有一定的帮助，但是它并不能说明句子之间的任何关系。这些问题驱使我们进行话语规划的研究。

本节也将讨论密切相关的内容选择问题。在前面已经看到，是根据交际目的从输出知识库中选择命题内容的处理。因为不同应用的知识库的形式和交际目的的性质各不相同，很难给出关于内容选择处理的一般的陈述。因此，为了使问题更具体，本节将集中讨论为简单的字处理应用生成说明的任务。我们假定：无论该知识库的内在结构如何，它都可以被看成是 KLONE 风格的知识库。我们也假定：交际目的是给系统的新用户解释那些已经被表示的过程。知识库将把保存一个文件（saving a file）的过程表示为一个简单的程序性的层级，如图 20.5 所示。这里指定的过程要求用户从文件菜单中选择保存选项（choose the save option），选择正确的文件夹（folder）以及文件名称（file name），然后单击保存按钮（click on the save button）。作为附带效应（side-effect），系统为响应正确的用户行为自动显示（display）并关闭（close）保存对话框。这个陈述给出了基本行为之间的程序上的关系，但它并没有说明与界面结构有关的任何领域的知识（比如，哪个选择对应于哪个菜单）或用于该过程的特定实体（比如文件或用户等）。我们也将假定这些都是可以从知识库中获取的。

20.4.1　文本说明图

除了本章开头讨论过的严格结构化的罐装文本和槽填充模板以外，生成文本的最简单方法是将文本结构与输入知识库的结构连接起来。例如，我们可以通过以序列的方式回顾 tic-tac-toe 或

棋子游戏进行的步骤而描述该游戏。然而，当我们面对为达到各种交际目的而表示的大量信息时，这个策略会很快失效。例如，包含图20.5所示的片断的知识库，就可能是一系列指令（例如指导手册中的所有指令）的表示，或一个按字母排序的程序函数集（例如参考手册中的所有函数）的表示。

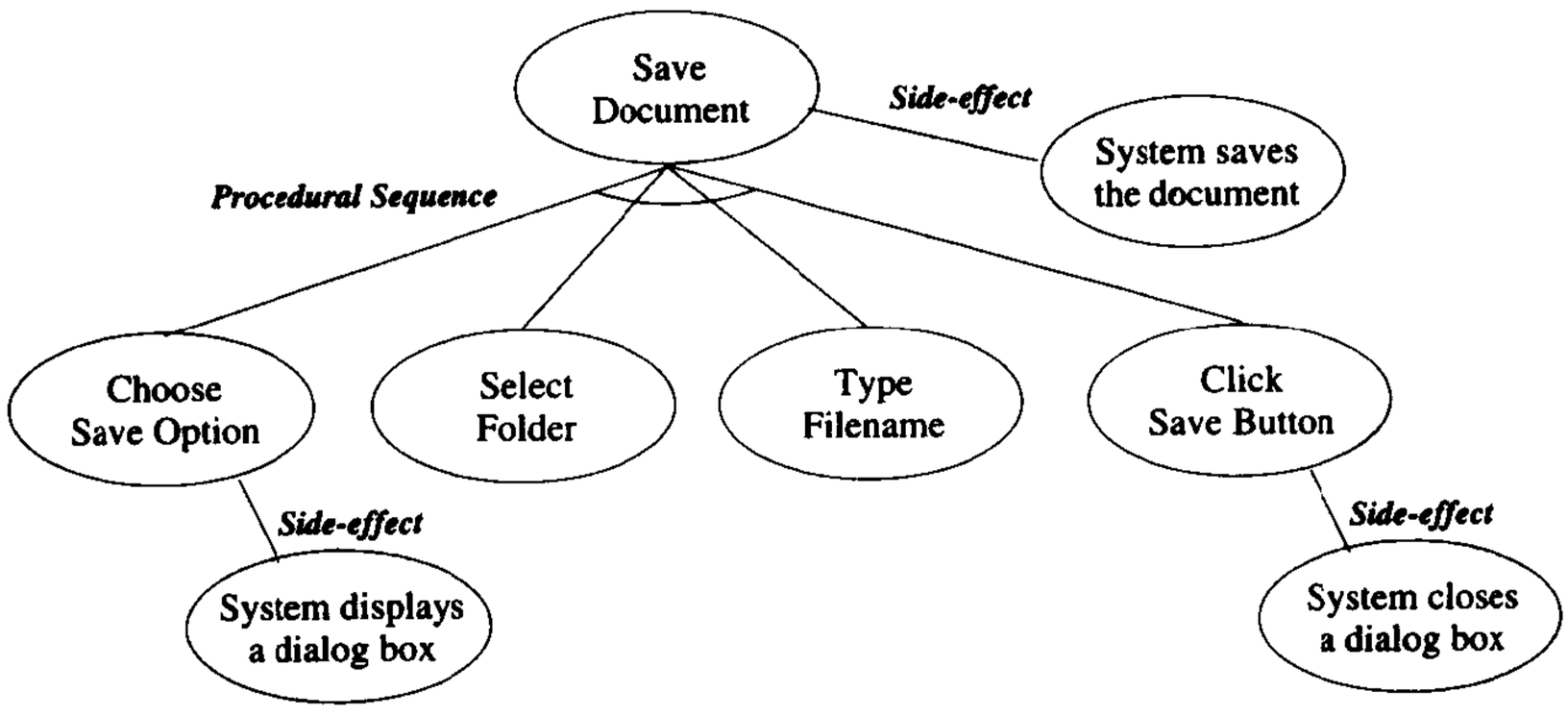


图20.5 保存过程知识库的一部分

对这个问题的一种处理方法基于这样的观察：文本倾向于遵循一致的结构模式。例如，撰写指令解释如何完成一个行为的典型方式是：将所需的行为按照它们执行的顺序来表示。这些行为的任何前提都需要在相关行为之前被提及。类似地，这些行为产生的效应则在相关行为之后被提及。在某些领域中，这种模式很少被打破。配备了这个信息，就能够建立一个表示该结构的说明图（schema），如图20.6所示。这个说明图被表示为扩充转移网络（augmented transition network，简称ATN），每个结点是一个状态，每个弧线是一个可选的转移（见第10章）。控制从图中左上部的黑色结点开始，沿着弧线前行直到执行终止于左下部的终结点。结点S0容许对任意数目的前提进行表达。转移至S1促使对该行为本身进行表达。S1容许递归调用网络以表达任何子步骤。转移至S2不需要任何行为，并且S2容许在执行停止之前对任何数目的附带效应进行表达。

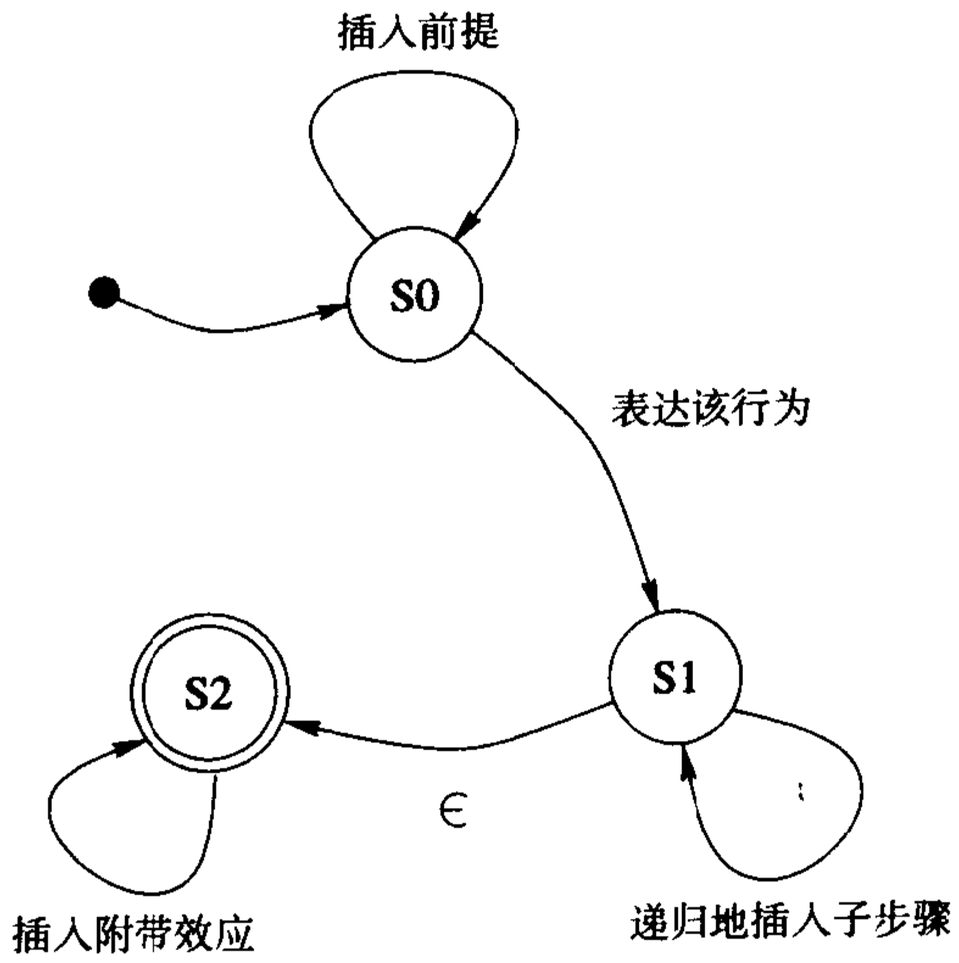


图20.6 用于表达处理过程的说明图

我们可以利用这个说明图对图 20.5 所示的样例过程的表达进行规划。当询问系统如何保存一个文件时，过程说明图就被激活了。我们假定知识库对保存一个文件的行为没有指定任何前提，因此直接前行至状态 S1，促使主要行为，即“保存文件”的表达。在状态 S2，我们为输入指定的 4 个子步骤中的每个子步骤递归地调用该网络。这将给出第一个子步骤“选择保存选项”的表达，以及它的附加效应，“这导致系统显示出保存对话框”。第一个子步骤没有前提或子步骤。其他子步骤都是以相同方式进行的，并且最终回到在状态 S2 执行的主行为，状态 S2 表示整个处理的结果，“这将导致系统保存文件”，然后终止。基于这样的规划的细节，最后就可以生成下面的文本：

Save the document: First, choose the save option from the file menu. This causes the system to display the Save-As dialog box. Next, choose the destination folder and type the filename. Finally, press the save button. This causes the system to save the document.

（保存文件：首先，从文件菜单选择保存选项。这会导致系统显示出保存对话框。接着，选择目标文件夹和文件的类型。最后，按下保存按钮。这将导致系统保存该文件。）

这些句子中的每一个都可以采用上一节讨论过的表层实现器中的一种来生成。我们可以看出，说明图的机制与模板或罐装文本的方法相比更灵活。它根据表达的已知模式对输出进行结构化，加入恰当的约束，就能够以各种顺序插入从知识库收集来的可选素材。另外，它不要求在知识库中对所有事物进行表示。例如，在这个知识库中不包括“单击保存按钮”行为的附带效应。

这个说明图机制生成的仅仅是一个较高层次的话语结构。其中，每个句子的细节形式的指定问题常常被称为微规划（microplanning），20.5 节将讨论有关内容。

20.4.2　修辞关系

倘若关于表达的一致模式的离散集可以找到并可以编码，则对话语规划而言，说明图是非常有用的。然而，说明图存在两个基本的不足。首先，当正在生成的文本需要更多的结构变化以及更丰富的表达时，说明图就会变得不可行。例如，某些条件可能规定我们以不同的方式来安排过程说明。某些上下文可能规定我们明确地举出过程中的某些步骤，或规定我们以不同的方式或不同的顺序表示文本中的某些片断。尽管原则上这些变化既可以通过为说明图添加约束和运算，也可以通过添加新的说明图来支持，但是要求的变化越多，基于说明图的方法就变得越困难。

基于说明图的机制的第二个问题是：它们产生的话语结构是一个句子生成请求的简单序列。它没能把与句子相关的高层次结构包括进来。在某些领域，特别是交互式领域中（见第 19 章），前面话语的结构是与未来的规划相关的。例如，我们已经详细解释了某个处理，也许就不愿意再重复。当这里存在一个关于前面话语结构的记录时，这些处理就会变得更容易。

这里的一个有用方法是：为了发现文本中起作用的更基本的修辞动态而观察说明图潜藏的本质。拥有这些动态系统可以基于它所面对的情形而建立自己的说明图。正如第 18 章的一些详细讨论，人们已经提出了许多试图对这些修辞动态进行形式化的理论。修辞结构理论（Rhetorical Structure Theory，简称为 RST）是这些理论中的一个，它是一种基于文本局部之间关系的关于文本组织的描述理论(Mann and Thompson, 1987)。例如，研究下面的两段文本：

I love to collect classic automobiles. My favorite car is my 1899 Duryea.　　(20.4)

I love to collect classic automobiles. My favorite car is my 1999 Toyota.　　(20.5)

第一段话有意义。作者喜欢 1899 年的 Duryea 的事实很自然地紧接着他喜欢古典汽车的事实。而第二段话是有缺陷的。该缺陷并不是单个句子的问题，它们单独看起来都是完美的。缺陷在于它

们的结合。两个句子顺序排列的事实暗示它们之间具有某种连贯关系。对于第一段文本来说，这种关系具有详述（elaboration）关系（见第19章）的特征。第二段文本具有对照（contrast）关系的特征，因此更恰当的表示是：

I love to collect classic automobiles. However, my favorite car is my 1999 Toyota. （20.6）

其中，However明显地将对照关系的信号传递给读者。RST总共给出了23个修辞关系，包括ELABORATION（详述）和CONTRAST（对照），足够描述各种文本的修辞结构。在实践中，研究人员倾向于从这些修辞关系中选出适合各自的应用领域的子集。

大部分RST关系将一段文本的中心片断（“I love to collect ...”）称为核心（nucleus），而将文本的周边片断（“My favorite car is ...”）称为外围（satellite）。这揭示出了这样一个事实：许多修辞关系是非对称的。第二个句子是根据第一个句子来解释的，但反之则不然。下面将看到并不是所有修辞关系都是非对称的。RST关系是根据它们施加于核心、外围以及核心和外围的结合处的约束来定义的。这里是一些常用的RST关系的定义：

详述（ELABORATION） 外围给出的是与核心内容有关的一些额外细节。该细节的可能形式有：

- 已知集的一个成员
- 已知抽象类的一个实例
- 已知整体的一个部分
- 已知过程的一个步骤
- 已知客体的一个属性
- 已知普遍化原则的一个特定实例

对照（CONTRAST） 核心给出的事物尽管在某些方面具有相似性，但是在某些重要方面又是不同的。这种关系具有多个核心（multi-nuclear），因为它并不区分核心和外围。

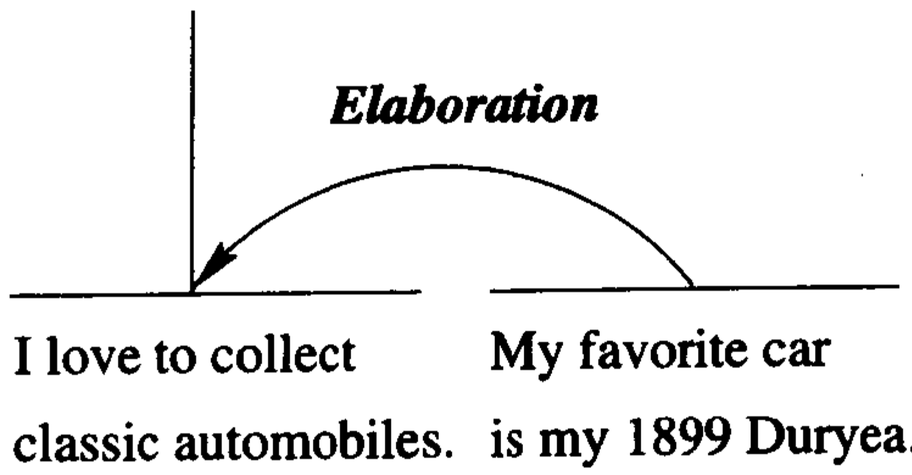
条件（CONDITION） 外围给出的某些事件必须在核心给出的情形出现之前就已经发生。

目的（PURPOSE） 外围给出的是实施核心所表示的行为的目的。

顺序（SEQUENCE） 这种关系是多个核心的。核心集被连续地实现。

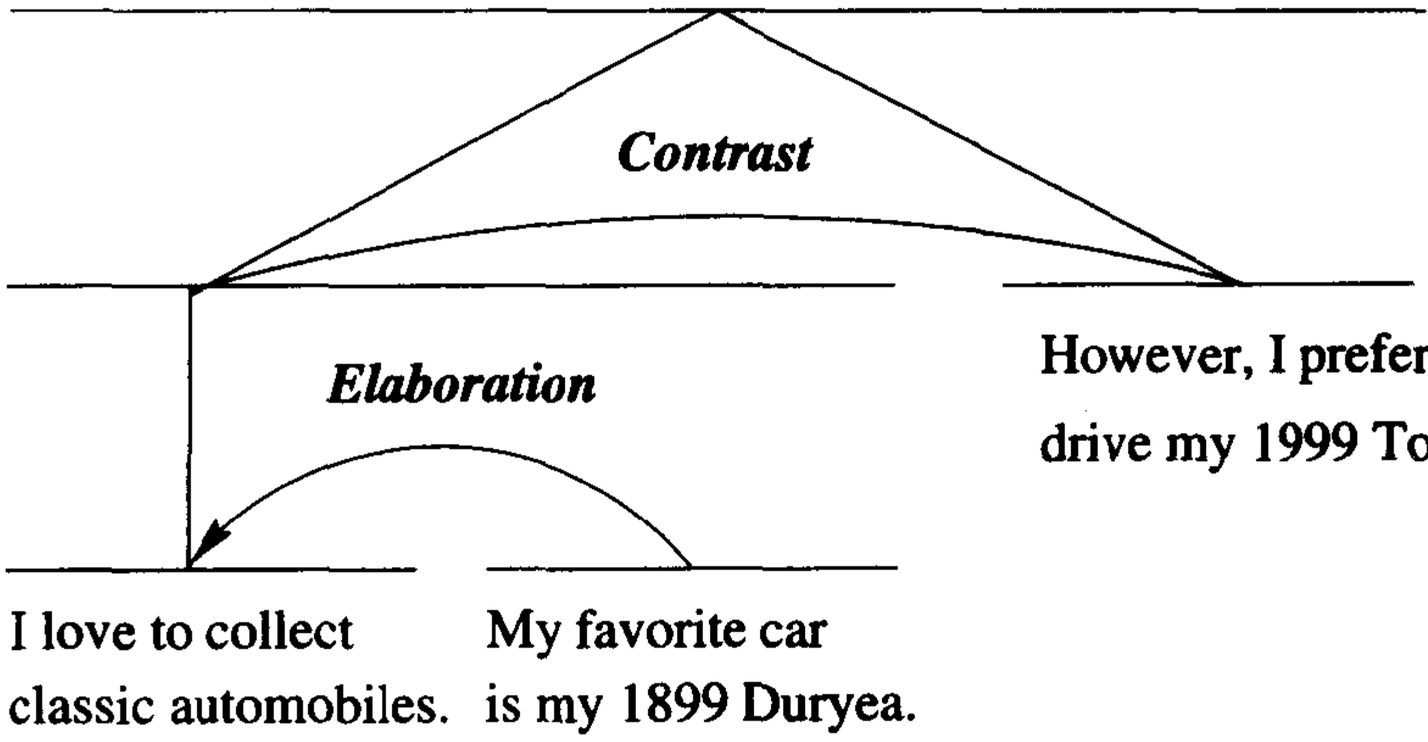
结果（RESULT） 核心中给出的情形是由外围中给出的情形产生的。

RST关系被表示为下面的典型图形：



这是例句（20.4）中修辞关系的图形表示。文本片断沿图形的底部顺序排列，在文本上所建的是修辞关系。单个文本片断通常是从句。

修辞结构分析是层级地进行组合的，因此可以采用一对相关文本作为其他高层关系的外围或核心。研究下面的三个句子的结构：



其中，前两个从句是通过详述关系而相关的，并且它们作为一个整体通过对照关系与第三个从句相关，同时也要注意多个核心的对照关系是如何描述的。这类递归结构容许 RST 为扩充的文本建立一个单一的分析树。

尽管最初提出 RST 是作为一个描述工具，但它也可以作为 NLG 的构建工具来使用。为了实现这一目的，修辞关系常常被重新塑造为一个 AI 风格的规划器的运算。作为例子，我们将考察一个可以用于基于修辞的文本规划的、通用目的的、自顶向下的、层级性的规划器[①]。

这类规划器的基本方法是为生成系统给出一个高层次的交际目的，该目的是按照这个文本传递给读者的效应来陈述的。对于这个说明文本的例子，要求这个规划器建立一个能够实现该目的（即让读者学会保存文件）的结构。实现该目的的最高层规划运算将插入一个适合该目的的修辞结点，并为该修辞关系的核心和外围插入一些子目标。这些子目标将被递归扩展，直到规划处理到达该修辞结构树的底部。插入一个结点可以被表示为一个简单的从句。

对于我们的例子，将给出下面的目的：

(COMPETENT hearer (DO-ACTION <some-action>))

这里，交际目的是使听话人能够完成某个行为。该行为将被表示为知识库中的一个状态，在本例中，交际目的为图 20.5 所示的过程层级的根结点。实现该目的的文本规划运算如下所示：

Name: Expand Purpose
Effect:
(COMPETENT hearer(DO-ACTION ?action))
Constraints:
(AND
(c-get-all-substeps ?action ?sub-actions)
(NOT (singular-list? ?sub-actions))
Nucleus:
(COMPETENT hearer(DO-SEQUENCE ?sub-actions))
Satellites:
(((RST-PURPOSE(INFORM s hearer(DO ?action)))
required))

这个规划操作的基本思想是通过解释如何进行一个特定行为的子步骤（?substeps）来解释如何进行该行为（?action）。注意上表中的 Effect（效应）域与前面给出的目的是相同的。只有当一个运算的

① 这个文本规划器改编自 Moore and Paris(1993)的工作。

约束被满足时，它才能被应用。在本例中，主行为（?action）必须具有多个子行为。对于当前的例子（见图20.5），这一点是满足的，运算在话语结构中插入一个修辞目的结点以及它的外围和核心的目的说明。外围告诉听话人实施主行为的目的，而核心列出了实现该行为所需的子行为。注意，运算的效应域、约束域、核心域和外围域都采用了变量（以“?”开始的标识符），在运算被应用时这些变量也将被统一。因此，目的行为是被绑定于“?action”的，并在整个随后的规划运算中都可以被访问。

规划运算的另一个值得注意的问题是实现内容选择的方式。约束域指定必须有子步骤并且必须有多个子步骤。为了确定是否满足第一个约束，需要系统从知识库中查询子步骤。接着，这些子步骤被用做构建的核心结点的内容。因此，当话语规划处理给出请求时，规划运算本身实现了内容选择。

图 20.7 给出的是由规划器生成的全部文本结构。这个树的根结点（即最顶端的水平线）是由前面的规划运算生成的结点。图20.7的第一个核心结点是包括所有子行为的多核心结点。生成该结点的规划运算如下所示：

Name: Expand Sub-Actions
Effect:
(COMPETENT hearer (DO-SEQUENCE ?actions))
Constraints:
NIL
Nucleus:
(foreach ?actions(RST-SEQUENCE
(COMPETENT hearer (DO-ACTION ?actions))))
Satellites:
NIL

该运算实现了由前面运算给出的核心目标。它给出了一个带有多个核心的修辞结点（每个核心对应于一个为实现主要目的而必须的子行为）。采用适当的规划运算集，该规划系统能够产生如图 20.7 所示的话语结构，接着该话语结构将被线性化为下面的文本：

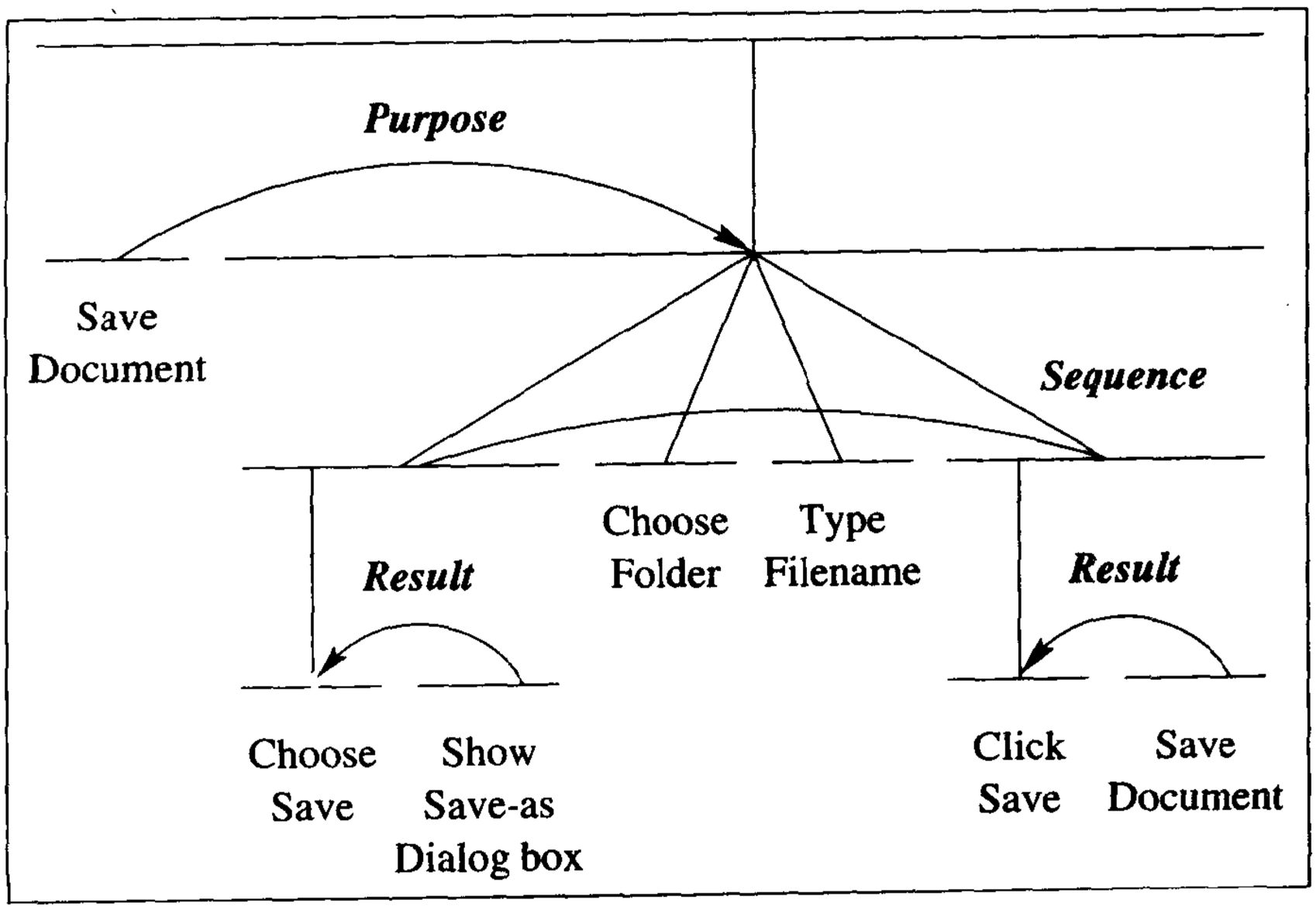


图 20.7 样例文本的全部修辞结构

To save a new file（保存新文件）

1. Choose the save option from the file menu.（从文件菜单选择保存选项。）
   The system will display the Save-file dialog box.（系统显示出保存对话框。）
2. Choose the folder.（选择文件夹。）
3. Type the file name.（键入文件名。）
4. Click the save button.（单击保存按钮。）

The system will save the document.（系统保存该文件。）

所有这些句子都可以通过一个表层实现器来生成。特别是最后一个句子与前面给出的例句（20.2）相同。正如在讨论基于说明图的话语规划时提及的，微规划问题被推迟到 20.5 节中叙述。

20.4.3　小结

本节讲述基于说明图的机制如何才能利用话语结构的一致模式的优点。尽管该方法在许多背景下已经被证实是有效的，但是它的灵活性还不足以处理更加多变的生成任务。而我们介绍的基于修辞关系的话语规划可以更灵活地处理这类任务。

20.5　其他问题

本节将介绍在前面几节没有仔细讨论的一些问题。

20.5.1　微规划

前面几节的确没有详细介绍从例子所描述的话语规划到表层实现器之间的映射过程。话语结构如图20.7所示，指定了文本的高层或宏观结构，但是几乎没有指定试图作为表层实现器输入的任何细节。这个更详细的规划问题被称为微规划（microplanning）。

在大部分生成应用中，微规划是简单的固定连接。例如，在说明生成系统中，可以在所有情形下采用同样的方式指向客体，用户行为可以被表示为独立的祈使句。这将使问题大大简化，但是容易产生如例句（20.3）所示的单调文本。以上说明微规划与两个基本领域有关：所指表达（referring expression）和集结（aggregation）。

所指表达的规划需要确定：当我们在一个特别的背景下指向一个实体时，应该采用该实体的哪些方面。如果该客体是讨论的焦点并且刚刚被提及，就可以采用简单的it，而如果要引入一个新的实体，就可能需要更精致的表示，例如 *a new document to hold your term paper*。这些问题在第 18 章中曾经详细讨论过。

集结是从知识库向短语、从句和句子长度的词块进行内容分配的问题。我们在引言中给出了一个这样的例子，例句（20.1）提及的两个行为被合并于一个从句：*you've just compiled and run a simple C program*。它的可读性好于例句（20.3）给出的不集结的版本（*You've just compiled a simple C program. You've just run a simple C program*）。

微规划常常被看成是放置于话语规划器和表层实现器之间的中介的流水线模块（见图 20.1）(Reiter and Dale, 2000)。而最近的研究工作强调了这样一种观点：微规划被看成是一种与话语规划和表层实现同等重要的任务。为实施微规划任务而将规划运算添加于本章所描述的基于RST的规划机制也是可能的。不管用何种方法实现微规划，它的目的都是建立从话语规划器的输出到表层实现器的输入的映射。

20.5.2 词汇选择

词汇选择是指挑选恰当的单词以表达选定内容的一般性问题。本章所讨论的表层实现器在需要封闭词类的词汇项时可以将其直接插入，但是对实词的选择将推迟给话语规划器。许多规划器都通过把单个词汇项与知识库中的每个实体联系起来的办法来简化这个问题。

以原则性的方式操作词汇选择需要生成系统处理好两个问题。首先，当存在多种选择时，生成系统必须能够选出正确的词汇项。例如，在上一节的文件保存文本中，系统生成了 Click the save button。对于词汇项 click，有多种可供选择的选项，包括 hit 和 press mouse left on 等。这些选项之间的选择需要考虑到：(1) 风格，在本例中 hit 也许是 click 的一种较不正式的表达方式；(2) 搭配，在本例中 click 更可能与 button 同现于该领域；(3) 用户知识，在本例中计算机的初学者也许需要更确切地指定 press mouse left on。

其次，生成系统必须能够选择概念表达的正确语法形式。例如，系统可以将该片断命名为 Saving a new file 而不是 To save a new file。这种在分词和不定式形式之间的选择常常要根据其中的哪个是说明语料库中最常采用的形式来决定取舍。

20.5.3 生成系统评价

在早期的 NLG 研究中，系统输出的品质都是由系统开发者自己来评估的。如果一个输出看起来不错，系统就被判断为成功。对系统品质而言，这不是一个非常有效的测试，最近有许多研究都关注于 NLG 系统的准确评测。

有一种技术，以统计方式比较生成器的输出与目标文本语料库的特征。如果生成器选择的形式是该语料库中最常用的形式，则判定为正确。这种方法的危险之处在于语料库中的文本通常是由作家写成的，而作家可能会出错，因此可能导致语料统计出现偏差。这里的假设，就像 Tolstoy（托尔斯泰）在 Anna Karenina（安娜卡列尼娜）中所说的话，“幸福的家庭是相似的，而不幸的家庭各有各的不幸”。换言之，好的文本会表现出不断出现的一致特征集，而不好的文本会表现出一些个别的特征，从统计角度看，这些特征不会累积。

另一种技术是召集一个专家小组，对生成器的输出与人类作者写出的文本进行比较。这是 Turing 测试的一种变体，专家并不知道哪些文本是由系统生成的，哪些是由人类作者写出的。计算机生成的文本的得分通常低于人类写出的文本，不过这些文本的品质接近于人类作者在某些限定领域写出的文本。

最后，用于判断这些被生成的文本是否有效的一种技术是看它的目的是否实现了。例如，如果该文本是用于描述某些客体的，它的品质可以根据读者在读完输出文本之后完成的内容测验的得分来测定。如果该文本是用于解释如何进行某些处理的，则它的品质就可以根据读者在读完输出文本之后实施该处理时出现的错误数来测定。

20.5.4 语音生成

本章重点讲述了文本生成而不是语音生成。然而，在许多情形下语音输出即使不是绝对必须也是更可取的。这些情形包括无法显示文本，比如当用户使用电话时，以及用户不能看文本的显示，比如当用户正在驾驶汽车或用户是残疾人时。

一个简单的方法是将生成系统产生的单词串传递给第 4 章、第 5 章和第 7 章所描述过的文本 - 语音合成系统。这种方法需要面对的是曾在 4.6 节和 16.1 节讨论过的一个问题：文本 - 语音转换系统必须能够处理同形词（homograph）（即拼写相同但读音不同的单词）。研究下面的例子：

Articulate people can clearly *articulate* the issues.　　(20.7)
(发音清晰的人能够清楚明白地说出这些问题。)

其中，拼写 articulate 的两个例子必须采用不同的发音。另一个问题是韵律（prosody）的处理，这需要为正在生成的语音指派恰当的音高曲拱和重音模式。

这种简单化的方法需要文本-语音转换系统通过分析输入文本来解决上述两个问题。同形词常常可以通过词性标注（articulate 的形容词和动词形式的发音不同）或第 17 章的词义排歧算法来进行区分。第4章所提及的自动生成韵律是一个非常难的问题。一些韵律信息可以通过区分疑问句与非疑问句以及通过观察逗号和句号来获得。然而，通常很不容易从输入文本中抽取我们所需要的这些信息。

这种简单化方法的一种替代方法是将NLG系统的丰富的表示传递给语音合成系统。一个典型的NLG系统包括准备生成单词的语义和词性信息，并且能够用这些信息标注该单词，以帮助选择正确的单词发音。这些系统也能够标注话语结构信息，以帮助合成正确的韵律。到目前为止，NLG的研究还几乎没有涉及这样的领域。

20.6　小结

语言生成是指从非语言输入构造自然语言输出的处理。作为一个研究领域，它通常并不包括对较简单的生成机制，比如罐装文本（canned text）和模板填充（template filling）的研究。

- 语言生成与语言理解的不同在于语言生成最关注的是语言选择（choice），而不是排歧。生成中的选择问题包括内容选择（content selection）、词汇选择（lexical selection）、集结（aggregation）、所指表达生成（referring expression generation）以及话语结构（discourse structuring）等。
- 语言生成系统包括一个规划话语结构的组件，称为话语规划器（discourse planner），以及一个生成单个句子的组件，称为表层实现器（surface realizer）。话语规划的方法包括文本说明图（text schemata）和修辞关系规划（rhetorical relation planning）。表层实现的方法包括系统语法（systemic grammar）和功能合一语法（functional unification grammar）。
- 微规划（microplanning）给出了从话语规划器的输出到表层实现器的输入之间的映射，包括所指表达（referring expression）生成、集结（aggregation）和词汇选择（lexical selection）中更细粒度的任务。

20.7　文献和历史说明

如果不将罐装文本和模板填充机制计算在内，自然语言生成领域相对于自然语言处理的其他领域而言是较年轻的。对该领域的一些平淡无奇的、不起眼的试验出现于20世纪50年代和60年代，并且大部分是在机器翻译的背景下进行的，对生成的集中研究直到20世纪70年代才出现。Simmons and Slocum(1972)的系统采用ATN从语义网络生成话语。Goldman的BABEL(1975)采用决策网来实现词汇选择，Davey 的PROTEUS(1979)生成了对 tic-tac-toe 游戏的描述。在 20 世纪 80 年代，生成被确立为一个独立的研究领域。McDonald(1980)和 PENMAN 项目(Mann，1983)对表层实现，McKeown(1985)和 Appelt(1985)对文本规划，都具有突出的贡献。20世纪90年代，随着面向生成的一些欧洲和国际会议以及组织(见Special Interest Group on language GENeration，http://www.aclweb.

org/siggen）的出现，人们对生成的兴趣持续增长。Kukich(1988)和Reiter and Dale(2000)对罐装文本和模板机制的使用以及缺陷进行了讨论。

在撰写本书时还没有讲述生成的书籍。但是，一本有关应用生成的教材已在印刷(Reiter and Dale，2000)，也有不少综述性的论文(Dale et al., 1998a; Uszkoreit, 1996; McDonald, 1992; Bateman and Hovy, 1992; McKeown and Swartout, 1988)，其中不少文章都涉及NLG的发展史以及NLG与语言处理的其他领域的关系。McDonald(1992)介绍了假设管理与选择之间的区别。

生成的典型体系结构是由规划和实现任务组成的流水线。流水线被用于约束每个模块的搜索空间，因此使得生成任务更可控(Reiter and Dale, 2000; McDonald, 1988; Thompson, 1977)。但是，这种体系结构也存在一个很明显的缺陷：话语规划的决策几乎不可能被实现器撤销(Meteer, 1992)。Applelt的KAMP(1985)基于AI规划来规划和实现一体化的体系结构。然而，已经证实这种方法无法用于较大规模领域的计算。人们也提出了用于生存系统的黑板体系结构(Nirenburg et al., 1989)。对微规划本身的各种关注引起了极大兴趣，包括对所指表达(Dale, 1992; Appelt, 1985)、集结(Dalianis, 1999; Mann and Moore, 1981)以及其他语法问题(Vander Linden and Martin, 1995; Meteer, 1992)的研究。相关的问题包括词汇选择(Stede, 1998; Reiter, 1990; Goldman, 1975)和为特殊听众裁剪输出(Paris, 1993; Hovy, 1988a)等也得到了关注。

在20世纪80年代晚期和90年代初期，我们看到了几个可重用的NLG系统的构建，包括两个已公开发布的系统：KPML(Bateman, 1997)和FUF(Elhadad, 1993)。这些工具可以从SIGGEN的网站下载。大部分工作采用的是Lisp语言，不过最近也有一些将这些系统移植到其他语言或平台的尝试。

系统功能语言学（Systemic functional linguistics，简称SFL）是由Halliday(1985)开创的。它基本上独立于生成语言学并且在整个语言处理学术界相对来说是不知名的。将它用于剖析的尝试只取得了有限的成功(O'Donnell, 1994; Kasper, 1988)。但是，它对于NLG有深远的影响，通过一两种形式用于许多生成系统，包括Winograd的SHRDLU(1972a)，Davey的PROTEUS，Patten的SLANG(1988)，PENMAN(Mann, 1983)，FUF(Elhadad, 1993)以及ILEX(Dale et al., 1998b)。本章介绍的系统语法的样例部分基于Winograd(1972a)的讨论。SFL的最具竞争力的计算实现是Komet-Penman MultiLingual development environment（KPML），它源自PENMAN。KPML与NIGEL（一种大规模的英语生成语法）以及开发多语言语法的环境一起被推出。其中也包括一个句子规划语言（Sentence Planning Language，简称SPL），它形成了比系统语法本身更便于使用的界面。与在系统网络中做出选择时所需包括的所有信息的说明相比，建立SPL说明简单得多，但是比本章给出的spl构造器的样例更灵活。我们来研究下面的SPL说明：

```
(s1 / save
    :actor (a1 / system
               :determiner the)
    :actee (a2 / document
               :determiner the)
    :tense future
    )
```

SPL解释器将这个说明扩展到生成例句（20.2）（*The system will save the document.*）的Nigel语法所需的一系列特征选择。说明中的每一项给出了实体的角色（比如*actor*，*actee*）以及语义类型（比如*save*，*system*和*document*）。这种语言类型是KLONE风格的概念，从属于一个称为上部模型(Bateman et al., 1990)的通用概念知识库（见第16章）。该知识库表示具有语义因果关系的语义特性，

被SPL用于确定被表达实体的类型，因此减少了直接包含于SPL说明中的信息数量。在这个例子中省略了本章例子曾包含的:speechact assertion项，因为SPL将其作为没有特别指定时的默认值。

功能合一语法是由 Kay(1979)开创的，详见第 11 章。它的用于生成的最有影响力的实现是由 Elhadad(Elhadad, 1993, 1992)开发的 Functional Unification Formalism（FUF）。它是与英语语法 SURGE一起发布的。虽然本章给出的例子采用了语法范畴(Elhadad, 1992)的简单短语结构方法，但是 SURGE 语法采用的是系统范畴。

另一个用于语言生成的有影响的语言学理论是 Mel'čuk(1988)的意义文本理论（Meaning Text Theory，简称 MTT）。MTT 假定了从深层句法直至表层结构的许多层次。采用它的表层实现器，包括 CoGenTex 的 REALPRO(Lavoie and Rambow, 1997)以及 ERLI 的 AlethGen(Coch, 1996b)，从深层开始逐层映射，直到到达表面层。

话语生成从一开始就受到 NLG 的关注。例如，Davey 的 PROTEUS，为 tic-tac-toe 游戏生成了段落长度的总结。他的系统几乎完全基于应用系统记录的游戏踪迹结构来构造输出。McKeown (1985)是基于说明图的文本构造的先驱，这种方法更灵活并且已得到许多应用(Milosavljevic, 1997; Paris, 1993; McCoy, 1985)。本章给出的基于说明图的样例基于 COMET 说明生成系统(McKeown et al., 1990)。尽管话语结构的其他理论（见第 18 章）也对 NLG 产生了影响，包括 Grosz and Sidner (1986)和 Hobbs(1979a)等提出的理论以及 Kamp(1981)的 DRT 理论，但是最具影响力的是由 Mann and Thompson(1987)开创的修辞结构理论(Rhetorical Structure Theory，简称RST)(Marcu, 1998; Scott and Souza, 1990; Hovy, 1988b)。本章给出的关于古典汽车的例子改编自 Mann and Thompson(1986)，基于 RST 的例子基于 Moore 和 Pair 的文本规划器(Moore and Paris, 1993)，它也被用于 DRAFTER (Paris and Vander Linden, 1996; Paris et al., 1995)，ISOLDE(Paris et al., 1998)以及WIP(Wahlster et al., 1993)等项目。在Moore and Paris(1993)中描述了将该规划器用于交互式对话系统背景的情形。最近 Marcu(1998)提出了不同的方法来替代这种方法。

NLG 应用倾向于集中在一个相对受限的子语言领域（见第 21 章），包括天气预报(Coch, 1998; Goldberg et al., 1994)、说明书(Paris et al., 1998; Paris and Vander Linden, 1996; Wahlster et al., 1993)、百科全书式的描述(Milosavljevic, 1997; Dale et al., 1998b)以及信件(Reiter et al., 1999)。输出可以是简单的文本或超文本(Lavoie et al., 1997; Paris and Vander Linden, 1996)，动态生成的超文本(Dale et al., 1998b)，多媒体陈述(Wahlster et al., 1993)以及语音(Van Deemter and Odijk, 1997)等。在SIGGEN 的网页上可以看到这些系统的详细信息。

NLG 系统的评测最近颇受关注。评测可以通过几种方法进行，包括评估输出与有代表性的语料库的相似性(Yeh and Mellish, 1997; Vander Linden and Martin, 1995)，召集专家小组判定输出文本(Lester and Porter, 1997; Coch, 1996a)，以及测试文本实现其交际目的的有效性(Reiter et al., 1999)。对 NLG 系统本身的有用性的评测也变得越来越普遍。

NLG 中受关注的其他问题包括连接主义（connectionist）和统计技术的使用(Langkilde and Knight, 1998; Ward, 1994)，以及多语言生成系统作为机器翻译的替代品的可行性等(Hartley and Paris, 1997; Goldberg et al., 1994)。

第21章　机器翻译[①]

"...Translation is a fine and exacting art, but there is much about it that is mechanical and routine."

Kay (1997)[②]

本章介绍机器翻译（machine translation，简称MT）技术。机器翻译是指利用计算机全自动或部分自动地将一种语言翻译为另一种语言的处理技术。从概括的角度看，翻译中充满艰辛却又令人神往，它与任何其他需要人类丰富创造力的领域一样，都需要耗费大量的人力。研究下面摘自18世纪曹雪芹的小说《石头记》（又名《红楼梦》）(Cao, 1792)中第四十五回末尾处的一段文字，标注了汉语拼音，英文翻译来自David Hawkes：

"黛玉自在床上感念宝钗[③]……
又听见窗外竹梢焦叶之上，雨声淅沥，清寒透幕，不觉又滴下泪来。"

As she lay there alone, Dai-yu's thoughts turned to Bao-chai... Then she listened to the insistent rustle of the rain on the bamboos and plantains outside her window. The coldness penetrated the curtains of her bed. Almost without noticing it she had begun to cry.

dai yu	*zi*	*zai*	*chuang*	*shang*	*gan nian*	*bao chai...*
Dai-yu	alone	on	bed	top	think-of-with-gratitude	Bao-chai

you	*ting jian*	*chuang*	*wai*	*zhu*	*shao*	*xiang*	*ye*	*zhe*
again	listen to	window	outside	bamboo	tip	plantain	leaf	of

shang,	*yu*	*sheng*	*xi*	*li,*	*qing*	*han*	*tou*	*mu,*
on-top,	rain	sound	sigh	drip,	clear	cold	penetrate	curtain,

bu	*jue*	*you*	*di*	*xia*	*lei*	*lai.*
not	feeling	again	fall	down	tears	come.

思考一下在这种文学翻译中需要涉及哪些问题。首先，由于曹雪芹常常采用双关语给人物命名而引起了中文人名的翻译问题。Hawkes的选择是对其中的主要人物都采用构成人名的汉字的音译，而对那些仆人的名字则采用意译，例如Aroma（袭人）和Skybright（晴雯）。其次，中文几乎没有动词时态和语态的变化，因此Hawkes不得不决定将中文"透"翻译为*penetrated*，而不是*was penetrating*或*had penetrated*。Hawkes还添加了物主代词*her*，使得*her window*相比*the window*更适合那种安静闲适的卧室气氛。为了使不熟悉中国床帷的英文读者能清楚地理解，Hawkes将"幕"翻译为*curtains of her bed*。最后，短语"竹梢焦叶"的中文很优雅，这种四字短语是有文化品位的标志，但是如果以词对词的方式翻译为英文，则很糟糕，因此Hawkes只是简单地将它翻译为*bamboos and plantains*。

显然这类翻译要求对源语言和输入文本具备博大精深的理解，同时也需要能够老练地、富有诗意地、创造性地支配目标语。因此，将任意文本自动从一种语言生成另一种语言的高质量的翻译问题是难以自动实现的。但是，当前的计算模型对一些较简单的翻译任务还是可以胜任的。例如，机

---

① 本章的主要作者是Nigel Ward。

② 译文：翻译是一门精致而严谨的艺术，但其中也不乏机械和常规的机制。

③ 原文为"黛玉自在枕上感念宝钗……"——译者注

器翻译系统的应用常常集中于：(1) 那些粗略翻译就足够的任务；(2) 那些人工编辑后可用于提高 MT 输出的任务；(3) 能够产生高质量译文的受限子语言领域的任务。

网络中的信息的获取是一种“信息采集”的任务，只要有非常粗略的翻译，读者就会感到满足。研究下面摘自法文网页和机器翻译的文本：

> Nous sommes une association type Loi de 1901, et notre raison d'être est de practiquer , de promouvoir, de faire découvrir le Paintball, et le cas échéant de supporter nos equipès de compétition: ...Si vous avez des questions, des envies d'organisatioin de parties, des envies de jouer tout courte et des envies de découvrir, n'hésitez pas à nous contacter par courrier ou par téléphone ou bien encore par eMail. ... Au sortir de la saison 97/98 et surtout au début de cette saison 98/99, les effectifs des HORS-TAXE sont modifiés.
>
> We are a standard association Loi of 1901, and our raison d'ecirc;tre① is to practice, promote, make discover Paintball, and to support our teams of competition if necessary:...If you have questions, desires of organization of parts, desires for playing very short and desires for discovering, do not hesitate to contact us by mail or telephone or even by eMail. ... With leaving season 97/98 and especially at the beginning of this season 98/99, manpower of the HORS-TAXE are modified!

从翻译中可以领会到这是一个彩弹球（paintball）队的主页，似乎很友好，也许他们愿意接收新会员。有了这些信息，我们接着就可以找人给我们进行正确的翻译，或者可以直接给组织者发一封电子邮件，问问可否参加。顺便说一下，将 MT 用于文本查询的这种用法有些可以通过更有效的跨语言信息检索（cross-language information retrieval）技术来实现，这种技术所关注的是如何采用一种语言的检索词来查询另一种语言的文本(Oard, 1997)。

粗略翻译也可以作为一个完整翻译过程的第一个阶段。MT 系统可以生成翻译草稿，然后由翻译工作者进行译后编辑（post-editing）处理。即使是一个草稿，有时也会加快整个翻译处理的进程。严格地说，采用这种方式的系统是机助人译（computer-aided human translation，简称为 CAHT 或 CAT）而不是（全自动）机器翻译。MT 的这种使用模式特别适合于翻译工作量大又需要快速更新的领域。我们最熟悉的例子是软件手册的本地化（localization）翻译。另一个例子是关于市场变化的财经新闻的翻译，例如股票经纪人使用的从日语到英语的翻译。

天气预报是子语言（sublanguage）领域的一个实例，在该领域的 MT 模型足以产生可以直接使用的原始MT输出，甚至不用做任何后编辑。天气预报包括的短语有：Cloudy with a chance of showers today and Thursday（今天和周四多云转阵雨），Low tonight 4（今晚最低 4 摄氏度），high Thursday 10（周四最高 10 摄氏度）以及 Outlook for Friday: Sunny（展望周五：晴）。该领域只有有限个词汇和一些基本短语类型，歧义很少，而且歧义单词的意义也可以基于局部上下文采用词类和语义特征加以排歧。常用的语义特征有：MONTH（月），PLACE（地点），DIRECTION（方向），TIME POINT（时间点），TIME DURATION（时间跨度）和 DEGREE-OF-POSSIBILITY（可能的程度）等。其他可以作为子语言的领域包括设备维护手册、航空旅行查询、约会安排以及饭店推荐等。

本章打破了前面章节的描述模式，不再侧重于对新技术的介绍，而是集中介绍早先给出的那些技术如何在实践中应用。本章的另一个特点是常常需要在各种可选方法和技术中进行艰难的权衡和选择。

21.1 节将简单说明语言之间的差异性。接下来的 4 个小节围绕 MT 的四种基本模型展开讨论：21.2 节将介绍克服语法差异的句法转换方法的使用，以及一些目标语单词的选择技术。21.3 节将介

---

① 英译文把法语 d'être 中的字母 ê 转写为 ecirc; ——译者注

绍在翻译中使用意义的一些方式，特别是题元角色和基元分解的使用。21.4节将给出采用最低限原则的“直接”方法。21.5节将讨论如何利用统计技术来改善MT的各个方面。最后，21.6节将讨论对MT的期望和MT的实际性能之间存在巨大差距的原因，并将讨论在有限的开发资源下满足用户需求的一些策略。

21.1 语言的相似性和差异性

当你碰巧选中了收音机里的某个外语节目时，会觉得这个节目与你在日常生活中熟悉的语言完全不同，就像噪音。但是在这个噪音中也存在一些模式，而且实际上人类语言的某些方面似乎具有普遍性（universal），是每种语言所共有的。许多普遍性都源于语言作为人类交际系统这一功能性的角色。例如，每种语言似乎都有表示人类活动的单词，例如女人（women）、男人（man）、小孩（children）、吃（eating）、喝（drinking）、讲礼貌（being polite）或不讲礼貌等。其他普遍性则更为微妙，例如第8章所提及的每种语言似乎都有名词和动词。

尽管语言之间存在不同，但这些不同也常常具有一定的体系。对不同语言的相似性和差异性进行系统研究的学科称为类型学（typology）(Croft, 1990; Comrie, 1989)。本节将概略介绍一些关于不同语言相似性和差异性的类型学的事实。这与我们的主题MT是有关的，从一种语言翻译为另一种语言的困难之处在很大程度上取决于这两种语言的词汇、语法和概念结构等的相似性。

从形态学角度，语言常常有两种不同的分类方法。第一种是根据每个单词的语素数，将语言分为：孤立语（isolating language），例如越南语和汉语，每个单词通常只有一个语素；多式综合语（polysynthetic language），例如Siberian Yupik（“Eskimo”）（“爱斯基摩语”），其中单个单词可能包含许多语素，甚至可以对应于英语中的整个句子。第二种是根据语素是否可以被切分，将语言分为：粘着语（agglutinative language），例如Turkish（土耳其语，第3章曾讨论过），语素都有相对比较清晰的边界；溶合语（fusion language），例如Russian（俄语），其中单个词缀可能合并了多个语素，例如单词stolom（table-SG-INSTR-DECL1）中的-om，溶合了工具格、单数格和第一变格等不同形态学范畴。

从句法角度看，语言之间的最突出的差别在于其简单陈述句的动词（verb）、主语（subject）和宾语（object）的基本顺序。例如，德语、法语、英语和汉语都是SVO语言，即动词倾向于放在主语和宾语之间。相反，印地语和日语是SOV语言，即动词倾向于放在基本从句的末尾，而爱尔兰语、古典阿拉伯语和圣经的希伯来语都是VSO语言。基本单词顺序一致的语言常常也会有其他相似性。例如，SVO语言通常具有介词（preposition，又称为前置词），而SOV语言通常具有后置词（postposition）。英语是to Yuriko，日语是Yuriko ni。

另一种重要的句法–形态学区别在于是中心语标记语言还是附属语标记语言(Nichols, 1986)。中心语标记语言倾向于将中心语和它的附属语之间的关系标记于中心语，而附属语标记语言则倾向于标记于非中心语。例如，Nichols(1986)注意到在匈牙利语中所属关系被标记为对中心语名词（head noun，简称为H）的一个词缀（affix，简称为A），而在英语中它被标记于非中心语。

(21.1)

English	the	man-^{A}'s	^{H}house
Hungarian	az	ember	^{H}ház-^{A}a
	the	man	house-his

句法区别与语义区别的关系在于语言如何将概念映射于单词。Talmy(1985, 1991)注意到，语言可以通过运动方向和方式是标记于动词还是标记于“外围词”（satellite）加以区分。外围词包括小

品词、介词短语或副词短语等。例如用英语描述的 *a bottle floating out of a cave* 中表示方向的是小品词 *out*：

The bottle floated out.　　(21.2)

而在西班牙语中方向是被标记于动词的：

La botello salió flotando.　　(21.3)
The bottle exited floating.

运动方向被标记于动词(外围词标记运动的方式)的语言被Talmy称为动词框架；Slobin(1996)给出了例子，如西班牙语中的acercarse(approach)，alcanzar(reach)，entrar(enter)和salir(exit)。运动方向被标记于外围词（动词标记运动的方式）的语言被 Talmy 称为外围词框架；Slobin(1996)给出了例子，如英语中的 crawl out，float off，jump down，walk over to 和 run after。Talmy(1991)注意到动词框架语言包括罗曼语（Romance）、闪美特语（Semitic）、日语（Japanese）、泰米尔语（Tamil）、玻利尼西亚语（Polynesian）、大多数的班图语（Bantu）、大多数的玛雅语（Mayan）、内慈佩尔塞语（Nez Perce）以及喀多语（Caddo），而外围词框架语言包括除了罗曼语以外的绝大多数印欧语系的语言、乌戈尔语（Finno-Ugric）、汉语（Chinese）、奥杰布瓦语（Ojibwa）和瓦尔皮黎语（Warlpiri）。

除了系统分布于许多种语言中的这类属性以外，也存在许多某些语言独有的具体特征。例如，英语中有一个涉及 *there* 的独特句法结构，常常被用于介绍故事中新出现的场景，例如 *there burst into the room three man with guns*（屋里突然闯入三个带枪的男人）。

如果你想知道语言之间的这些差异到底有多么寻常而关键，那就看看日期。日期在各种语言中不但是以各种形式出现（举几个典型的例子，在日语中是 YYMMDD，在美国英语中是 MM-DD-YY，在英国英语中是DD/MM/YY），而且可能采用不同的历法。例如，日语中的日期常常采用相对于当前君主统治开始的年份而不是基督元年。

现在来看看词汇的组织问题，这里有许多有趣的模式。许多单词可以通过相对直接的方式翻译为其他语言。例如，英语单词 *dog* 翻译为汉语中的“狗”。英语单词 *chocolate* 翻译为意大利语中的 *cioccolato*，日语中的 *chokoreeto*[①]。

有时这种翻译的结果并不是目标语的单个单词，而是一个固定短语。例如，法语 informatique 翻译为英语 *computer science*。而在一些更复杂的情形中，一种语言中的一个单词并不能简单地映射为另一种语言中的一个单词或短语。

例如，从语法角度，一个单词最好的翻译可能是目标语中不同词性的单词。许多涉及动词 like 的英语句子在翻译为德语时必须使用副词 *gern*，因此 *she likes to sing* 对应于 *sie singt gerne*，这里句法结构也受到了影响。

有时，与其他语言相比，某些语言对单词选择会施加更多的语法约束。例如，英语需要区分代词的性别，而汉语则不需要；因此，将汉语的第三人称单数翻译为英语时需要确定原始的所指对象到底是阳性还是阴性。同样地，当将英语中的复数代词 *they*（未指定性别）翻译为法语（阳性 *ils*，阴性 *elles*）时，也会碰到这种情形。在日语中，由于没有单个单词对应于 *is*，因此说话人必须基于主语是否有生命[②]而在 *iru* 或 *aru* 之间进行选择。

① 其实日语中的 *chokoreeto* 要比英语中的 *chocolate* 更正式，因为日语中还有一个非正式的较短的表达形式 *choko*。

② 正在服务的出租汽车和公共汽车有时被认为是有生命的。

这种个体性的差异也表现在语义方面：在一个特定的概念领域，一种语言可能比另一种语言具有更细致的划分。例如，表示亲属关系的英语词汇特别贫瘠，单个单词brother既可以表示哥哥，又可以表示弟弟。相反，日语和汉语在氏族关系中就会区分长幼。图21.1给出了一些此类的例子。

英语	*brother*	日语	*otooto*（幼）
		日语	*oniisan*（长）
		汉语	*gege*（长）
		汉语	*didi*（长）
英语	*wall*	德语	*Wand*（里）
		德语	*Mauer*（外）
英语	*know*	法语	*connaître*（熟悉）
		法语	*savoir*（了解某个命题）
英语	*they*	法语	*ils*（阳性）
		法语	*elles*（阴性）
德语	*berg*	英语	*hill*
		英语	*mountain*
汉语	*tā*	英语	*he, she* 或 *it*

图21.1 语言个体性方面的差异

不同语言在以词汇划分概念空间时，不同的方式可能比这种一对多的翻译问题更复杂，它将导致多对多的映射。例如，图21.2是由Hutchins and Somers(1992)给出的对这种复杂性的讨论，以英语单词*leg*，*foot*和*paw*翻译为法语的*jambe*，*pied*和*patte*为例。

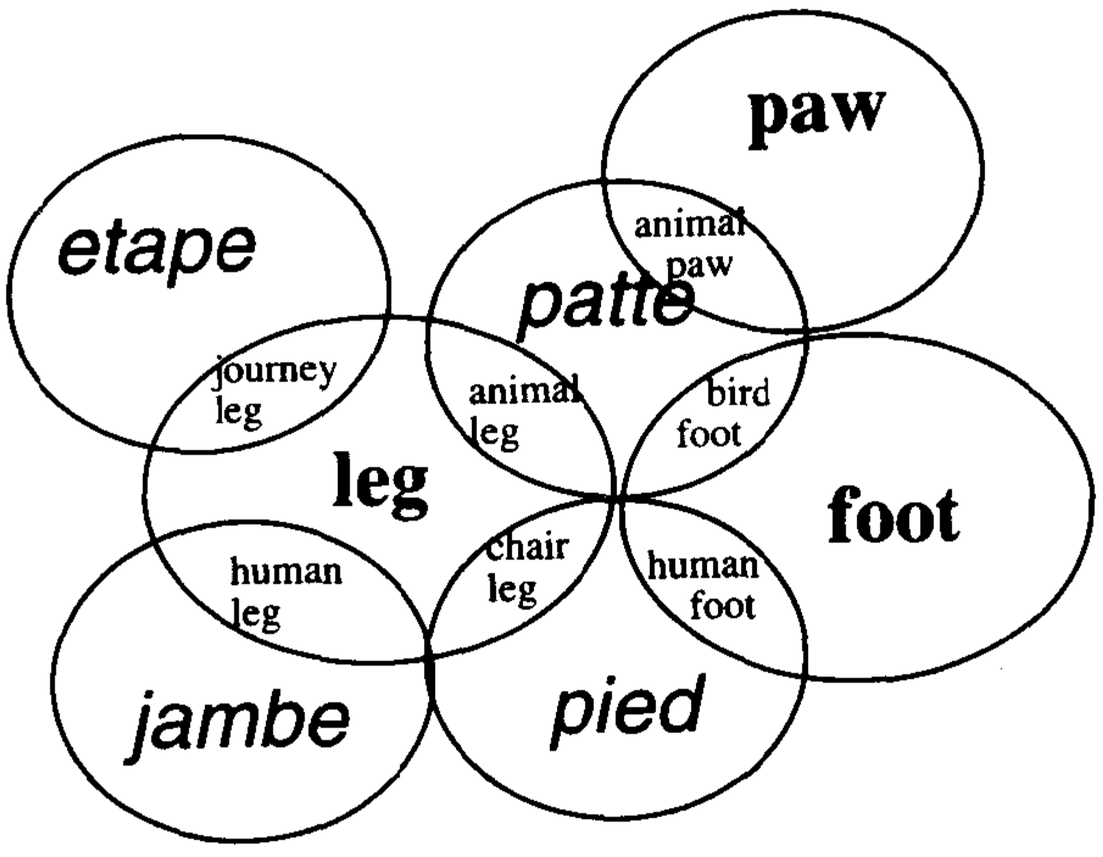


图21.2 在英语单词*leg*和*foot*与各种法语译词之间的复杂重叠

此外，一种语言也许会存在词汇空白（lexical gap），除了解释性的注解以外没有任何单词或短语能够用于表达另一种语言中某个单词的意义。例如，日语中没有任何单词可以表示英语的*privacy*（独处），而英语中也没有单词能够表示日语的*oyakoko*（孝顺）。

再者，原文和译文读者的背景和期望所表明的那类文化背景会使问题进一步复杂化。许多翻译理论家(Steiner, 1975; Barnstone, 1993; Hofstadter, 1997)都提及Jorge Luis Borges的一个精巧的故事，以表明这样一种观点：即使两种语言的文本具有相同的单词和语法，也可能会因为它们不同的文化背景而具有不同的意义。Borges编造了关于Menard的故事，Menard是一个20世纪30年代的作家，他的目标是逐字地重新创作Cervantes（塞万提斯）的Don Quixote（唐吉诃德）：

Cervantes的文本和Menard的文本的用字是相同的，但是后者几乎是无限丰富的。它的贬低者也许会说是具有更多的歧义，但是歧义事实上也是一种丰富。比较Menard的Don Quijote与Cervantes的文本就会发现这一点。例如，Cervantes写到：

> *... la verdad, cuya madre es la historia, émula del tiempo, depósito de las acciones, testigo de lo pasado, ejemplo y aviso de lo presente, advertencia de lo por venir.*

而Menard写到：

> *.... la verdad, cuya madre es la historia, émula del tiempo, depósito de las acciones, testigo de lo pasado, ejemplo y aviso de lo presente, advertencia de lo por venir.*

文字上完全相同，但是在风格上相反。Menard的古老的风格——在最后的分析中，一个异乡人——体验到某种情感。而他的先驱就不是这样，他能够轻松地驾驭与他同时代的普通的西班牙语。

最后这些观点提出了一个关于文化差异和翻译可能性（或不可能性）的更广泛的问题。一种理论观点，有时被称为萨丕尔-沃尔夫假说（Sapir-Whorf hypothesis），认为语言可能会约束思想，即你说的语言可能会影响你的思想方式。如果这个假设为真，那么就不存在完美的翻译，因为源语和目标语的说话人必然分属于不同的概念系统。无论怎么说，下面这一点是清楚的：语言之间的差异越深，翻译处理就会越困难。

21.2　转换模型

正如上一节所阐明的，语言之间具有差异。MT的一种策略就是通过能够克服这些差异的处理来进行翻译，即改变输入语言的结构而使之顺从于目标语言的规则。这可以通过应用对比知识（contrastive knowledge）（即两种语言之间的不同）得以实现。采用这种策略的系统有时被称为是基于转换模型（transfer model）的。

由于转换时需要有输入的结构表示，因此转换必须以某种形式的剖析为先决条件。而且，由于转换的结果只是目标语的一个结构，因此必须通过随后的生成阶段来生成输出句。可见，这种模型的MT涉及三个阶段：分析（analysis）、转换（transfer）和生成（generation）。其中，转换在源语剖析器的输出和目标语的生成器的输入之间架起了一座桥梁。图21.3给出了这种转换体系结构的草图。

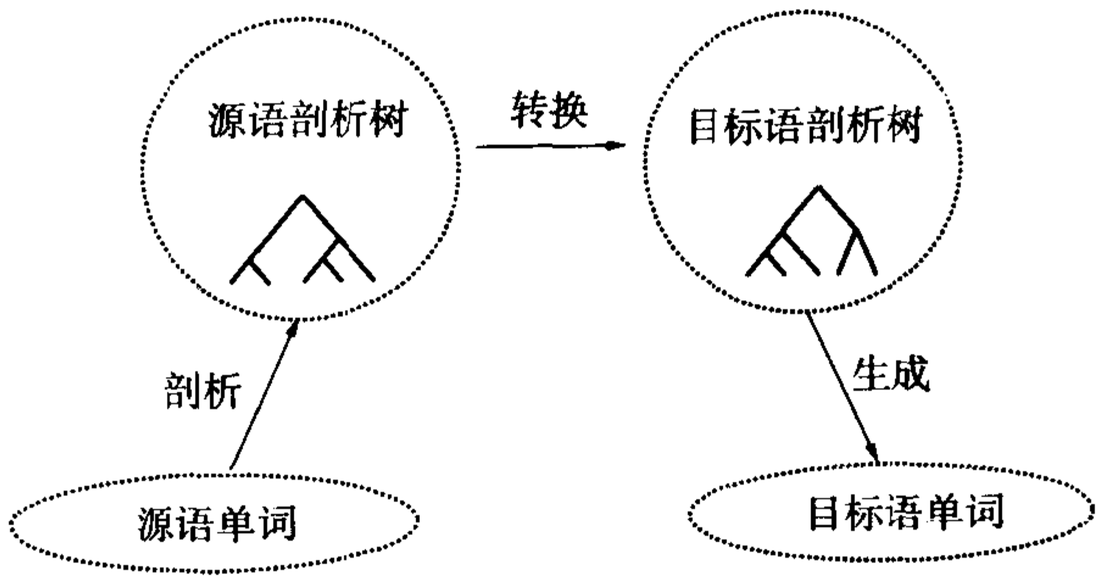


图21.3　转换机器翻译系统的体系结构

注意，用于MT的剖析器与其他目的所需的剖析器有可能不同。例如，假定我们需要将*John saw the girl with the binoculars*翻译为法语。剖析器并不需要确定这个介词短语附着的歧义问题，因

为这两种可能性导致的法语句子是相同的。当然，并不是所有介词短语附着问题都是如此，因此MT系统也需要能够表示判定歧义后的剖析，尽管仍要能够处理带歧义的剖析。

21.2.1 句法转换

让我们从研究句法之间的不同开始。上一节提到，英语中一个名词短语的默认顺序是形容词在名词之前，而法语中是形容词在名词之后[①]。让我们暂时推迟如何翻译单词的问题，研究一个MT系统如何克服这种差异。

图21.4给出了基本思想。这里将一个适合描述英语短语的剖析树转换为一个适合描述法语句子的剖析树。一般来说，句法转换是从一种树结构映射到另一种树结构的操作。

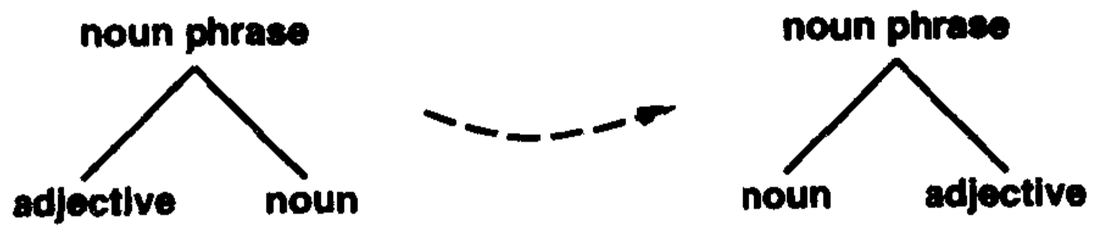
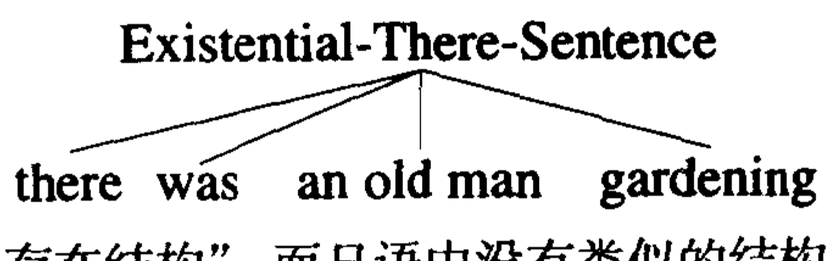


图21.4 一个对形容词（adjective）和名词（noun）进行重新排序的简单转换

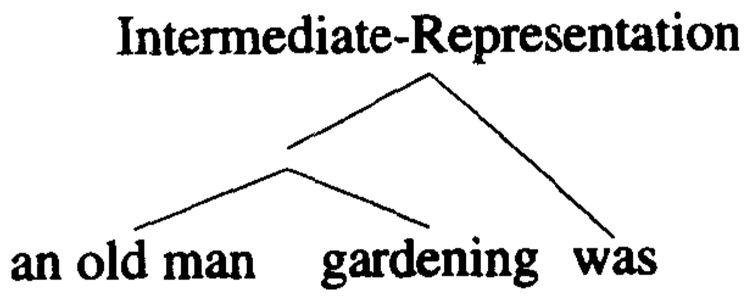
下面利用一个经简化的句子来粗略地说明如何通过这种转换来重组整个句子：

There was an old man gardening. (21.4)

我们将假定剖析器可以给出下面的结构，也将假定系统从该树的根结点开始自顶向下实施转换：

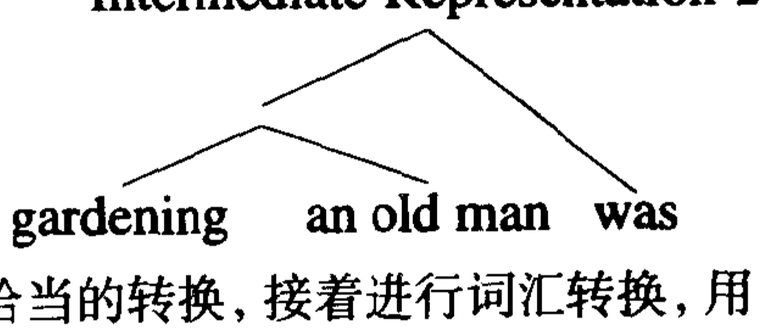


因为该句涉及一个“*there*存在结构”，而日语中没有类似的结构，因此我们不得不直接应用一个转换，删除句子开头的*there*，并且将第四个成分转换为一个修饰名词的关系从句，产生类似下面的结构：

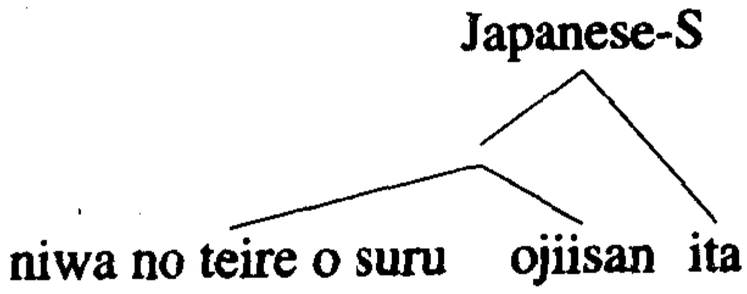


这种结构更类似于伪英语句子 *an old man, who was gardening, was* 的结构。

接着，应用另一个转换来颠倒名词短语和关系从句的顺序，即可得到下面的结构：



目前为止已经应用了所有恰当的转换，接着进行词汇转换，用日语单词代替英语单词（下一节将讨论有关内容）。这样即可得到下面的结构：



① 这个普遍总则存在一些特例，例如英语中的 galore 和法语中的 gros；而且在法语中有些形容词可以出现于名词之前，只是意义不同；例如 route mauvaise（坏路）和 mauvaise route（错路）(Waugh, 1976)。

之后，还需要少许句法工作以产生实际的日语句子。这些句法工作包括：(1) 添加单词 *ga*，日语中用它来标记主语；(2) 根据有生性（animacy）来选择与主语一致的动词，也就是 *iru*，而不是 *aru*；(3) 动词变形。在最后的生成步骤，或者否定或者线性化该剖析树，以生成单词串。尽管这些生成任务可以通过第20章介绍的技术来实现，但实际系统通常直接采用简单的处理过程。无论如何，最后的输出应该是：

niwa	*no*	*teire*	*o*	*shite*	*ita*	*ojiisan*	*ga*	*ita.*
garden	GEN	upkeep	OBJ	do	PAST-PROG	old man	SUBJ	was

图21.5给出了上面讨论的转换的概略表示。这种转换可以通过改写模式规则加以实现：如果输入与一个转换的左边相匹配，则根据该转换的右边对它进行改写。

	English to French:
1.	$NP \rightarrow Adjective_1\ Noun_2$ $\Rightarrow$ $NP \rightarrow Noun_2\ Adjective_1$
	Japanese to English:
2.	$Existential\text{-}There\text{-}Sentence \rightarrow There_1\ Verb_2\ NP_3\ Postnominal_4$ $\Rightarrow$ $Sentence \rightarrow (NP \rightarrow NP_3\ Relative\text{-}Clause_4)\ Verb_2$
3.	$NP \rightarrow NP_1\ Relative\ Clause_2$ $\Rightarrow$ $NP \rightarrow Relative\text{-}Clause_2\ NP_1$

图21.5 一些转换的非正式描述

对MT系统中的转换，也可以加入更复杂的应用条件，可以包括一个“触发器”，即为了效率而用于索引该模式的一个特别的单词。形式化转换的一种方式是采用基于合一的模型，的确如第11章所述，MT中需要一个可回溯的运算，这是特征结构合一(Kay, 1984)和项目合一(Colmerauer and Roussel, 1996)的原始动机。然而，合一运算因为计算开销太大而未能普遍采用。

21.2.2 词汇转换

有些输出单词是在句法转换或生成过程中确定的。在上面给出的例子中，功能词 *ga* 和 *ita* 主要是由语法控制的，而实词的情形有所不同。找到与输入实词等价的目标语单词的处理，称为词汇转换（lexical transfer），这个过程并不容易，原因可参阅21.1节。

词汇转换的基础是对一个跨语言词典进行的查询处理。从早先的讨论可以知道，翻译等价词可能是单个单词，也可能是短语，在本例中 *gardening* 变为 *niwa no teire o suru*（do garden upkeep）。而且，如本例所示，有时在这个阶段必须通过随后的生成处理对单词进行变形。

在21.1节也讨论了一个单词具有多个翻译的问题。在本例中 *man* 就是这样的单词。这里的正确选择是 *ojiisan*（老人），但如果输入句子是 *man is the only linguistic animal*，则 *man* 的翻译可能是 *ningen*（human being, man, men）；在大部分其他例子中，*man* 也可能是hito（person, persons, man, men）或其他可能正确的单词。幸运的是，这里至少可以采用两种方式来解决该问题：在剖析阶段或在生成阶段。第一种方法是把这个单词当成歧义单词来处理。也就是说，我们假定 *man* 可以对应于两个以上的概念（可能是HUMAN和ADULT MALE），则选择正确的日语单词的处理就好比对这些概念进行排歧。这种对待词汇转换的方式使得我们可以应用词汇排歧的所有标准技术

(见第16章)。第二种方法认为这类单词只有一个意义，而通过目标语生成所施加的约束从多个可能的翻译(*ningen*，*hito*，*ojiisan*等)中进行选择(Whitelock, 1992)。在实践中，对于这类例子常常在剖析阶段进行处理，因为生成阶段词汇选择算法的开销很大(Ward, 1994)，特别是对于实词(见21.5节)。

在本例中，如何翻译*man*的选择是比较容易的。因为前一个单词是*old*，所以正确的翻译为*ojiisan*(老人)。多个源语单词必须由单个目标语单词表示的这类输入，处理起来可能会比较困难，需要一般类型的推理。但是在许多例子(也包括本例)中，我们可以简单地将它们当成成语，在双语词典中添加有关它们本身的条目即可。

21.3 中间语的思想：使用意义

转换模型带来的一个问题是，每一对语言都需要一个截然不同的翻译规则集。对于应用于类似欧共体那样的多语言环境的翻译系统，这显然不是一种最理想的选择。

这提出了一种关于翻译本质的不同观点。转换模型把翻译看成是一个将输入句子的结构和单词转换为目标语的正确句子的处理过程。而另一种观点把翻译看成一个获取输入的意义，然后将该意义表达为目标语的处理过程。按照这种观点，一个MT系统可以不通过对比知识，而仅仅依赖于与语言的标准解释器和生成器一致的句法和语义规则来进行翻译。因此，所需的知识与MT系统能够处理的语言的数目只是成线性比例增长，而不是成平方次幂增长。

这种方案必须以存在意义表示或中间语(interlingua)为先决条件，这类中间语类似于第14章介绍过的意义表示，是一种与语言无关的范式。其核心思想是：采用中间语表示所有句子意味着无论该事件所依托的是哪种语言，该事件总还是以相同的方式进行着的“相同”的事件。采用该模型的翻译处理过程为：首先通过对输入实施语义分析，将语言X处理为中间语表示，接着由中间语生成语言Y。

在中间语表示中，常常采用的元素是题元角色(见第16章)的一个小的固定集合。当用于中间语时，这些题元角色被认为是所有语言通用的。图21.6给出了例句*there was an old man gardening*的可能的中间语表示，该表示是一个合一风格的特征结构[①]。通过第15章，我们已经知道一个语义分析器可以产生如*man*和*gardening*之间带有的AGENT关系这样的结构。注意，中间语除了句法结构还需要这样的语义解释，因此它比转换模型需要更多的分析工作，转换模型仅需要句法剖析。但是，生成可以从中间语直接处理而不需要句法转换。

$$\begin{bmatrix} \text{EVENT} & \text{GARDENING} \\ \text{AGENT} & \begin{bmatrix} \text{MAN} & \\ \text{NUMBER} & \text{SG} \\ \text{DEFINITENESS} & \text{INDEF} \end{bmatrix} \\ \text{ASPECT} & \text{PROGRESSIVE} \\ \text{TENSE} & \text{PAST} \end{bmatrix}$$

图21.6 *there was an old man gardening*的中间语表示

注意，在图21.6中是以GARDENING作为EVENT特征的值的。尽管通过第14章我们已经熟悉了这种形式，但是也许有人会觉得这不是一个真正的中间语表示，而更像一个英语单词。这里存

① 当然这对于解释there存在结构的意义是远远不够的。事实上，当前对英语there结构句法和语义的最少的不完全的解释也要124页的篇幅(Lakoff, 1987)。

在一个关于中间语的概念和关系的正确分类的非常深刻的问题，即采用哪种本体知识体系（ontology）。当然，一个意义表示的设计者可以自由地选择标记集并确定它们的意义。但是，为MT选择本体知识体系不能有丝毫的轻率，因为它会对整个系统的体系结构产生约束。例如，回想第16章讨论过的题元角色的两种可能的分类，一种包括 AGENT 和 FORCE，而另一种只包括 AGENT。采用哪种分类会影响到系统对句子的翻译方式，例如将句子 *the quake broke glass*（见第16章）翻译为日语，这里 *quake* 需要以 *de* 为标记，而不是通常的主语标记 *ga*，因为地震不是有生命的。如果采用较小的只包括 AGENT 的分类来设计中间语，则这个句子的表示将为 *quake* 施加 AGENT 角色，而 *de* 与 *ga* 的问题将留给生成器。然而，如果采用图 16.8 所示的扩展分类，则该表示将包括 FORCE 角色以及由语义分析器实施的做出该决策所必需的工作。

中间语思想的本质不但包括句法转换，还包括词汇转换。该思想是避免源语单词和目标语单词之间关系的明显描述，主张通过概念之间的映射（也就是说，本体知识体系中与语言无关的元素）来进行词汇转换。回忆早先讨论过的是否应该将 *man* 翻译为 *otoko*，*ningen* 或 *ojiisan* 等的问题，显然所涉及的大部分处理与翻译为日语的目标并没有关系。对于 *man*，可以采用更一般化的概念，例如GENERIC-HUMAN 和 MALE-HUMAN。如果在中间语中采用这种概念，则大部分翻译处理都可以通过一般的语言处理技术和模块来实现，而从英语到日语的翻译任务所特有的处理将被消除或至少得以削减。

一些中间语以及一些其他的表示则更进一步，采用了词汇分解，也就是将单词分解为组成它们的成分的意义。在图 21.6 中可以看到这种形式，*was* 映射为 PAST 和 PROGRESSIVE，*a* 映射为 SINGULAR 和 INDEFINITE。实词的分解也是可能的：单词 *drink* 可以通过(INGEST，FLUID, BY-MOUTH)[①]来表示。通过分解单词这种方式表示句子，似乎确实可以捕捉到意义的某些方面，而不仅仅是对与输入类似的英语单词标记的重排。而且，这种表示也可能用于基于推理的排歧。例如，有可能采用单词的意义来推断在句子 *the policeman saw the man with a telescope* 和 *the policeman shot the man with a telescope* 中介词短语修饰的对象。然而，除了在很小的领域，一般很难将这类推理应用于多个例子。通常在实践中并不采用这种高性能的基于中间语的技术。

掠过许多重要的细节，现在让我们来对比一下转换模型与中间语模型的不同。最关键的一点是中间语模型通过使剖析器/解释器和/或生成器多增加一点工作，消除了在对比知识方面的需求，如图 21.7 所示。

然而，采用中间语所涉及的这点额外工作并不总是能够非常容易地完成。这需要系统设计者对该领域的语义进行详尽的分析，并将其形式化于一个本体知识体系(Levin et al., 1998)。尽管在子语言领域这还算比较容易处理，但是今天来看这更像一门艺术而不是科学。在某些情形下，例如在航空旅行、酒店预订、餐厅推荐等领域，这种语义可以主要通过一个数据库模型来加以捕捉。在这些领域数据库中的定义确定了可能的实体和关系，而MT系统设计者的任务主要在于如何使这些实体和关系映射到所涉及的两种语言的单词和结构。

中间语思想的另一个问题是，它的纯形式需要系统在所有时间都能彻底排除歧义。对于一个真正通用的中间语，这可能引起某些不必要的工作。例如，为了从日语翻译为汉语，该中间语必须包括一些概念，例如 ELDER-BROTHER 和 YOUNGER-BROTHER。然而，采用这些相同的概念在从德语到英语的翻译过程中将需要一个剖析器实施一些不必要的排歧工作，并且进一步需要系统具有

---

① 采用这个语义分解可使我们清楚地知道 *drink* 意义的哪个元素与 *eat* 有关，哪个元素与 *eat* 无关。但是，正如第16章中的讨论，词汇语义通常并不是这么容易。例如，如何用形式语义来表示 *heft*（分量）的意义以及它与 *weight*（重量）的不同，或 *sporadic*（零星的）和 *intermittent*（间歇的）的不同。

保留歧义（preserving ambiguity）的技术，以确保输出以与输入完全一致的方式保留了歧义或模糊。即使对 Sapir-Whorf 思想持怀疑态度，采用通用意义表示所有语言的思想显然也不是没有问题的。

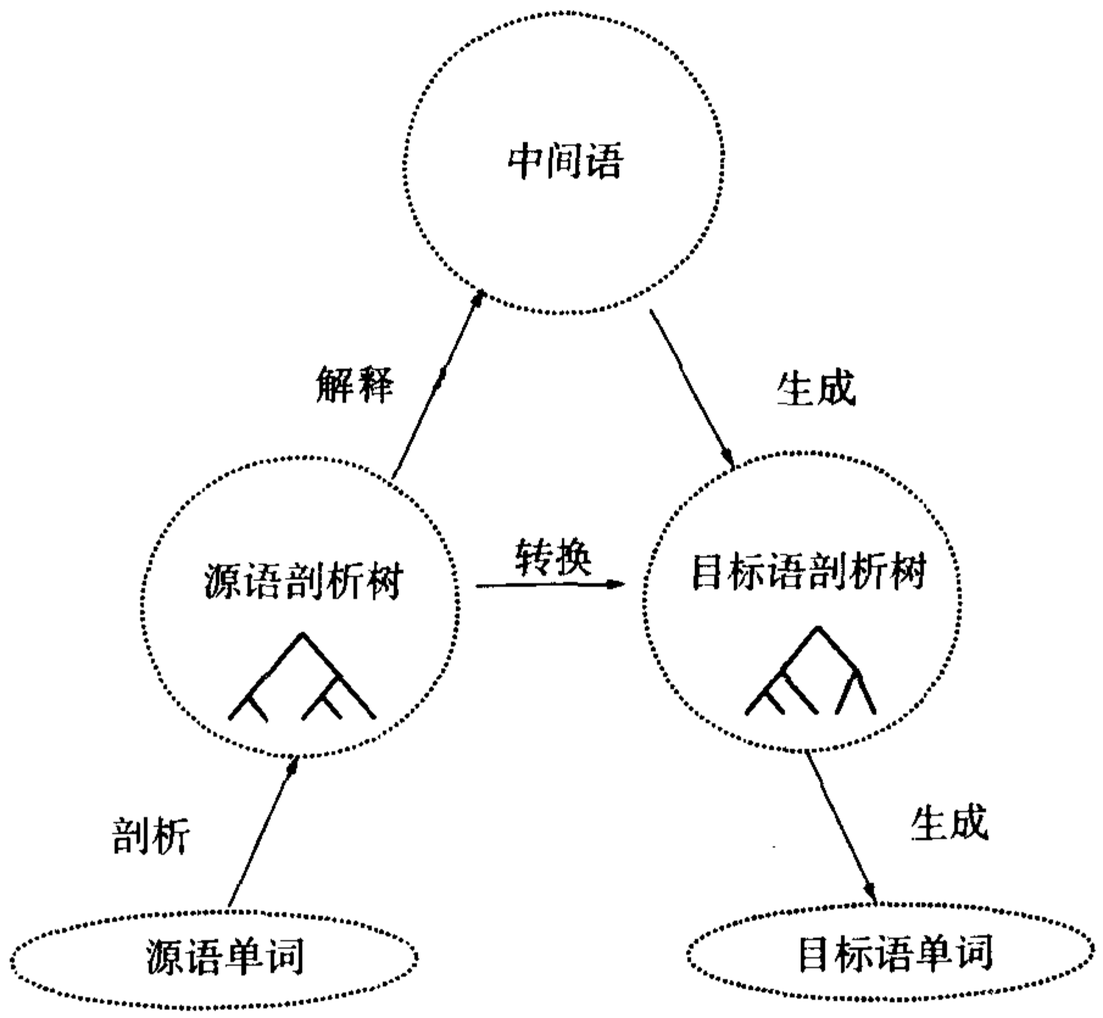


图 21.7 转换和中间语模型之间关系的图形表示，这个表示通常归功于 Vauqois

21.4 直接转换

所有这些模型都很不错，但是如果分析失败会怎么样呢？用户并不喜欢因为“没有找到剖析树”而导致的“零”输出。通常，与“零”输出相比，用户更愿意接受那些不够完美的结果。这对于中间语模型尤其是一个挑战，系统不应该仅仅因为不能确定it的题元角色是FORCE还是AGENT，就不能对句子 *it broke the glass* 做出翻译。

几种方法都可以用于解决该问题。一种方法是采用第 15 章讨论过的鲁棒的剖析技术，它有时相当于通过片断来进行翻译。另一种方法是从根本上放弃生成精致的结构分析的思想，而仅仅进行一些简单的能够可靠实施的操作。更激进地，我们可以采纳这样一个原则：MT 系统应该尽可能少做工作。根据这个原则开发的系统有时被称为直接翻译（direct MT）系统。通常这种系统只考虑一对语言，所有实施的处理只是那些从一种指定语言变为另一种指定语言所必需的处理。

典型的直接MT系统一般由几个阶段组成，每个阶段集中于一种类型的问题。例如，我们可以将一个日语句子通过6个阶段改写为一个英语句子，如图21.8所示。图21.9以一个简单的例子说明了这几个阶段的处理过程。

阶段	操作
1	词形分析
2	实词的词汇转换
3	与介词有关的各类工作
4	SVO重新排列
5	其他杂类的处理
6	词形生成

图 21.8 采用直接机器翻译系统把日语翻译为英语的 6 个阶段

Input:	watashihatsukuenouenopenwojonniageta.
After stage 1:	watashi ha tsukue no ue no pen wo jon ni ageru PAST.
After stage 2:	I ha desk no ue no pen wo John ni give PAST.
After stage 3:	I ha pen on desk wo John to give PAST.
After stage 4:	I give PAST pen on desk John to.
After stage 5:	I give PAST the pen on the desk to John.
After stage 6:	I gave the pen on the desk to John.

图21.9 说明直接系统处理过程的例子

图21.9中的阶段1把输入切分为单词（我们知道日语与汉语一样在词与词之间没有空格），并对复杂的动词形式进行形态分析。这些可以通过第3章的有限状态技术和第5章描述的切分算法（例如概率算法）来实现。

阶段2为所有实词选择等价的英语翻译。这可以通过采用双语词典，或基于局部上下文和已经选出的目标语单词的处理方法来选择正确的翻译。图21.10给出了该方法的处理程序。

```
function DIRECTLY_TRANSLATE_MUCH/MANY(Russian word) returns

if preceding word is how
   return skol'ko
else if preceding word is as
   return stol'ko zhe
else if word is much
   if preceding word is very
      return nil (not translated)
   else if following word is a noun
      return mnogo
else /* word is many */
   if preceding word is a preposition and following word is a noun
   return mnogii
else return mnogo
```

图21.10 一段将 *much* 和 *many* 翻译为俄语的处理程序，改编自 Hutchin(1986,. p133)对 Panov1960 的讨论

在这个例子中词汇转换很平常，比较简单。通常，在目标语单词之间可能存在相互依赖，因此词汇转换应该在下一个阶段进行，例如，动词在名词之前，名词在形容词之前。例如，研究将日语 *nomu* 翻译为英语的问题，这里的翻译只可能为 *drink* 或 *take*(medicine)。这个决策必须在修饰成分的翻译选择之前就确定，这样才会给出像 *drinking heavily* 和 *taking a lot of medicine* 这样的翻译，而不是这两者杂乱的拼凑。通常，尽管已有一些标准的解决方案（见第20章），但是做出决策的正确顺序的问题非常错综复杂。

在阶段3将 *no ue no* 翻译为 *on*，并倒置相关的两个名词短语（*desk* 和 *pen*），因为在英语中介词短语是位于它们所修饰的单词之后，而不是之前。依照 *gave* 的词典条目（存在关于 *give* 用法的详细说明），选择 *to* 作为 *ni* 的翻译。

在阶段4调用一个程序，以实现将动词从句子末尾到主语之后的位置转移，并且删除主语和直接宾语的格标记。

在阶段5进行一些混杂的处理，例如将格标记移到名词之前，插入冠词等。

最后在阶段6进行动词的屈折变化。

该方法与前述的方法有几个明显的不同。首先，这是MT任务的一种新的模块化方式，与图21.7所示的转换和中间语的模块化方式互不相关。在直接模型中，涉及某个指定问题分析的所有处理都是在一个阶段完成的，包括分析、转换和生成等几个方面。这种处理的优点是：一次解决一个指定问题可能更易于操作。另外，如果在所有各类的单个分析问题之间或所有单个生成问题之间存在协作，则可以将这些处理组织为较大的模块（分析、转换、合成），而这是非常有益的。

直接系统的第二个特点是词汇转换可能更具过程性。词汇转换过程可以兼收并蓄地查询相邻单词、依附语以及中心语的句法类和语义属性，例如图21.10所示的将*much*和*many*翻译为俄语的类似决策树的处理过程。

直接模型的第三个特点是一般比较保守，仅当单词对单词的直接替换结果中出现明显不符合语法的现象时才会改变单词的顺序。特别是，直接系统通常将词汇转换置于句法处理之前。

也许直接模型的最关键的特点是抛开了完全的结构和表示。通常，它们将输入当成一个单词串（或音素串），并直接对它进行各种操作，例如用目标语单词替换源语单词，给单词重新排序等，以最终生成目标语中的符号串。

当然，在实践中运行的MT系统倾向于将直接模型、转换模型和中间语模型结合起来。毫无疑问，句法处理并不意味着全部，也不是一无是处。即使系统没有完全的剖析结果，也可以通过像词性标注、从句或短语切分、依存关系以及括弧标记等各种有用的句法信息来为输入增添可用信息。许多以直接翻译为特征的系统也常常采用与转换和中间语方法有关的各种技术(Hutchins and Somers, 1992)。

21.5 使用统计技术

在前几节介绍的三种MT的体系结构（转换、直接语和直接模型）中，都需要回答采用哪种表示以及通过怎样的步骤来实现翻译的问题。但是，在翻译问题的处理中也存在另外的方式：将注意力放在结果上，而不是放在过程上。让我们从这个视角来研究一个句子是一些其他句子的翻译到底意味着什么。

翻译学家已经给出了许多有关这个问题的思考。令人沮丧的是，他们的一致意见似乎是：严格地说不可能将一种语言的句子翻译为另一种语言的句子。例如，一个人不可能真正将希伯来语(Hebrew)*adonai roi*（the Lord is my shepherd，上帝是我的牧羊人）翻译为一种在其文化中没有“羊”（sheep）的语言。一方面，我们可以用目标语给出对一种事物的清晰描述，但代价是损失一部分对原文的忠实性，例如*the Lord will look after me*（上帝会关照我）。而另一方面，我们可以忠实于原文，但代价是给目标语读者带来某些理解上的障碍，例如*the Lord is for me like somebody who looks after animals with cotton-like hair*。再拿另一个例子来看，如果将日语短语*fukaku hansei shite orimasu*翻译为*we apologize*，则没能忠实于原文，而如果将它翻译为*we are deeply reflecting (on our past behavior, and what we did wrong, and how to avoid the problem next time)*，则输出就有些笨拙或不够清晰。这些问题不仅出现于碰到特定文化的概念时，而且还出现于一种语言使用了另一种语言没有精确对应的比喻、结构、单词或时态时。

因此，真正能够既忠实于源语又像目标语中的话段那样自然流畅的翻译，有时是不可能的。如果打算无论如何都要硬着头皮勇往直前而生成一个翻译，那就不得不在忠实与流畅这两者之间进行折中。这也是翻译家在实践中的真实写照：他们力图生成这两种标准都可以容忍的翻译。

这提示着我们如何实现MT。我们可以将翻译目标模型化为寻找某个输出，它能够最大限度地表示忠实性（faithfulness）和流畅性（fluency）这两个重要因素的一些函数。如果选择忠实性和流畅性的乘积作为翻译性能的衡量标准，则可以将翻译问题形式化为下面的公式：

$$\text{best-translation } \hat{T} = \text{argmax}_T \text{ fluency(T) faithfulness(T,S)}$$

其中，T 是目标语句子，S 是源语句子。

这个翻译模型是由语音识别的研究者(Brown et al., 1990, 1993)首次提出的，显然该模型类似于第 7 章介绍过的用于语音识别和 5.4 节用于拼写校对的贝叶斯模型。如果反方向来思考翻译问题，则可以更清楚地看出它们之间的类似性，并且可以应用 5.4 节的噪声信道模型：假定需要翻译的输入是一些目标语句子的错误版本。因此，我们的任务就是找到满足下面的条件的目标语句子：

$$\text{best-translation } \hat{T} = \text{argmax}_T \text{ P(T) } P(S|T)$$

为了实现该目的，需要完成三件事：量化流畅性 $P(T)$、量化忠实性 $P(S \mid T)$以及设计一个能找出这两项乘积最大的句子的算法。

这是一个创新。在转换、中间语和直接模型中，处理的每一步对输入句子进行某些调整，以使它更接近于一个流畅的目标语句子，同时又遵守不对输入句子的意义进行大的改动的约束。在这些模型中处理是呆板的，往往不能灵活地为了少量自然性（即流畅性）而牺牲少许忠实性，或者反过来，为了拘泥于手边指定的输入句子（忠实性）而牺牲流畅性。这个有时被称为统计翻译模型（statistical model of translation）的新模型恰好容许这一点。

21.5.1　流畅性的量化

令人欣慰的是，我们已经有一些用于度量一个句子与真正的英语句子之间近似程度的有用的标准：即第 6 章和第 8 章介绍过的语言模型。这使得我们可以区分出哪些句子具有一定的可读性但不是真正的英语句子（例如 *that car was almost crash noto me*），而哪些才是更流畅的英语句子（例如 *that car almost hit me*）。这对于确定单词顺序和搭配特别有用，也是对第20章生成技术的有益补充。

流畅性的模型可以任意地复杂，任何能够给目标语词串指派一个更好的概率的技术都是恰当的，包括第 12 章的更复杂的概率语法或第 17 章的统计语义技术。

当然，采用单语（monolingual）语言知识来改善MT的输出的思想与建立这些知识的统计模型的决策是彼此独立的。许多 MT 系统，特别是直接模型的 MT 系统，都有一个最终阶段，系统在该阶段采用局部考虑来调整输出句子中单词的选择。例如，对每个作为动词的主语（the white house announced today）出现的 *white house* 的首字母进行大写，就是一个合理的启发式规则。

21.5.2　忠实性的量化

给定法语句子 *ca me plaît* (*that me pleases*)和一些可以想到的英语翻译 *that pleases me*，*I like it*，*I'll take that one* 以及 *yes* 和 *good*，从直觉上看第一个句子更忠实于原文。

尽管很难量化这个直觉，但常用于度量忠实性的一个基本因子是：句子中的单词是另一种语言中某个单词的翻译的可能性的大小。因此，可以将一个句子是一个好的翻译的可能性用每个目标语单词是某些源语单词的正确翻译的概率的乘积来近似地表示。为此必须知道每个源语单词与每个可能的目标语单词之间的互译概率。

从哪里才能获得这些概率呢？标准的双语词典并不包括这些信息，但是可以从双语语料库（bilingual corpora）（也就是说，同时拥有两种语言互译文本的语料库）通过计算获得这些信息。这

不是一件简单的事，因为双语语料库并没有标记出某个单词对应的其他语言的单词。为了解决这个问题，首先需要找到双语语料库中句子对齐（sentence alignment）的实现方法。句子对齐是指确定源语中的句子与目标语中的句子之间的对应关系的处理，目前已经取得了比较好的正确率(Kay and Röscheisen, 1993; Gale and Church, 1993; Melamed, 1999; Manning and Schütze, 1999)。接着就是词对齐（word alignment），也就是确定目标语中的哪个或哪些单词对应于源语中的哪个单词或短语的处理。词对齐相当困难（Melamed，pear），常常采用EM算法（见第7章）。通过依照这些方法对齐的双语语料库，可以计算出一个单词、短语或结构对应于它的每个可能的翻译的次数。这种对齐不但可以用于MT，而且也可以帮助翻译家自动生成双语词典的条目(Dagan and Church, 1997; Fung and McKeown, 1997)。

现在让我们来研究一个例子。假定想将由两个单词组成的日语短语 *2000nen taio* 翻译为英语。假定第一个单词最可能的翻译依次为 *2000*，*year 2000*，*Y2K*，*2000 years*，*2000 year* 以及其他的可能。假定第二个单词最可能的翻译依次为 *correspondence*，*corresponding*，*equivalent*，*tackle*，*deal with*，*dealing with*，*countermeasures*，*respond*，*response*，*counterpart* 和 *antithesis* 等。因此，只根据翻译模型，最可能的候选翻译项将是排在最前面的单词的组合，即 *2000 correspondence*[①]。但是，当引入流畅性模型时，也许是一个二元模型，则候选翻译项 *dealing with Y2K* 将获得最高的总分。

当然，更复杂的翻译模型也是可能的，只要能生成多种翻译并能给出对它们进行排序的任何方法。我们甚至可以采用“多引擎”的翻译来实现，将几个翻译模型（例如，一个强大但脆弱的中间语模型和一个鲁棒但性能差的直接模型）并行处理生成的各种翻译和片断进行组合，并以具有最高可信值的组合为最后输出(Brown and Frederking, 1995)。

21.5.3 输出的搜索

迄今为止，我们探讨了有关最佳句子的理论，但是还没有提及寻找它的方法。由于可能的翻译的数目极为庞大，不能通过生成所有可能的翻译的无限集来找出最佳输出。这恰好是一个解码问题，我们已经知道可以通过第7章介绍的修剪 Viterbi 算法（定向束搜索）和 A* 算法来解决这类问题。用于 MT 的解码算法并没有什么特殊：不断生成输出（翻译），在每一点进行评价。如果在任何一点的概率低于某些标准，则剪掉这条前进的路径。生成可以是从左到右或从头（head）向外的。

对于统计 MT，好的介绍性文献包括(Brown et al., 1990)和(Knight, 1997)。文献(Knight et al., 1994)给出了对其中一个最有影响的系统的描述。

21.6 可用性与系统开发

由于 MT 系统通常是由用户操作的，因此用户可以帮助机器。人工介入的一种方式是交互式，也就是说当系统碰到问题时，可以向用户咨询。例如，一个系统给定的输入 *the chicken are ready to eat* 可以生成两种可能的意义解释，系统可以让用户确定该句的意义到底是 *the chicken are ready to be eaten* 还是 *the chicken are ready to eat something*。已经证实，这种交互式 MT 带给用户的是令人难以置信的苦恼，用户不喜欢被迫回答计算机的提问，或者用户感到他们存在的价值就是靠帮助计算机完成它的工作(Cooper, 1995)。而另一方面，人们对修改蹩脚的句子却情有独钟，因此译后编辑（post-editing）成了人们与 MT 系统进行交互时常常采用的方式。

人们也可以编辑源语句子，通过各种方式的简化以改善对输入的翻译。这种译前编辑（pre-editing）的性价比常常比译后编辑更高，特别是如果单个文本需要被翻译为几种语言时，对于需要

① 原文为 countermeasures，明显有误。——译者注

将物品销往多个国家的公司，常常会遇到这种情形(Mitamura and Nyberg, 1995)，译前编辑的成本就会因这几种语言的分摊而得以降低。为了确定译前编辑的内容，一种方法是首先使用MT找出那些错误的内容，然后回头改写这些源语句子。另一种方法是对MT能够处理的情形建立模型，并且要求输入句子都被改写为这种类型的子语言，例如，通过不容许介词短语PP而减少歧义。如果建立了这样的模型，实际上就可以通过训练科技撰稿者只撰写简单的、无歧义的受控语言(controlled language)，省去译前编辑阶段。受控语言的一个英语版本是由通过子语言语法检查器约束的英语现象组成的。采用受控语言也可以使源语文本变得更容易理解。受控语言只是应用于那些有可能改善整个处理过程的较大规模的任务（例如，用户从科技撰稿者那里获取信息），而不是那些原来就有的现成任务（例如，翻译某些现成的文本）。

通常，对MT系统而言用户的满意度极其重要。各种评测标准都可以用于预测系统的可接受性。MT的评测倾向于采用未处理过的（用于信息获取的）标准，包括句子翻译的正确率或近似正确率，正确与否是同时根据忠实性和流畅性来评价的。对将要进行译后编辑的MT输出的典型评测标准是编辑耗费（edit cost），它既可以通过相对于标准翻译的编辑距离的自动算法来计算（类似于第7章对语音识别的评测），也可以通过直接计算将输出修改为可接受文本所花费的时间（或击键次数）而获得。

通常实词是非常重要的，用户一般有能力使混乱的句法复原，而正确地翻译实词才是关键。特别是，使用MT系统的其中一个主要好处就是省去了大部分查询双语词典的令人生厌的工作[①]。因此，专业的MT用户非常重视词典的规模和质量。这些用户常常通过购买特定领域的词典以提升其基本系统词典，这些领域词典是根据该领域的特点而设计的，包括医药、电子、金融、军事报道等。但是，定制词典，即使只是一个从特定领域文本语料库中抽取的词典，对该领域的特定用户来说也是非常接近他们所需的词典的。已经成立的翻译机构通常会投入大量精力来增加他们自己的系统词典。这些词典的结构比较简单，因为通常任何领域的专业词典都是无歧义的，还因为术语几乎总是没有句法特性的开放词类的单词，例如光子（photon）无论出现于什么上下文，都是光子。

很显然，如果MT系统的词典中不但包括单词，而且包含成语、固定短语甚至常见的从句和句子，则MT系统的性能会好得多。这类数据有时可以自动从语料库中抽取。而且，在某些情形下，如果能够在翻译的同时在线抽取而不是将结果存入词典，那就会更有价值，这就是基于实例的机器翻译的核心思想(Sumita and Iida, 1991; Brown, 1996)。

已经证实，用户满意度依赖于翻译质量以外的因素。与翻译质量相比，许多用户更关注其他因素，例如费用、速度、存储要求、直接嵌入他们所喜欢的编辑器的能力、保留SGML标记的能力等。能够存储和召回以前修正后的翻译的翻译记忆库（translation memory）也是一个用户很关注的问题。

依照本书的说明目的，前面只介绍了翻译中出现的一些基本问题，很重要的一点是这些问题远远不够，它们只是MT系统必须解决的所有问题中的一小部分。21.1节曾指出，语言的差异实质上是无尽的复杂性的源泉，如果读过前几节的脚注，你也许会为我们提及的语言的每个“事实”实际上都经过过分简化而感到苦恼。的确，无论选择哪种整体架构，许多MT系统的开发工作都不得不淹没于这类细节的处理过程中。而且，因为一个正在工作的MT系统与任何大型软件系统一样，都是由大量的脆弱代码组成的，添加更多的知识并不总会有帮助。改善了一种现象或一个句子翻译中的缺陷，可能导致其他以前翻译正确的句子变得出错。

---

① MT系统也可以节省大量用于目标语单词输入的时间，特别是对于像汉语和日语这样的需要耗费大量时间进行单词输入的语言。

尽管如此，令人吃惊的是MT系统的性能仍然在不断地完善。一个已经证实的有价值的开发技术是重复开发，创建系统，在实际应用中评测系统和改善系统，并不断地重复进行。在这个过程中，MT 系统会不断地适应某个领域、它的使用者的习惯，并输出用户需要的结果。

21.7 小结

- 尽管MT系统利用了许多标准语言处理技术，但是也有一些技术是MT所特有的，例如引人注目的句法转换。
- 我们给出了 MT 的四个模型：转换（transfer）、中间语（interlingua）、直接（direct）和统计（statistical）模型。但是，目前实用的MT系统常常会对这些模型中的几种思想加以合并，当然 MT 的研究系统也正在设计空间里探索其他更合适的位置。
- MT系统设计是一项艰辛的工作，需要仔细选择模型和算法，然后将它们合并为一个有用的系统。目前来看，MT更像一门手艺而不是科学，特别是因为这必须通过最低限度的开发成本来实现。
- 尽管目前的MT系统设计是特定的，但是也有一些为开发通用的翻译形式化模型而正在进行的努力(Alshawi et al., 1998; Knight and AI-Onaizan, 1998; Wu and Wong, 1998)。
- 尽管存在许多令人印象深刻的改善MT的可能性，但是系统输出令用户满意的应用目前只是以信息获取为目的的粗略翻译（rough translation）、以翻译专家进行译后编辑为目的的草稿翻译（draft translation）以及子语言（sublanguage）领域的翻译。
- 与许多软件任务一样，MT 中的用户界面问题也是至关重要的。对用户而言，MT 系统的价值并不直接与算法或表示的复杂性相关，甚至也不一定与输出质量有关。
- 尽管经过了半个多世纪的研究，MT离成功还有相当大的距离。人类语言是一个无比丰富和令人神往的领域，我们只不过是刚刚开始开采这个宝藏。

21.8 文献和历史说明

对翻译的目标和处理的建模研究至少可以追溯到 4 世纪的 Saint Jerome(Kelley, 1979)。去除人类语言中的缺陷的逻辑语言可用于进行正确的推理、传递真理，因此也可以用于翻译，至少从17世纪人们就开始从事逻辑语言的研究(Hutchins, 1986)。

到了 20 世纪 40 年代，电子计算机出现后不久，MT 的思想就被郑重地提出了(Weaver, 1955)。1954 年第一个 MT 系统原型的公开演示(Dostert, 1955)给新闻界带来了极度的兴奋(Hutchins, 1997)。在接下来的 10 年间，我们看到了 MT 思想的蓬勃发展，这预示着其后的辉煌。但是，该研究工作超前于它所处的时代——MT思想的实现受到诸多事实的限制，例如，由于悬而未决的磁盘研制的事实，没有好的存储词典信息的方式。

高质量MT被证实是难以捉摸的(Bar-Hillel, 1960)，在20世纪60年代中期，越来越多的研究者都一致认识到对形式化和计算语言学的新领域需要更多的基础性研究，同时MT研究的经费受到惊人的削减。MT研究失去了在学术界的地位，机器翻译和计算语言学学会（Association for Machine Translation and Computational Linguistics）去掉了名称中的机器翻译。然而，一些 MT 的开发者仍旧坚持不懈，逐步地、稳定地改善着他们的系统，并且渐渐地赢得了更多的用户。特别是最初由 Peter Toma 开发的 Systran，在 40 多年里一直在不断进行完善。它的早期用户是为了进行信息获取，例如美国空军为了理解俄罗斯的文件。在 1976 年，它的英法版本被欧共体用于生成能够进行编辑

的各种行政文件的粗略翻译。我们在介绍中给出的例子是由 Systran 的网上 Babelfish 免费版本生成的。另外一个早期的成功 MT 系统是 Météo，它可以将英语的天气预报翻译为法语。顺便说一下，它的最初的实现(1976)所采用的 Q-systems 是一种早期的合一模型。

在20世纪70年代末期，学术界再次重现对MT的兴趣。这种兴奋的源泉可能在于采用人工智能技术的思想，而人工智能最初是用于故事理解和知识工程而开发的(Carbonell et al., 1981)。这种对基于意义的技术的兴趣也是对那个时代计算语言学中以句法为核心的反映。采用中间语模型的另一个动机是他们的似乎合理的反省：MT 系统应该像人一样进行翻译工作的思想（假定人们是通过利用他们的理解能力来进行翻译的）。那时的反省可能是一种误解，因为人类的翻译处理是非常复杂的，并且对机器翻译的能力并不清楚。在20世纪80年代晚期和20世纪90年代早期有不少对这类问题的关注，例如文献 Tsujii(1986)，Nirenburg et al.(1992)，Ward(1994)和 Carbonell et al.(1992)。同时，受全球贸易不断增长，需要将所有文件翻译为多种官方语言的政府政策的增加，文字的电子处理的激增和个人计算机以及互联网的普及的影响，MT 的使用在不断地增加。

在20世纪90年代，我们看到了基于大规模语料库的统计方法的应用。建立语音到语音的翻译系统带给我们“巨大挑战”的兴奋(Kay et al., 1992; Bub et al., 1997; Frederking et al., 1997)，这里 MT 与计算机的现代版本一起被嵌入我们的生活，无所不在，彼此交互。在实践中，随着用户群的增加，用户的需要对 MT 的研究和开发越来越起到决定性的作用。

Hutchins(1986, 1997)给出了对早期 MT 历史的精彩综述。Hutchins and Somers(1992)包括使翻译困难的语言现象的丰富样例，并且包括对一些历史上著名的 MT 系统的详尽描述。

有关机器翻译的学术论文可以在期刊 Machine Translation 和两年一届（奇数年）的会议 Theoretical and Methodological Issue in Machine Translation 中找到。

有关系统、市场和用户经验的报告可以在 MT News International，以及国际机器翻译学会（International Association for Machine Translation）的时事通讯中看到。国际机器翻译学会下设的分会包括：美国的 MT 学会、亚洲太平洋地区的 MT 学会和欧洲的 MT 学会。这些学会每年都召开能够将开发者与用户召集在一起的年会。通常会出版两年一期的 MT Summit（在奇数年出版）会议论文集。计算语言学的主流期刊和会议也都会不时地出现关于机器翻译的报告。

附录 A　正则表达式的算符

Perl	grep	MS Word	说明
		简单字符表达式	
\..	\..	\..	一个特殊字符
.	.	?	任何单个字符
[...]	[...]	[...]	列出的任何单独字符
[...-...]	[...-...]	[...-...]	在范围中的任何单独字符
[^...]	[^...]	[!...]	没有列出的任何单独字符
[^...-...]	[^...-...]	[!...-...]	不在范围中的任何单独字符
		用于匹配位置的锚号 / 表达式	
^	^	^	行的开始
$	$	$	行的结束
\b	–	–	词界
\B	–	–	非词界
-	\<	<	词的开始
-	\>	>	词的结束
		用于说明前面表达式的计数符 / 表达式	
*	*	–	前面零个或多个正则表达式
+	–	@	前面一个或多个正则表达式
?	–	–	前面只有一个或零个正则表达式
{n}	\{n\}	{n}	前面 *n* 个正则表达式
{n, m}	\{n, m\}	{n, m}	前面 *n*~*m* 个正则表达式
{n, }	\{n, \}	{n, }	前面至少 *n* 个正则表达式

图 A.1　基本正则表达式

Perl	grep	MS Word	说明
		其他	
.*	.*	*	任何字符串
... | ...	–	–	析取，- 匹配正则表达式
(...)	\(... \)	(...)	分组，存储
		简写法	
\d	[0-9]	[0-9]	任何数字字符
\D	[^0-9]	[^0-9]	任何非数字字符
\w	[a-zA-Z0-9_]	[a-zA-Z0-9_]	任何字母数字字符或空白
\W	[^a-zA-Z0-9_]	[^a-zA-Z0-9_]	任何非字母数字字符
\s	[\r\t\n\f]	–	空白区域（空白，表格）
\S	[^ \r\t\n\f]	–	非空白区域

图 A.2　更多的正则表达式

附录B　PORTER STEMMING算法

为了便于描述Porter(1980)的算法，我们把辅音（consonant）定义为除了A，E，I，O和U以及前面出现辅音的Y之外的字母。其他任何字母都称为元音（vowel）（这样的定义当然只是在正词法上的近似定义）。令c表示一个辅音，v表示一个元音。令C表示包含一个或多个辅音的串，V表示包含一个或多个元音的串。这样，任何书面英语的词或者词的某个部分，都可以用下面的正则表达式来表示（这里，圆括号( )表示可选的成分）：

$$(C)(VC)^m(V)$$

例如，单词 *trouble* 可以映射为如下的符号序列：

```
troubles
C V C VC
```

这个符号序列没有最后的元音V。我们把Kleene算符*m*称为单词或词的某个部分测度值（measure），这个测度值大致相当于单词或词的某个部分中的音节数目。例如：

m=0	TR, EE, TREE, Y, BY
m=1	TROUBLE, OATS, TREES, IVY
m=2	TROUBLES, PRIVATE, OATEN, ORRERY

下面的规则全都具有如下的格式：

```
(condition) S1 → S2
```

这个规则的含义是："如果某个单词以S1结尾，并且S1之前的词干满足括号中的条件，那么就用S2来替换S1"。所谓"条件"包括下面的各项以及它们的布尔组合：

m	词干的测度值
*S	以S（以及其他相似字母）为结尾的词干
v	包括元音的词干
*d	以双重辅音（例如-TT，-SS）为结尾的词干
*o	以CVC为结尾的词干，其中第二个c不是W，X或Y（例如-WIL，-HOP）

Porter算法包括按顺序使用的7个简单规则集。在每一步时，如果存在一个以上的规则可以使用，那么只使用具有最长匹配后缀的规则。

步骤1：关于复数名词和单数第三人称动词的规则

该规则集没有条件：

SSES	→	SS	caresses	→	caress
IES	→	I	ponies	→	poni
			ties	→	ti
SS	→	SS	caress	→	caress
S	→	ε	cats	→	cat

步骤 2a：关于动词过去时态和进行式的规则

(m> 1) EED → EE	feed → feed agreed → agree
(*v*) ED → ε	plastered → plaster bled → bled
(*v*) ING → ε	motoring → motor sing → sing

步骤 2b：清除

如果步骤 2a 中的第 2 条和第 3 条规则使用成功，那么再运行下面的规则（这个规则去掉双重字母，把 E 置于 AT/BL 之后，成为 ATE/BLE）。

AT → ATE	conflat(ed) → conflate
BL → BLE	troubl(ing) → trouble
IZ → IZE	siz(ed) → size
(*d & !(*L or *S or *Z)) → single letter	hopp(ing) → hop tann(ed) → tan fall(ing) → fall hiss(ing) → hiss fizz(ed) → fizz
(m=1 & *o) → E	fail(ing) → fail fil(ing) → file

步骤 3：Y → I

(*v*) Y → I	happy → happi sky → sky

步骤 4：关于派生形态学的规则 1：多后缀

(m > 0)	ATIONAL	→	ATE	relational	→ relate
(m > 0)	TIONAL	→	TION	conditional	→ condition
				rational	→ rational
(m > 0)	ENCI	→	ENCE	valenci	→ valence
(m > 0)	ANCI	→	ANCE	hesitanci	→ hesitance
(m > 0)	IZER	→	IZE	digitizer	→ digitize
(m > 0)	ABLI	→	ABLE	conformabli	→ conformable
(m > 0)	ALLI	→	AL	radicalli	→ radical
(m > 0)	ENTLI	→	ENT	differentli	→ different
(m > 0)	ELI	→	E	vileli	→ vile
(m > 0)	OUSLI	→	OUS	analogousli	→ analogous
(m > 0)	IZATION	→	IZE	vietnamization	→ vietnamize
(m > 0)	ATION	→	ATE	predication	→ predicate
(m > 0)	ATOR	→	ATE	operator	→ operate
(m > 0)	ALISM	→	AL	feudalism	→ feudal
(m > 0)	IVENESS	→	IVE	decisiveness	→ decisive
(m > 0)	FULNESS	→	FUL	hopefulness	→ hopeful
(m > 0)	OUSNESS	→	OUS	callousness	→ callous
(m > 0)	ALITI	→	AL	formaliti	→ formal
(m > 0)	IVITI	→	IVE	sensitiviti	→ sensitive
(m > 0)	BILITI	→	BLE	sensibiliti	→ sensible

步骤 5：关于派生形态学的规则 II：其他多后缀

(m > 0)	ICATE	→	IC	triplicate	→	triplic
(m > 0)	ATIVE	→	ε	formative	→	form
(m > 0)	ALIZE	→	AL	formalize	→	formal
(m > 0)	ICITI	→	IC	electriciti	→	electric
(m > 0)	FUL	→	ε	hopeful	→	hope
(m > 0)	NESS	→	ε	goodness	→	good

步骤 6：关于派生形态学的规则 III：单后缀

(m > 1)	AL	→	ε	revival	→	reviv
(m > 1)	ANCE	→	ε	allowance	→	allow
(m > 1)	ENCE	→	ε	inference	→	infer
(m > 1)	ER	→	ε	airliner	→	airlin
(m > 1)	IC	→	ε	gyroscopic	→	gyroscop
(m > 1)	ABLE	→	ε	defensible	→	defens
(m > 1)	ANT	→	ε	irritant	→	irrit
(m > 1)	EMENT	→	ε	replacement	→	replac
(m > 1)	MENT	→	ε	adjustment	→	adjust
(m > 1)	ENT	→	ε	dependent	→	depend
(m > 1)	(*S or *T) & ION	→	ε	adoption	→	adopt
(m > 1)	OU	→	ε	homologou	→	homolog
(m > 1)	ISM	→	ε	communism	→	commun
(m > 1)	ATE	→	ε	activate	→	activ
(m > 1)	ITI	→	ε	angulariti	→	angular
(m > 1)	OUS	→	ε	homologous	→	homolog
(m > 1)	IVE	→	ε	effective	→	effect
(m > 1)	IZE	→	ε	bowdlerize	→	bowdler

步骤 7a：清除

(m > 1)	E	→	ε	probate	→	probat
				rate	→	rate
(m = 1 & ! *o)	E	→	ε	cease	→	ceas

步骤 7b：清除

(m > 1 & *d *L)	→	[single letter]	controll	→	control
			roll	→	roll

附录C　标记集C5和C7

标记	说明	例子
AJ0	adjective (unmarked)	*good, old*
AJC	comparative adjective	*better, older*
AJS	superlative adjective	*best, oldest*
AT0	article	*the, a, an*
AV0	adverb (unmarked)	*often, well, longer, furthest*
AVP	adverb particle	*up, off, out*
AVQ	wh-adverb	*when, how, why*
CJC	coordinating conjunction	*and, or*
CJS	subordinating conjunction	*although, when*
CJT	the conjunction *that*	
CRD	cardinal numeral (except *one*)	*3, twenty-five, 734*
DPS	possessive determiner	*your, their*
DT0	general determiner	*these, some*
DTQ	wh-determiner	*whose, which*
EX0	existential *there*	
ITJ	interjection or other isolate	*oh, yes, mhm*
NN0	noun (neutral for number)	*aircraft, data*
NN1	singular noun	*pencil, goose*
NN2	plural noun	*pencils, geese*
NP0	proper noun	*London, Michael, Mars*
ORD	ordinal	*sixth, 77th, last*
PNI	indefinite pronoun	*none, everything*
PNP	personal pronoun	*you, them, ours*
PNQ	wh-pronoun	*who, whoever*

图C.1　英国国家语料库（BNC）的标记集UCREL C5的前一半(Garside et al.,1997)

标记	说明	例子
PNX	reflexive pronoun	*itself, ourselves*
POS	possessive *'s* or '	
PRF	the preposition *of*	
PRP	preposition (except *of*)	*for, above, to*
PUL	punctuation – left bracket	(or [
PUN	punctuation – general mark	. ! , : ; - ? ...
PUQ	punctuation – quotation mark	‘ ’ ”
PUR	punctuation – right bracket	) or]
TO0	infinitive marker *to*	
UNC	unclassified items (not English)	
VBB	base forms of *be* (except infinitive)	*am, are*
VBD	past form of *be*	*was, were*
VBG	-ing form of *be*	*being*
VBI	infinitive of *be*	
VBN	past participle of *be*	*been*
VBZ	-s form of *be*	*is, 's*
VDB	base form of *do*(except infinitive)	*does*

图C.2　标记集UCREL C5的其他部分(Garside et al., 1997)

标记	说明	例子
VDD	past form of *do*	*did*
VDG	-ing form of *do*	*doing*
VDI	infinitive of *do*	*to do*
VDN	past participle of *do*	*done*
VDZ	-s form of *do*	*does*
VHB	base form of *have* (except infinitive)	*have*
VHD	past tense form of *have*	*had, 'd*
VHG	-ing form of *have*	*having*
VHI	infinitive of *have*	
VHN	past participle of *have*	*had*
VHZ	-s form of *have*	*has, 's*
VM0	modal auxiliary verb	*can, could, will, 'll*
VVB	base form of lexical verb (except infin.)	*take, live*
VVD	past tense form of lexical verb	*took, lived*
VVG	-ing form of lexical verb	*taking, living*
VVI	infinitive of lexical verb	*take, live*
VVN	past participle form of lex. verb	*taken, lived*
VVZ	-s form of lexical verb	*takes, lives*
XX0	the negative *not* or *n't*	
ZZ0	alphabetical symbol	*A, B, c, d*

图C.2（续）　标记集UCREL C5的其他部分(Garside et al., 1997)

标记	说明	例子
!	punctuation tag - exclamation mark	
"	punctuation tag - quotation marks	
(	punctuation tag - left bracket	
)	punctuation tag - right bracket	
,	punctuation tag - comma	
-	punctuation tag - dash	
——	new sentence marker	
.	punctuation tag - full-stop	
...	punctuation tag - ellipsis	
:	punctuation tag - colon	
;	punctuation tag - semi-colon	
?	punctuation tag - question-mark	
APPGE	possessive pronoun, prenominal	*my, your, our* etc.
AT	article	*the, no*
AT1	singular article	*a, an, every*
BCL	before-clause marker	*in order [that]*
CC	coordinating conjunction	*and, or*
CCB	coordinating conjunction	*but*
CS	subordinating conjunction	*if, because, unless*
CSA	*as* as a conjunction	
CSN	*than* as a conjunction	
CST	*that* as a conjunction	
CSW	*whether* as a conjunction	
DA	post-determiner/pronoun	*such, former, same*
DA1	singular after-determiner	*little, much*
DA2	plural after-determiner	*few, several, many*
DAR	comparative after-determiner	*more, less*
DAT	superlative after-determiner	*most, least*
DB	pre-determiner/pronoun	*all, half*
DB2	plural pre-determiner/pronoun	*both*
DD	determiner/pronoun	*any, some*
DD1	singular determiner	*this, that, another*
DD2	plural determiner	*these, those*
DDQ	wh-determiner	*which, what*

图C.3　英国国家语料库（BNC）的标记集UCREL C7的前面部分(Garside et al., 1997)

标记	说明	例子
DDQGE	wh-determiner, genitive	*whose*
DDQV	wh-ever determiner	*whichever, whatever*
EX	existential *there*	
FO	formula	
FU	unclassified	

图 C.3（续） 英国国家语料库（BNC）的标记集 UCREL C7 的前面部分(Garside et al., 1997)

标记	说明	例子
FW	foreign word	
GE	germanic genitive marker -	' or *'s*
IF	*for* as a preposition	
II	preposition	*in, on, to*
IO	*of* as a preposition	
IW	*with*; *without* as preposition	
JJ	general adjective	*big, old*
JJR	general comparative adjective	*older, better, bigger*
JJT	general superlative adjective	*oldest, best, biggest*
JK	adjective catenative	*able* in *be able to*
		willing in *be willing to*
MC	cardinal number (neutral for number)	*two, three...*
MC1	singular cardinal number	*one*
MC2	plural cardinal number	*tens, twenties*
MCMC	hyphenated number	*40-50, 1770-1827*
MD	ordinal number	*first, 2nd, next, last*
ND1	singular noun of direction	*north, southeast*
NN	common noun (neutral for number)	*sheep, cod*
NN1	singular common noun	*book, girl*
NN2	plural common noun	*books, girls*
NNA	following noun of title	*M.A.*
NNB	preceding noun of title	*Mr, Prof*
NNL1	singular locative noun	*street, Bay*
NNL2	plural locative noun	*islands, roads*
NNO	numeral noun (neutral for number)	*dozen, thousand*
NNO2	plural numeral noun	*hundreds, thousands*
NNT	temporal noun (neutral for number)	no known examples
NNT1	singular temporal noun	*day, week, year*
NNT2	plural temporal noun	*days, weeks, years*
NNU	unit of measurement (neutral for number)	*in., cc.*
NNU1	singular unit of measurement	*inch, centimetre*
NNU2	plural unit of measurement	*inches, centimetres*
NP	proper noun (neutral for number)	*Phillipines, Mercedes*
NP1	singular proper noun	*London, Jane, Frederick*
NP2	plural proper noun	*Browns, Reagans, Koreas*
NPD1	singular weekday noun	*Sunday*
NPD2	plural weekday noun	*Sundays*

图 C.4 标记集 UCREL C7 的更多部分(Garside et al., 1997)

标记	说明	例子
NPM1	singular month noun	*October*
NPM2	plural month noun	*Octobers*
PN	indefinite pronoun (neutral for number)	*none*
PN1	singular indefinite pronoun	*one, everything, nobody*
PNQO		*whom*
PNQS		*who*
PNQV		*whoever, whomever*
		whomsoever, whosoever

图 C.5 标记集 UCREL C7 的更多部分(Garside et al., 1997)

标记	说明	例子
PNX1	reflexive indefinite pronoun	*oneself*
PPGE	nominal possessive personal pronoun	*mine, yours*
PPH1		*it*
PPHO1		*him, her*
PPHO2		*them*
PPHS1		*She, she*
PPHS2		*they*
PPIO1		*me*
PPIO2		*us*
PPIS1		*I*
PPIS2		*we*
PPX1	singular reflexive personal pronoun	*yourself, itself*
PPX2	plural reflexive personal pronoun	*yourselves, ourselves*
PPY		*you*
RA	adverb, after nominal head	*else, galore*
REX	adverb introducing appositional constructions	*namely, viz, eg.*
RG	degree adverb	*very, so, too*
RGQ	wh- degree adverb	*how*
RGQV	wh-ever degree adverb	*however*
RGR	comparative degree adverb	*more, less*
RGT	superlative degree adverb	*most, least*
RL	locative adverb	*alongside, forward*
RP	prepositional adverb; particle	*in, up, about*
RPK	prepositional adverb, catenative	*about* in *be about to*
RR	general adverb	*actually*
RRQ	wh- general adverb	*where, when, why, how*
RRQV	wh-ever general adverb	*wherever, whenever*
RRR	comparative general adverb	*better, longer*
RRT	superlative general adverb	*best, longest*
RT	nominal adverb of time	*now, tommorow*

图C.5（续）　标记集UCREL C7的更多部分(Garside et al., 1997)

标记	说明	例子
TO	infinitive marker	*to*
UH	interjection	*oh, yes, um*
VB0		*be*
VBDR		*were*
VBDZ		*was*
VBG		*being*
VBI	infinitive *be*	
VBM		*am*
VBN		*been*
VBR		*are*
VBZ		*is*
VD0		*do*
VDD		*did*
VDG		*doing*
VDI	infinitive *do*	
VDN		*done*
VDZ		*does*
VH0		*have*
VHD	past tense *had*	
VHG		*having*
VHI	infinitive *have*	
VHN	past participle *had*	
VHZ		*has*
VM	modal auxiliary	*can, will, would* etc.
VMK	modal catenative	*ought, used*

图C.6　标记集UCREL C7的其余部分(Garside et al., 1997)

标记	说明	例子
VV0	base form of lexical verb	*give*, *work* etc.
VVD	past tense form of lexical verb	*gave*, *worked* etc.
VVG	-ing form of lexical verb	*giving*, *working* etc.
VVGK	-ing form in a catenative verb	*going* in *be going to*
VVI	infinitive of lexical verb	*[to] give*, *[to] work* etc.
VVN	past participle form of lexical verb	*given*, *worked* etc.
VVNK	past part. in a catenative verb	*bound* in *be bound to*
VVZ	-s form of lexical verb	*gives*, *works* etc.
XX		*not*, *n't*
ZZ1	singular letter of the alphabet	*A*, *a*, *B*, etc.
ZZ2	plural letter of the alphabet	*As*, *b's*, etc.

图C.6（续） 标记集UCREL C7的其余部分(Garside et al., 1997)

附录 D　HMM 模型的训练：向前 - 向后算法

这个附录扼要地介绍向前 - 向后算法（forward-backward algorithm），或称 Baum-Welch 算法 (Baum, 1972)，这是期望最大化算法（Expectation-Maximization algorithm，简称EM算法）(Dempster et al., 1977)的一种特殊情形。这个算法将帮助我们训练 HMM 的转移概率 a_{ij} 和发射概率 $b_i(o_t)$。虽然在理论上训练HMM的这两个网络结构都是可能的，但现在还没有能够进行这种双重归纳的优秀算法。因此，实际上大多数 HMM 的结构是手工设计的，转移概率和发射概率都是从观察序列 O 的一个很大的集合训练得出的。另外，如何设置参数 a 和 b，使观察序列 O 的概率恰好达到最大值，还是一个没有解决的问题。这里给出的算法只能保证我们发现局部的最大值。向前 - 向后算法广泛用于语音和语言处理中，例如在第 8 章中用这个算法来训练基于 HMM 的词类标注系统。向前 - 向后算法的扩充也很重要，例如可以利用向内向外算法来训练随机上下文无关文法（见第 12 章）。

开始时我们可以这样想像：我们训练的不是一个隐马尔可夫模型，而是一个普通马尔可夫模型（vanilla Markov Model）。训练时，在观察序列上运行这个模型，并且了解究竟使用了哪种转移和观察。为了易于描述，在其他部分假定只对训练数据中的一个单独序列进行训练（称为 O），但在真正的语音识别系统中要训练成百上千的序列（几千个句子）。与 HMM 不同，普通马尔可夫模型不是隐藏的，因此可以看到观察序列，确切地知道通过模型时要进行哪个转换，以及生成每个观察符号时需要经过什么样的状态。因为每个状态只能生成一个观察符号，所以观察 b 的概率总和应该为 1.0。在状态 i 和状态 j 之间的一个特定转移的概率 a_{ij} 可以通过转移的次数来计算，我们把它记为 $C(i \rightarrow j)$，然后可以用从状态 i 开始的所有转移次数来对它进行归一化：

$$a_{ij} = \frac{C(i \rightarrow j)}{\sum_{q \in Q} C(i \rightarrow q)} \tag{D.1}$$

对于 HMM 来说，我们不能直接从所观察的句子（或句子的集合）直接计数，因为不知道对于一个给定的输入，通过机器的状态究竟走哪条路径。Baum-Welch 提出了用两个符合直觉的思想来解决这个问题。第一个思想是，反复地估计所得的计数。我们从转移概率和观察概率的一个估计值开始，然后反复使用这些估计概率来推出越来越好的概率。第二个思想是，对于一个观察，计算它的向前概率，从而得到估计概率，然后，把这个概率量在对于向前概率有贡献的所有不同路径上进行分摊。

为了理解这个算法，我们需要回顾第 5 章的向前算法，并且更形式化地定义两个有关的概率，它们将被用来计算最后的概率，这两个概率是向前概率（forward probability）和向后概率（backward probability）。我们把向前概率记为 α，把向后概率记为 β。以前讲过，向前概率是对于给定的自动机 λ 在观看了前面 t 个观察之后在状态 i 的概率：

$$\alpha_t(i) = P(o_1, o_2 \cdots o_t, q_t = i|\lambda) \tag{D.2}$$

在第 5 章中曾使用矩阵来递归地计算向前概率，现在来形式化地定义这样的递归。

1. 初始化：

$$\alpha_h(1) = a_{1j} b_j(o_1) \quad 1 < j < N \tag{D.3}$$

2. 递归迭代（因为状态 1 和 N 是非发射的）：

$$\alpha_j(t) = \left[\sum_{i=2}^{N-1} \alpha_i(t-1)\, a_{ij}\right] b_j(o_t) \quad 1 < j < N, 1 < t < T \tag{D.4}$$

3. 终结：

$$P(O|\lambda) = \alpha_N(T) = \sum_{i=2}^{N-1} \alpha_i(T)\, a_{iN} \tag{D.5}$$

正如在第 5 章中看到的，向前概率通过矩阵或格（lattice）来计算，其中每个列是通过扩充前面的列的路径的方法来计算的。图 D.1 说明了计算新单元中的值的归纳步骤。

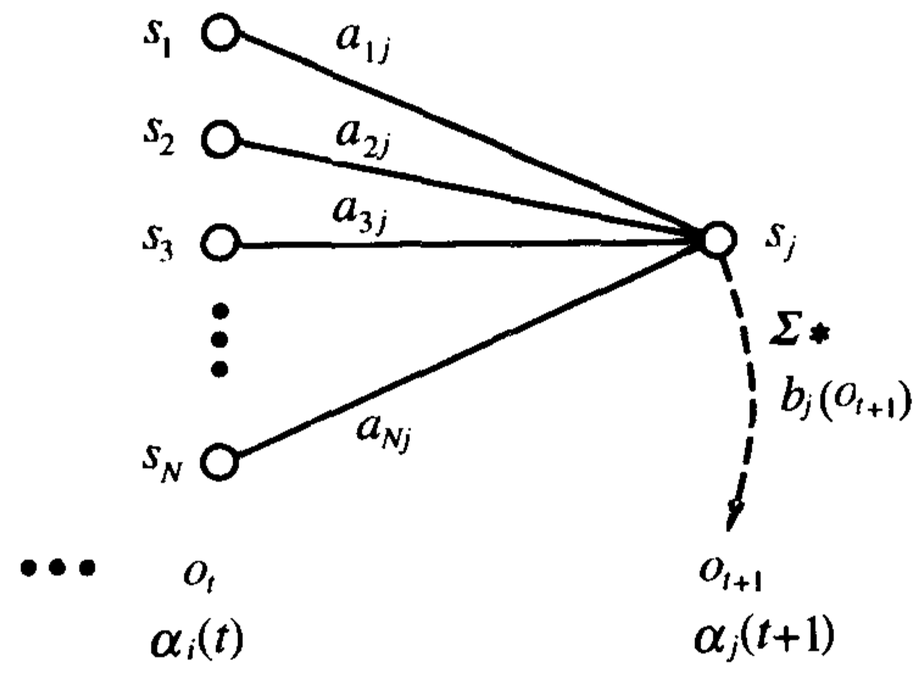


图 D.1 把前面所有的值 α_{i-1} 相加来计算 $\alpha_i(t)$，计算时要使用它们的转换概率 a 来加权，并乘以观察概率 $b_i(o_{t+1})$。当然，在任何给定的 HMM 中，很多转移概率或者大多数转移概率都为零，所以并不是前面的所有状态对当前状态的向前概率都有贡献

向前-向后算法的另一个重要部分是向后概率，它几乎是向前概率的镜像。向后概率要计算在状态 j 和时刻 t 从时刻 $t+1$ 到终点的观察概率（当然是对于给定的自动机 λ）：

$$\beta_i(o_t) = P(o_{t+1}, o_{t+2} \cdots o_T | q_t = j, \lambda) \tag{D.6}$$

与向前概率一样，向后概率也使用归纳方法来进行计算。

1. 初始化：

$$\beta_i(t) = a_{iN}, \quad 1 < i < N \tag{D.7}$$

2. 递归迭代（因为状态 1 和 N 也是非发射的）：

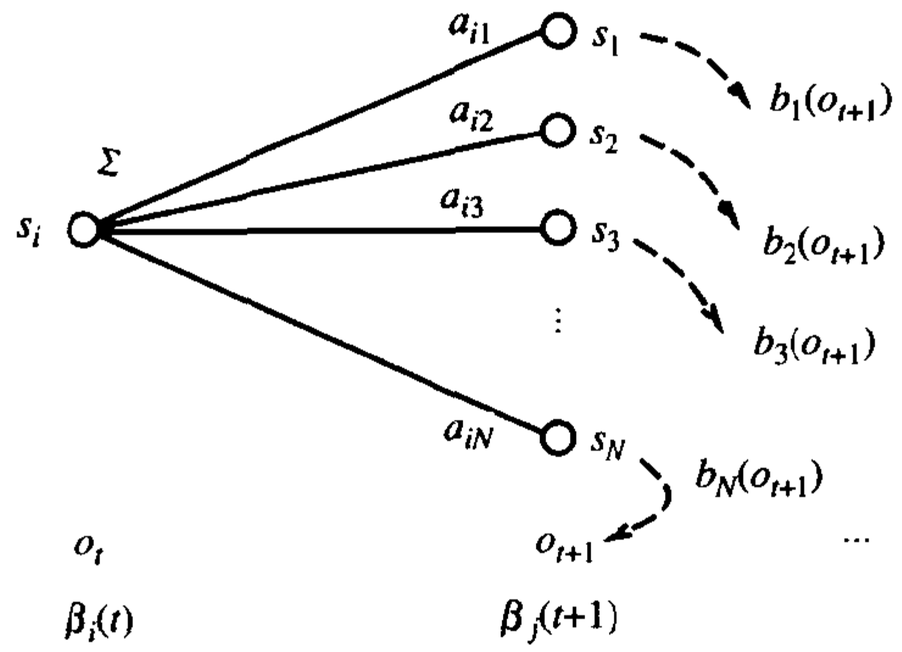
$$\beta_i(t) = \sum_{i=2}^{N-1} a_{ij} b_j(o_{t+1}) \beta_j(t+1) \quad 1 < i < N, T > t \geqslant 1 \tag{D.8}$$

3. 终结：

$$P(O|\lambda) = \alpha_N(T) = \beta_1(T) = \sum_{j=2}^{N-1} a_{1j} b_j(o_1) \beta_j(1) \tag{D.9}$$

图 D.2 说明了向后归纳的步骤。

现在来解释向前和向后概率怎样帮助从一个观察序列计算转移概率 a_{ij} 和观察概率 $b_i(o_t)$，尽管实际上通过机器的路径是隐藏的。



图D.2　把所有β_{i+1}的值连续相加来计算$\beta_i(t)$，计算时用它们的转移概率a和观察概率$b_i(o_{i+1})$来加权

首先来说明如何重新估计a_{ij}。我们把式（D.1）改变为另一种形式来估计$\hat{a}_{ij}$:

$$\hat{a}_{ij} = \frac{\text{从状态}i\text{转移到状态}j\text{的期望次数}}{\text{由状态}i\text{转移出的期望次数}} \tag{D.10}$$

怎样计算分子呢？这里是根据直觉来计算的。假定对于给定的转移$i \to j$在观察序列中特定时刻t发生这个事件有某个概率估计，如果对于每个特定时刻t都知道这个概率，就可以把所有时刻t加起来求和，从而估计出转移$i \to j$总计数。

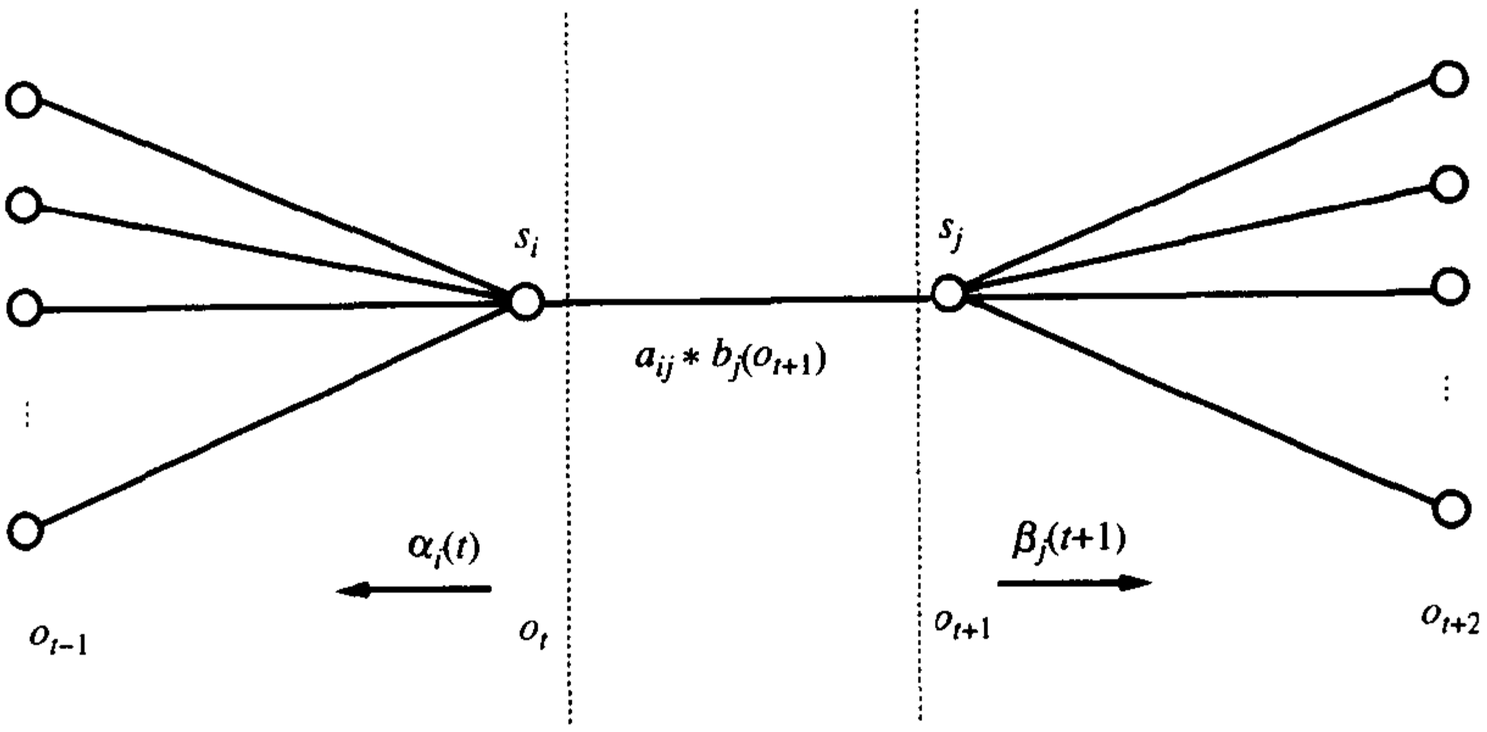
更形式化地说，对于给定的观察序列和模型，让我们把概率τ_t（τ表示转移）定义为在时刻t状态为i且在时刻t+1状态为j的概率：

$$\tau_t(i,j) = P(q_t = i, q_{t+1} = j | O, \lambda) \tag{D.11}$$

为了计算τ_t，首先计算近似于τ_t的概率，我们把它记为not-quite-τ_t，这个概率与τ_t的不同之处是还包含观察概率：

$$\text{not-quite-}\tau_t(i,j) = P(q_t = i, q_{t+1} = j, O | \lambda) \tag{D.12}$$

图D.3说明了用来计算not-quite-τ_t的各个概率：在有关弧上的转移概率，在该弧之前的概率α，在该弧之后的概率β，恰恰在该弧之后的符号的观察概率。



图D.3　计算在时刻t状态为i且在时刻t+1状态为j的联合概率。图中说明了需要结合起来产生概率$P(q_t = i, q_{t+1} = j, O|\lambda)$的各个概率：概率$\alpha$和$\beta$，转移概率$a_{ij}$，观察概率$b_j(o_t+1)$(Rabiner，1989)

把这些概率相乘就得到了 not-quite-τ_t[①]，计算公式如下：

$$\text{not-quite-}\tau_t(i,j) = \alpha_i(t)\, a_{ij} b_j(o_{t+1})\beta_j(t+1) \tag{D.13}$$

为了从 not-quite-τ_t 来计算 τ_t，概率定理告诉我们可以用 $P(O|\lambda)$来除，因为：

$$P(X|O,\lambda) = \frac{P(X,O|\lambda)}{P(O|\lambda)} \tag{D.14}$$

对于给定的模型，观察概率就是整个语段的向前概率（或者换一种说法，整个语段的向后概率），因此它可以有许多方法来计算：

$$P(O|\lambda) = \alpha_N(T) = \beta_1(T) = \sum_{j=1}^{N} \alpha_j(t)\beta_j(t) \tag{D.15}$$

这样，计算 τ_t 的最后等式就是：

$$\tau_t(i,j) = \frac{\alpha_i(t)\, a_{ij} b_j(o_{t+1})\beta_j(t+1)}{\alpha_N(T)} \tag{D.16}$$

从状态 i 转移到状态 j 的期望次数就是 τ 在所有 t 上的总和。对于式（D.10）中 a_{ij} 的估计，现在只再需要一项，这就是由状态 i 转移出的所有期望次数。把从状态 i 转移出的所有期望次数加起来就可以得到它。下面是 $\hat{a}_{ij}$ 的最后的计算公式：

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \tau_t(i,j)}{\sum_{t=1}^{T-1}\sum_{j=1}^{N} \tau_t(i,j)} \tag{D.17}$$

我们还需要一个重新计算观察概率的公式。这是对于给定的状态 j 和对于观察词汇 V 中的一个给定符号 v_k 的概率，记为 $\hat{b}_j(v_k)$。

$$\hat{b}_j(v_k) = \frac{\text{在状态}j\text{和观察符号}v_k\text{的期望次数}}{\text{在状态}j\text{的期望次数}} \tag{D.18}$$

为此，需要知道在时刻 t 和状态 j 的概率，我们把这个概率记为 $\sigma_j(t)$（σ 表示状态）：

$$\sigma_j(t) = P(q_t = j|O,\lambda) \tag{D.19}$$

这里需要再次把观察序列包括到概率中来进行计算：

$$\sigma_j(t) = \frac{P(q_t = j, O|\lambda)}{P(O|\lambda)} \tag{D.20}$$

正如图 D.4 所说明的，式（D.20）中的分子部分等于向前概率和向后概率的乘积：

$$\sigma_j(t) = \frac{\alpha_j(t)\beta_j(t)}{P(O|\lambda)} \tag{D.21}$$

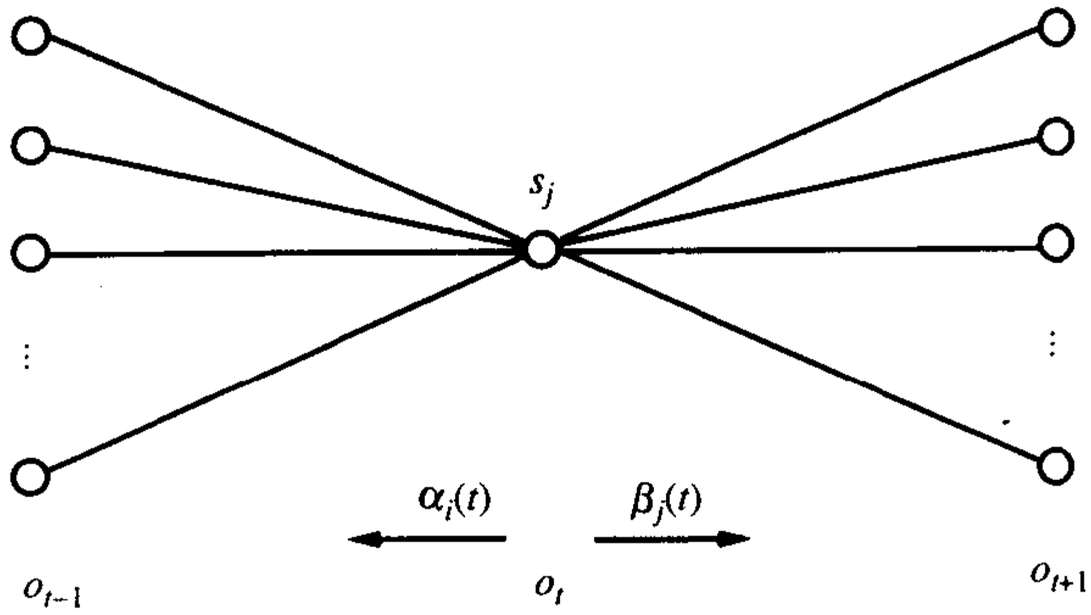
现在我们准备计算 b。对于分子部分，对所有时间步骤 t 求总和 $\sigma_j(t)$。其中，观察 o_t 就是我们感兴趣的符号 v_k。对于分母部分，对所有的时间步骤 t 求总和 $\sigma_j(t)$。其结果是，当在状态 j 看到符号 v_k 的时间的百分比（记号 $\sum_{t=1 s.t. O_t=v_k}^{T}$ 的意思是“在时刻 t 的观察为 v_k 时的所有时间上的总和”）：

$$\hat{b}_j(v_k) = \frac{\sum_{t=1 s.t. O_t=v_k}^{T} \sigma_j(t)}{\sum_{t=1}^{T} \sigma_j(t)} \tag{D.22}$$

对于一个观察 O，假定已经有了转移概率 a 和观察概率 b 的初始估计，现在有了一种方法来重新估计转移概率 a 和观察概率 b 的值。HMM 的整个训练过程称为嵌入训练（embedded training），在

① 此处原文为 not-quite-*τ_t，可能有误。——译者注

嵌入训练中首先选择对于a和b的某些估计值，然后反复迭代使用式（D.22）和式（D.17）进行重新估值，直到a和b的值收敛于一个最优点为止。在下一节中将会看到，怎样使用高斯函数把向前－向后算法推广到非离散（连续观察强度）的情形。7.7节讨论了这个嵌入训练算法怎样得到a和b的初始估计。



图D.4　计算在时刻t和状态j的概率$\sigma_j(t)$。注意，这里的σ实际上是t的一种退化的情况。因此，这个图就像把图D.3中的状态i和状态j折叠起来而形成的

连续概率强度

前面各节中描述进行参数重新估计的方法时都做了这样的假设：输入观察来自一个规模可观的字母表中的离散符号。这样的假定在HMM的某些应用中是没有问题的，例如第8章介绍了如何应用HMM来进行词类标注，处理的对象就是离散符号。在词类标注中，观察就是英语的单词，它们是一个规模可观的有限集合，大约有10万（100 K）单词。对于语音识别，我们介绍过的LPC倒谱特征构成了一个很大的字母表（11个特征，每个特征有32比特浮点数），总的词汇容量达$2^{(11\times 32)}$。事实上，由于在实际应用中通常用的不是11个特征，而是增量特征（delta-features）或双增量特征（double-delta-features），所以实际的词汇容量将非常大。在第7章中提到，解决这个问题的一个办法是聚类（cluster）和矢量量化（vector quantize），把倒谱特征变成较小的离散观察符号的集合。在第7章中曾经提过其他更有效的方法，或者使用高斯估计神经网络（多层感知器）混合的方法来估计概率强度函数，或者在一个连续空间上使用概率强度函数（pdf）的方法。

使用高斯观察概率估计的HHM只要简单地扩充向前－向后算法就可以训练出来。在第7章中曾经提过高斯模型的最简单的使用方法，我们假定观察特征矢量的所有可能值o_t是成正态分布的，所以可以把观察概率函数$b_j(o_t)$表示为平均矢量为μ_j，协方差矩阵为Σ_j的高斯曲线（下标表示矢量转置）：

$$b_j(o_t)=\frac{1}{\sqrt{(2\pi)|\sum j|}}e^{[(o_t-\mu_j)'\Sigma_j^{-1}(o_t-\mu_j)]} \tag{D.23}$$

为了简化起见，通常我们假定协方差矩阵Σ_j是对角矩阵，这意味着实际上在特征矢量中，对于每个特征都只保持一个单独的平均值和方差。

高斯估计中的平均值和协方差怎样计算呢？考虑只有一个状态i的非隐马尔可夫模型的最简单情况是会有帮助的。特征平均矢量μ和协方差矢量Σ可以通过求平均来估计：

$$\hat{\mu}_i=\frac{1}{T}\sum_{t=1}^{T}o_t \tag{D.24}$$

$$\hat{\Sigma}_i=\frac{1}{T}\sum_{t=1}^{T}[(o_t-\mu_j)'(o_t-\mu_j)] \tag{D.25}$$

但是，因为存在着多个隐藏的状态，我们不知道哪个观察矢量o_t是由哪个状态产生的。我们想做的事就是给每个可能的状态i分别指派一个观察矢量o_t，然后再给它们分摊在状态i和时刻t的HMM的概率。幸运的是，我们已经知道了怎样来进行这样的分摊；在状态i和时刻t的HMM的概率就是$\sigma_i(t)$，前面已经知道怎样计算它了！当然，我们需要反复迭代来计算概率$\sigma_i(t)$，因为如果得到了比较好的观察概率b，将有助于更确信在特定状态和特定时刻的概率σ。这样，实际的再估计等式如下所示：

$$\hat{\mu}_i = \frac{\sum_{t=1}^{T} \sigma_i(t) o_t}{\sum_{t=1}^{T} \sigma_i(t)} \tag{D.26}$$

$$\hat{\Sigma}_i = \frac{\sum_{t=1}^{T} \sigma_i(t)(o_t - \mu_i)'(o_t - \mu_i)}{\sum_{t=1}^{T} \sigma_i(t)} \tag{D.27}$$

分母的求和与式（D.22）中分母的求和是相同的。因此，式（D.26）和式（D.27）可使用于训练HMM的向前－向后（Baum-Welch）算法中。首先对某些初始的估计置μ_i和σ_i的值，然后反复估计，一直到数值收敛为止。

关于向前－向后算法的更全面的描述，可参阅Jelinek(1997)或Rabiner and Juang(1993)。Jelinek(1997)也说明了向前－向后算法和EM之间的关系。

参考文献

缩写及符号说明

AAAI-XX	Proceedings of the Yth National Conference on Artificial Intelligence(in year XX) 第 Y 届美国人工智能学会会议论文集（XX 年）
ACL-XX	Proceedings of the Yth Annual Conference of the Association for Computational Linguistics(in year XX) 第 Y 届美国计算语言学会年会论文集（XX 年）
CLS-XX	Papers from the Yth Annual Regional Meeting of the Chicago Linguistics Society(in year XX) 第 Y 届芝加哥语言学协会地区年会论文集（XX 年）
COGSCI-XX	Proceedings of the Yth Annual Conference of the Cognitive Science Society(in year XX) 第 Y 届人工智能学会年会论文集（XX 年）
COLING-XX	Proceedings of the Yth International Conference on Computational Linguistics(in year XX) 第 Y 届国际计算语言学会议论文集（XX 年）
EUROSPEECH-XX	Proceedings of the Yth European Conference on Speech Communication and Technology(in year XX) 第 Y 届欧洲语音通信和技术会议论文集（XX 年）
ICSLP-XX	Proceedings of the International Conference on Spoken Language Processing (in year XX) 口语处理国际会议论文集（XX 年）
IEEE ICASSP-XX	Proceedings of the IEEE International Conference on Acoustics, Speech, & Signal Processing(in year XX) IEEE 国际声学、语音和信号处理会议论文集（XX 年）
IJCAI-XX	Proceedings of the Yth International Joint Conference on Artificial Intelligence(in year XX) 第 Y 届国际人工智能联合会议论文集（XX 年）
Dagger(†)	marks references that we did not have access to; the details of these references thus may not be correct. 以该符号标记的参考文献是我们没有查到的参考文献，即这些文献的细节有可能不准确

Abney, S. P. (1991). Parsing by chunks. In Berwick, R. C., Abney, S. P., and Tenny, C. (Eds.), *Principle-Based Parsing: Computation and Psycholinguistics*, pp. 257–278. Kluwer, Dordrecht.

Abney, S. P. (1997). Stochastic attribute-value grammars. *Computational Linguistics*, *23*(4), 597–618.

Abney, S. P., Schapire, R. E., and Singer, Y. (1999). Boosting applied to tagging and PP attachment. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, College Park, MD, pp. 38–45. ACL.

Ades, A. E. and Steedman, M. J. (1982). On the order of words. *Linguistics and Philosophy*, *4*, 517–558.

Adjukiewicz, K. (1935). Die syntaktische Konnexität. *Studia Philosophica*, *1*, 1–27. English translation "Syntactic Connexion" by H. Weber in McCall, S. (Ed.) *Polish Logic*, pp. 207–231, Oxford University Press, Oxford, 1967.

Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, *6*, 37–66.

Aho, A. V., Sethi, R., and Ullman, J. D. (1986). *Compilers: Principles, Techniques, and Tools*. Addison-Wesley, Reading, MA.

Aho, A. V. and Ullman, J. D. (1972). *The Theory of Parsing, Translation, and Compiling*, Vol. 1. Prentice-Hall, Englewood Cliffs, NJ.

Aït-Kaci, H. (1984). *A Lattice-Theoretic Approach to Computation Based on a Calculus of Partially Ordered Types*. Ph.D. thesis, University of Pennsylvania.

Algoet, P. H. and Cover, T. M. (1988). A sandwich proof of the Shannon-McMillan-Breiman theorem. *The Annals of Probability*, *16*(2), 899–909.

Allen, J. (1984). Towards a general theory of action and time. *Artificial Intelligence*, *23*(2), 123–154.

Allen, J. (1995). *Natural Language Understanding*. Benjamin Cummings, Menlo Park, CA.

Allen, J. and Core, M. (1997). Draft of DAMSL: Dialog act markup in several layers. Unpublished manuscript.

Allen, J., Ferguson, G., Miller, B., and Ringger, E. (1995). Spoken dialogue and interactive planning. In *Proceedings ARPA Speech and Natural Language Workshop*, Austin, TX, pp. 202–207. Morgan Kaufmann.

Allen, J. and Perrault, C. R. (1980). Analyzing intention in utterances. *Artificial Intelligence*, *15*, 143–178.

Allen, J., Hunnicut, M. S., and Klatt, D. H. (1987). *From Text to Speech: The MITalk system*. Cambridge University Press, Cambridge.

Allwood, J., Nivre, J., and Ahlsén, E. (1992). On the semantics and pragmatics of linguistic feedback. *Journal of Semantics*, *9*, 1–26.

Alshawi, H. (1987). *Memory and Context for Language Interpretation*. Cambridge University Press, Cambridge.

Alshawi, H. (Ed.). (1992). *The Core Language Engine*. MIT Press, Cambridge, MA.

Alshawi, H., Bangalore, S., and Douglas, S. (1998). Automatic acquisition of hierarchical transduction models for machine translation. In *COLING/ACL-98*, Montreal, pp. 41–47. ACL.

Amsler, R. A. (1980). *The Structure of the Merriam-Webster Pocket Dictionary*. Ph.D. thesis, University of Texas, Austin, Texas. Report No.

Amsler, R. A. (1981). A taxonomy of English nouns and verbs. In *ACL-81*, Stanford, CA, pp. 133–138. ACL.

Anderson, J. A. (1995). *An Introduction to Neural Networks*. MIT Press, Cambridge, MA.

Anderson, S. R. (1985). *Phonology in the Twentieth Century*. Cambridge University Press, Cambridge.

Antworth, E. L. (1990). *PC-KIMMO: A Two-level Processor for Morphological Analysis*. Summer Institute of Linguistics, Dallas, TX.

Aone, C. and Bennett, S. W. (1995). Evaluating automated and manual acquisition of anaphora resolution strategies. In *ACL-95*, Cambridge, MA, pp. 122–129. ACL.

Appelt, D. E. (1985). *Natural Language Generation*. Cambridge University Press, Cambridge.

Appelt, D. E., Hobbs, J. R., Bear, J., Israel, D., Kameyama, M., Kehler, A., Martin, D., Myers, K., and Tyson, M. (1995). SRI International FASTUS system MUC-6 test results and analysis. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Francisco, pp. 237–248. Morgan Kaufmann.

Appelt, D. E. and Israel, D. (1997). ANLP-97 tutorial: Building information extraction systems. Available as `www.ai.sri.com/~appelt/ie-tutorial/`.

Archangeli, D. (1984). *Underspecification in Yawelmani Phonology and Morphology*. Ph.D. thesis, MIT, Cambridge, MA.

Archangeli, D. (1997). Optimality theory: An introduction to linguistics in the 1990s. In Archangeli, D. and Langendoen, D. T. (Eds.), *Optimality Theory: An Overview*. Basil Blackwell, Oxford.

Archangeli, D. and Langendoen, D. T. (Eds.). (1997). *Optimality Theory: An Overview*. Basil Blackwell, Oxford.

Arens, Y., Granacki, J., and Parker, A. (1987). Phrasal analysis of long noun sequences. In *ACL-87*, Stanford, CA, pp. 59–64. ACL.

Argamon, S., Dagan, I., and Krymolowski, Y. (1998). A memory-based approach to learning shallow natural language patterns. In *COLING/ACL-98*, Montreal, pp. 67–73. ACL.

Ariel, M. (1990). *Accessing Noun Phrase Antecedents*. Routledge.

Asher, N. (1993). *Reference to Abstract Objects in Discourse*. SLAP 50, Dordrecht, Kluwer.

Atal, B. S. and Hanauer, S. (1971). Speech analysis and synthesis by prediction of the speech wave. *Journal of the Acoustical Society of America*, *50*, 637–655.

Atkinson, M. and Drew, P. (1979). *Order in Court*. Macmillan, London.

Attar, R. and Fraenkel, A. S. (1977). Local feedback in full-text retrieval systems. *Journal of the ACM*, *24*(3), 398–417.

Austin, J. L. (1962). *How to Do Things with Words*. Harvard University Press, Cambridge, MA.

Austin, S., Schwartz, R., and Placeway, P. (1991). The forward-backward search algorithm. In *IEEE ICASSP-91*, Vol. 1, pp. 697–700. IEEE.

Baayen, H. and Sproat, R. (1996). Estimating lexical priors for low-frequency morphologically ambiguous forms. *Computational Linguistics*, *22*(2), 155–166.

Babyonyshev, M. and Gibson, E. (1999). The complexity of nested structures in Japanese. *Language*, *75*(3), 423–450.

Bach, E. (1976). An extension of classical transformational grammar. In *Problems of Linguistic Metatheory (Proceedings of the 1976 Conference)*. Michigan State University.

Bach, E. (1988). Categorial grammars as theories of language. In Oehrle, R. T., Bach, E., and Wheeler, D. (Eds.), *Categorial Grammars and Natural Language Structures*, pp. 17–34. D. Reidel, Dordrecht.

Backus, J. W. (1959). The syntax and semantics of the proposed international algebraic language of the Zurch ACM-GAMM Conference. In *Information Processing: Proceedings of the International Conference on Information Processing, Paris*, pp. 125–132. UNESCO.

Backus, J. W. (1996). Transcript of question and answer session. In Wexelblat, R. L. (Ed.), *History of Programming Languages*, p. 162. Academic Press, New York.

Bacon, F. (1620). *Novum Organum*. Annotated edition edited by Thomas Fowler published by Clarendon Press, Oxford, 1889.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. ACM Press, New York.

Bagley, W. C. (1900–1901). The apperception of the spoken sentence: A study in the psychology of language. *The American Journal of Psychology*, *12*, 80–130. †.

Bahl, L. R., de Souza, P. V., Gopalakrishnan, P. S., Nahamoo, D., and Picheny, M. A. (1992). A fast match for continuous speech recognition using allophonic models. In *IEEE ICASSP-92*, San Francisco, CA, pp. I.17–20. IEEE.

Bahl, L. R. and Mercer, R. L. (1976). Part of speech assignment by a statistical decision algorithm. In *Proceedings IEEE International Symposium on Information Theory*, pp. 88–89.

Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *5*(2), 179–190.

Bailey, D., Feldman, J., Narayanan, S., and Lakoff, G. (1997). Modeling embodied lexical development. In *COGSCI-97*, Stanford, CA, pp. 19–24. Lawrence Erlbaum.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet project. In *COLING/ACL-98*, pp. 86–90.

Baker, J. K. (1979). Trainable grammars for speech recognition. In Klatt, D. H. and Wolf, J. J. (Eds.), *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pp. 547–550.

Baker, J. K. (1975). The DRAGON system – An overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *ASSP-23*(1), 24–29.

Bangalore, S. and Joshi, A. K. (1999). Supertagging: An approach to almost parsing. *Computational Linguistics*, *25*(2), 237–265.

Bar-Hillel, Y. (1953). A quasi-arithmetical notation for syntactic description. *Language*, *29*, 47–58. Reprinted in Y. Bar-Hillel. (1964). *Language and Information: Selected Essays on their Theory and Application*, Addison-Wesley 1964, 61–74.

Bar-Hillel, Y. (1960). The present status of automatic translation of languages. In Alt, F. (Ed.), *Advances in Computers 1*, pp. 91–163. Academic Press.

Bar-Hillel, Y., Perles, M., and Shamir, E. (1961). On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*, *14*, 143–172. Reprinted in Y. Bar-Hillel. (1964). *Language and Information: Selected Essays*

on their Theory and Application, Addison-Wesley 1964, 116–150.

Barnstone, W. (1993). *The Poetics of Translation*. Yale University Press.

Barton, Jr., G. E., Berwick, R. C., and Ristad, E. S. (1987). *Computational Complexity and Natural Language*. MIT Press, Cambridge, MA.

Bateman, J. A. (1997). Enabling technology for multilingual natural language generation: the KPML development environment. *Natural Language Engineering*, *3*(1), 15–55.

Bateman, J. A. and Hovy, E. H. (1992). An overview of computational text generation. In Butler, C. S. (Ed.), *Computers and Texts: An Applied Perspective*, pp. 53–74. Basil Blackwell, Oxford.

Bateman, J. A., Kasper, R. T., Moore, J. D., and Whitney, R. (1990). A general organization of knowledge for natural language processing: The Penman Upper Model. Tech. rep., USC/ISI.

Bates, R. (1997). The corrections officer: Can John Kidd save Ulysses. *Lingua Franca*. October.

Bauer, L. (1983). *English word-formation*. Cambridge University Press, Cambridge.

Baum, L. E. (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. In Shisha, O. (Ed.), *Inequalities III: Proceedings of the Third Symposium on Inequalities*, University of California, Los Angeles, pp. 1–8. Academic Press.

Baum, L. E. and Eagon, J. A. (1967). An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, *73*(3), 360–363.

Baum, L. E. and Petrie, T. (1966). Statistical inference for probabilistic functions of finite-state Markov chains. *Annals of Mathematical Statistics*, *37*(6), 1554–1563.

Bayes, T. (1763). *An Essay Toward Solving a Problem in the Doctrine of Chances*, Vol. 53. Reprinted in *Facsimiles of two papers by Bayes*, Hafner Publishing Company, New York, 1963.

Bazell, C. E. (1952). The correspondence fallacy in structural linguistics. In *Studies by Members of the English Department, Istanbul University (3)*. Reprinted in Eric P. Hamp, Fred W. Householder, and Robert Austerlitz, (Eds.), *Readings in Linguistics II* (1966), pp. 271–298. University of Chicago Press, Chicago, IL.

Becker (1975). The phrasal lexicon. In Schank, R. and Nash-Webber, B. (Eds.), *Theoretical Issues in Natural Language Processing*. Cambridge, MA.

Beckman, M. E. and Pierrehumbert, J. (1986). Intonational structure in English and Japanese. *Phonology Yearbook*, *3*, 255–310.

Beesley, K. R. (1996). Arabic finite-state morphological analysis and generation. In *COLING-96*, Copenhagen, pp. 89–94.

Bellegarda, J. R. (1999). Speech recognition experiments using multi-span statistical language models. In *IEEE ICASSP-99*, pp. 717–720. IEEE.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press, Princeton, NJ.

Berger, A. and Printz, H. (1998). Recognition performance of a large-scale dependency grammar language model. In *ICSLP-98*, Sydney, Vol. 6, pp. 2487–2490.

Berry, M. W., Drmac, Z., and Jessup, E. R. (1999). Matrices, vector spaces, and information retrieval. *SIAM Review*, *41*(2), 335–362.

Bever, T. G. (1970). The cognitive basis for linguistic structures. In Hayes, J. R. (Ed.), *Cognition and the Development of Language*, pp. 279–352. Wiley, New York.

Bird, S. and Ellison, T. M. (1994). One-level phonology: Autosegmental representations and rules as finite automata. *Computational Linguistics*, *20*(1).

Birnbaum, L. and Selfridge, M. (1981). Conceptual analysis of natural language. In Schank, R. C. and Riesbeck, C. K. (Eds.), *Inside Computer Understanding: Five Programs plus Miniatures*, pp. 318–353. Lawrence Erlbaum, Hillsdale.

Black, A. W., Taylor, P., and Caley, R. (1996-1999). The Festival Speech Synthesis System system. Manual and source code available at www.cstr.ed.ac.uk/projects/festival.html.

Black, E., Abney, S. P., Flickinger, D., Gdaniec, C., Grishman, R., Harrison, P., Hindle, D., Ingria, R., Jelinek, F., Klavans, J., Liberman, M., Marcus, M. P., Roukos, S., Santorini, B., and Strzalkowski, T. (1991). A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings DARPA Speech and Natural Language Workshop*, Pacific Grove, CA, pp. 306–311. Morgan Kaufmann.

Black, E. (1988). An experiment in computational discrimination of English word senses. *IBM Journal of Research and Development*, *32*(2), 185–194.

Black, E., Jelinek, F., Lafferty, J. D., Magerman, D. M., Mercer, R. L., and Roukos, S. (1992). Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings DARPA Speech and Natural Language Workshop*, Harriman, New York, pp. 134–139. Morgan Kaufmann.

Blair, C. R. (1960). A program for correcting spelling errors. *Information and Control*, *3*, 60–67.

Bledsoe, W. W. and Browning, I. (1959). Pattern recognition and reading by machine. In *1959 Proceedings of the Eastern Joint Computer Conference*, pp. 225–232. Academic, New York.

Bloomfield, L. (1914). *An Introduction to the Study of Language*. Henry Holt and Company, New York.

Bloomfield, L. (1933). *Language*. University of Chicago Press, Chicago.

Bobrow, D. G., Kaplan, R. M., Kay, M., Norman, D. A., Thompson, H., and Winograd, T. (1977). Gus, a frame driven dialog system. *Artificial Intelligence*, *8*, 155–173.

Bobrow, D. G. and Winograd, T. (1977). An overview of KRL, a knowledge representation language. *Cognitive Science*, *1*(1), 3–46.

Bobrow, R. J. and Webber, B. (1980). Knowledge representation for syntactic/semantic processing. In *AAAI-80*, Stanford, CA, pp. 316–323. Morgan Kaufmann.

Bock, K. and Loebell, H. (1990). Framing sentences. *Cognition*, *35*, 1–39.

Bod, R. (1993). Using an annotated corpus as a stochastic grammar. In *Proceedings of the Sixth Conference of the European Chapter of the ACL*, pp. 37–44. ACL.

Boguraev, B. and Briscoe, T. (Eds.). (1989). *Computational Lexicography for Natural Language Processing*. Longman, London.

Bolinger, D. (1981). Two kinds of vowels, two kinds of rhythm. Indiana University Linguistics Club.

Booth, T. L. (1969). Probabilistic representation of formal languages. In *IEEE Conference Record of the 1969 Tenth Annual Symposium on Switching and Automata Theory*, pp. 74–81.

Booth, T. L. and Thompson, R. A. (1973). Applying probability measures to abstract languages. *IEEE Transactions on Computers*, *C-22*(5), 442–450.

Bourlard, H. and Morgan, N. (1994). *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Press.

Brachman, R. J. (1979). On the epistemogical status of semantic networks. In Findler, N. V. (Ed.), *Associative Networks: Representation and Use of Knowledge by Computers*, pp. 3–50. Academic Press, New York.

Brachman, R. J. and Levesque, H. J. (Eds.). (1985). *Readings in Knowledge Representation*. Morgan Kaufmann, San Mateo, CA.

Brachman, R. J. and Schmolze, J. G. (1985). An overview of the KL-ONE knowledge representation system. *Cognitive Science*, *9*(2), 171–216.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth & Brooks, Pacific Grove, California.

Brennan, S. E. (1995). Centering attention in discourse. *Language and Cognitive Processes*, *10*, 137–167.

Brennan, S. E., Friedman, M. W., and Pollard, C. (1987). A centering approach to pronouns. In *ACL-87*, Stanford, CA, pp. 155–162. ACL.

Brent, M. R. (1993). From grammar to lexicon: Unsupervised learning of lexical syntax. *Computational Linguistics*, *19*(2), 243–262.

Brent, M. R. and Cartwright, T. A. (1996). Distributional regularity and phonotactic constraints are useful for segmentation. *Cognition*, *61*, 93–125.

Bresnan, J. (Ed.). (1982). *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA.

Bresnan, J. and Kaplan, R. M. (1982). Introduction: Grammars as mental representations of language. In Bresnan, J. (Ed.), *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, MA.

Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, *21*(4), 543–566.

Brill, E. (1997). Unsupervised learning of disambiguation rules for part of speech tagging. Unpublished manuscript.

Brill, E. and Resnik, P. (1994). A rule-based approach to prepositional phrase attachment disambiguation. In *COLING-94*, Kyoto, pp. 1198–1204.

Briscoe, T. and Carroll, J. (1993). Generalized Probabilistic LR parsing of natural language (corpora) with unification-based grammars. *Computational Linguistics*, *19*(1), 25–59.

Briscoe, T. and Carroll, J. (1997). Automatic extraction of subcategorization from corpora. In *Fifth Conference on Applied Natural Language Processing*, Washington, D.C., pp. 356–363. ACL.

Bromberger, S. and Halle, M. (1989). Why phonology is different. *Linguistic Inquiry*, *20*, 51–70.

Brown, J. S. and Burton, R. R. (1975). Multiple representations of knowledge for tutorial reasoning. In Bobrow, D. G. and Collins, A. (Eds.), *Representation and Understanding*, pp. 311–350. Academic Press, New York.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational Linguistics*, *16*(2), 79–85.

Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., Lai, J. C., and Mercer, R. L. (1992). An estimate of an upper bound for the entropy of English. *Computational Linguistics*, *18*(1), 31–40.

Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, *19*(2), 263–311.

Brown, P. F., Della Pietra, V. J., deSouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based *n*-gram models of natural language. *Computational Linguistics*, *18*(4), 467–479.

Brown, R. D. (1996). Example-based machine translation in the Pangloss system. In *COLING-96*, Copenhagen, pp. 169–174.

Brown, R. D. and Frederking, R. (1995). Applying statistical English language modeling to symbolic machine translation. In *6th International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium, pp. 221–239.

Bub, T., Wahlster, W., and Waibel, A. (1997). Verbmobil: The combination of deep and shallow processing for spontaneous speech translation. In *IEEE ICASSP-97*, pp. 71–74. IEEE.

Bunt, H. (1994). Context and dialogue control. *Think*, *3*, 19–31.

Bybee, J. L. (1985). *Morphology: A study of the relation between meaning and form*. John Benjamins, Amsterdam.

Bybee, J. L. (1995). Regular morphology and the lexicon. *Language and Cognitive Processes*, *10*(5), 425–455.

Bybee, J. L. (1996). The phonology of the lexicon: evidence from lexical diffusion. In Barlow, M. and Kemmer, S. (Eds.), *Usage-based Models of Language*.

Bybee, J. L. and Slobin, D. I. (1982). Rules and schemas in the development and use of English past tense. *Language*, *58*, 265–289.

Calder, J. (1987). Typed unification for natural language processing. In Kahn, G., MacQueen, D., and Plotkin, G. (Eds.), *Categories, Polymorphism, and Unification*. Centre for Cognitive Science, University of Edinburgh, Edinburgh, Scotland†.

Callan, J. P., Croft, W. B., and Harding, S. M. (1992). The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert System Applications*, Valencia, Spain, pp. 78–83. Springer-Verlag.

Cao, X. (1792). *The Story of the Stone*. (Also known as *The Dream of the Red Chamber*). Penguin Classics, London. First published in Chinese in 1792, translated into English by David Hawkes and published by Penguin in 1973.

Caramazza, A., Grober, E., Garvey, C., and Yates, J. (1977). Comprehension of anaphoric pronouns. *Journal of Verbal Learning and Verbal Behaviour*, *16*, 601–609.

Carberry, S. (1990). *Plan Recognition in Natural Language Dialog*. MIT Press, Cambridge, MA.

Carbonell, J. (1982). Metaphor: An inescapable phenomenon in natural language comprehension. In Lehnert, W. and Ringle, M. (Eds.), *Strategies for Natural Language Processing*, pp. 415–434. Lawrence Erlbaum.

Carbonell, J., Cullingford, R. E., and Gershman, A. V. (1981). Steps toward knowledge-based machine translation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *3*(4), 376–392.

Carbonell, J., Mitamura, T., and Nyberg, E. H. (1992). The KANT perspective: A critique of pure transfer (and pure interlingua, pure statistics, ...). In *International Conference on Theoretical and Methodological Issues in Machine Translation*.

Cardie, C. (1993). A case-based approach to knowledge acquisition for domain specific sentence analysis. In *AAAI-93*, pp. 798–803. AAAI Press.

Cardie, C. (1994). *Domain-Specific Knowledge Acquisition for Conceptual Sentence Analysis*. Ph.D. thesis, University of Massachusetts, Amherst, MA. Available as CMPSCI Technical Report 94-74.

Carletta, J., Dahlbäck, N., Reithinger, N., and Walker, M. A. (1997). Standards for dialogue coding in natural language processing. Tech. rep. Report no. 167, Dagstuhl Seminars. Report from Dagstuhl seminar number 9706.

Carletta, J. (1996). Assessing agreement on classification tasks: The Kappa statistic. *Computational Linguistics*, *22*(2), 249–254.

Carletta, J., Isard, A., Isard, S., Kowtko, J. C., Doherty-Sneddon, G., and Anderson, A. H. (1997). The reliability of a dialogue structure coding scheme. *Computational Linguistics*, *23*(1), 13–32.

Carpenter, B. (1991). The generative power of categorial grammars and head-driven phrase structure grammars with lexical rules. *Computational Linguistics*, *17*(3), 301–313.

Carpenter, B. (1992). *The Logic of Typed Feature Structures*. Cambridge University Press, Cambridge.

Carpenter, B. and Penn, G. (1994). The Attribute Logic Engine Users's Guide Version 2.0.1. Tech. rep., Carnegie Mellon University.

Carroll, J., Briscoe, T., and Sanfilippo, A. (1998). Parser evaluation: a survey and a new proposal. In *Proceedings, First International Conference on Language Resources and Evaluation*, Granada, Spain, pp. 447–454. European Language Resources Association.

Casey, R. G. and Lecolinet, E. (1996). A survey of methods and strategies in character segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(7), 690–706.

Celex (1993). The CELEX lexical database. Centre for Lexical Information, Max Planck Institute for Psycholinguistics.

Chafe, W. L. (1976). Givenness, contrastiveness, definiteness, subjects, topics, and point of view. In Li, C. N. (Ed.), *Subject and Topic*, pp. 25–55. Academic Press, New York.

Chandioux, J. (1976). MÉTÉO: un système opérationnel pour la traduction automatique des bulletins météorologiques destinés au grand public. *Meta*, *21*, 127–133.

Chandler, S. (1991). Metaphor comprehension: A connectionist approach to implications for the mental lexicon. *Metaphor and Symbolic Activity*, 6(4), 227–258.

Chang, N., Gildea, D., and Narayanan, S. (1998). A dynamic model of aspectual composition. In *COGSCI-98*, Madison, WI, pp. 226–231. Lawrence Erlbaum.

Charniak, E. (1993). *Statistical Language Learning*. MIT Press.

Charniak, E. and Shimony, S. E. (1990). Probabilistic semantics for cost based abduction. In Dietterich, T. S. W. (Ed.), *AAAI-90*, Boston, MA, pp. 106–111. MIT Press.

Charniak, E. (1997). Statistical parsing with a context-free grammar and word statistics. In *AAAI-97*, Menlo Park. AAAI Press.

Charniak, E. and Goldman, R. (1988). A logic for semantic interpretation. In *ACL-88*, Buffalo, NY. ACL.

Charniak, E., Hendrickson, C., Jacobson, N., and Perkowitz, M. (1993). Equations for part-of-speech tagging. In *AAAI-93*, Washington, D.C., pp. 784–789. AAAI Press.

Charniak, E. and McDermott, D. (1985). *Introduction to Artificial Intelligence*. Addison Wesley.

Chelba, C., Engle, D., Jelinek, F., Jimenez, V., Khudanpur, S., Mangu, L., Printz, H., Ristad, E. S., Rosenfeld, R., Stolcke, A., and Wu, D. (1997). Structure and performance of a dependency language model. In *EUROSPEECH-97*, Vol. 5, pp. 2775–2778.

Chelba, C. and Jelinek, F. (1998). Exploiting syntactic structure for language modeling. In *COLING/ACL-98*, Montreal, pp. 225–231. ACL.

Chen, J. N. and Chang, J. S. (1998). Topical clustering of MRD senses based on information retrieval techniques. *Computational Linguistics*, *24*(1), 61–96.

Chen, S. S., Eide, E. M., Gales, M. J. F., Gopinath, R. A., Kanevsky, D., and Olsen, P. (1999). Recent improvements to IBM's speech recognition system for automatic transcription of Broadcast News. In *IEEE ICASSP-99*, pp. 37–40. IEEE.

Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *ACL-96*, Santa Cruz, CA, pp. 310–318. ACL.

Chen, S. F., Seymore, K., and Rosenfeld, R. (1998). Topic adaptation for language modeling using unnormalized exponential models. In *IEEE ICASSP-98*, pp. 681–684. IEEE.

Chierchia, G. and McConnell-Ginet, S. (1991). *Meaning and Grammar*. MIT Press, Cambridge, MA.

Chinchor, N., Hirschman, L., and Lewis, D. L. (1993). Evaluating Message Understanding systems: An analysis of the third Message Understanding Conference. *Computational Linguistics*, *19*(3), 409–449.

Chomsky, N. (1956). Three models for the description of language. *IRI Transactions on Information Theory*, *2*(3), 113–124.

Chomsky, N. (1956/1975). *The Logical Structure of Linguistic Theory*. Plenum.

Chomsky, N. (1957). *Syntactic Structures*. Mouton, The Hague.

Chomsky, N. (1959a). On certain formal properties of grammars. *Information and Control*, *2*, 137–167.

Chomsky, N. (1959b). A review of B. F. Skinner's "Verbal Behavior". *Language*, *35*, 26–58.

Chomsky, N. (1963). Formal properties of grammars. In Luce, R. D., Bush, R., and Galanter, E. (Eds.), *Handbook of Mathematical Psychology*, Vol. 2, pp. 323–418. Wiley, New York.

Chomsky, N. (1965). *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Chomsky, N. (1969). Quine's empirical assumptions. In Davidson, D. and Hintikka,

J. (Eds.), *Words and objections. Essays on the work of W. V. Quine*, pp. 53–68. D. Reidel, Dordrecht.

Chomsky, N. (1981). *Lectures on Government and Binding*. Foris, Dordrecht.

Chomsky, N. and Halle, M. (1968). *The Sound Pattern of English*. Harper and Row, New York.

Chomsky, N. and Miller, G. A. (1958). Finite-state languages. *Information and Control*, *1*, 91–112.

Chomsky, N. and Miller, G. A. (1963). Introduction to the formal analysis of natural languages. In Luce, R. D., Bush, R., and Galanter, E. (Eds.), *Handbook of Mathematical Psychology*, Vol. 2, pp. 269–322. Wiley, New York.

Chu-Carroll, J. (1998). A statistical model for discourse act recognition in dialogue interactions. In Chu-Carroll, J. and Green, N. (Eds.), *Applying Machine Learning to Discourse Processing. Papers from the 1998 AAAI Spring Symposium.* Tech. rep. SS-98-01, pp. 12–17. AAAI Press, Menlo Park, CA.

Chu-Carroll, J. and Carberry, S. (1998). Collaborative response generation in planning dialogues. *Computational Linguistics*, *24*(3), 355–400.

Church, A. (1940). A formulation of a simple theory of types. *Journal of Symbolic Logic*, *5*, 56–68.

Church, K. W. (1988). A stochastic parts program and noun phrase parser for unrestricted text. In *Second Conference on Applied Natural Language Processing*, pp. 136–143. ACL.

Church, K. W. and Gale, W. A. (1991). A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, *5*, 19–54.

Church, K. W., Gale, W. A., and Kruskal, J. B. (1991). Appendix A: the Good-Turing theorem. In *Computer Speech and Language* (Church and Gale, 1991), pp. 19–54.

Church, K. W. and Patil, R. (1982). Coping with syntactic ambiguity. *American Journal of Computational Linguistics*, *8*(3-4), 139–149.

Church, K. W. (1980). On memory limitations in natural language processing. Master's thesis, MIT. Distributed by the Indiana University Linguistics Club.

Clark, H. H. and Sengal, C. J. (1979). In search of referents for nouns and pronouns. *Memory and Cognition*, *7*, 35–41.

Clark, H. H. (1994). Managing problems in speaking. *Speech Communication*, *15*, 243–250.

Clark, H. H. and Clark, E. V. (1977). *Psychology and Language*. Harcourt Brace Jovanovich, San Diego.

Clark, H. H. and Schaefer, E. F. (1989). Contributing to discourse. *Cognitive Science*, *13*, 259–294.

Clark, H. H. and Wilkes-Gibbs, D. (1986). Referring as a collaborative process. *Cognition*, *22*, 1–39.

Coccaro, N. and Jurafsky, D. (1998). Towards better integration of semantic predictors in statistical language modeling. In *ICSLP-98*, Sydney, Vol. 6, pp. 2403–2406.

Coch, J. (1996a). Evaluating and comparing three text-production techniques. In *COLING-96*, Copenhagen, pp. 249–254.

Coch, J. (1996b). Overview of AlethGen. In *Demonstration overview for the Proceedings of the Eighth International Workshop on Natural Language Generation,* Herstmonceux, England, pp. 25–28.

Coch, J. (1998). Interactive generation and knowledge administration in MultiMétéo. In *Proceedings of the Ninth International Workshop on Natural Language Generation,* Ontario, Canada, pp. 300–303. System Demonstration.

Cohen, M. H. (1989). *Phonological Structures for Speech Recognition*. Ph.D. thesis, University of California, Berkeley.

Cohen, P. R., Johnston, M., McGee, D., Oviatt, S. L., Clow, J., and Smith, I. (1998). The efficiency of multimodal interaction: a case study. In *ICSLP-98*, Sydney, Vol. 2, pp. 249–252.

Cohen, P. R. (1995). *Empirical Methods for Artificial Intelligence*. MIT Press, Cambridge, MA.

Cohen, P. R. and Perrault, C. R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science*, *3*(3), 177–212.

Colby, K. M., Weber, S., and Hilf, F. D. (1971). Artificial paranoia. *Artificial Intelligence*, *2*(1), 1–25.

Cole, J. S. and Kisseberth, C. W. (1995). Restricting multi-level constraint evaluation. Rutgers Optimality Archive ROA-98.

Cole, R. A., Novick, D. G., Vermeulen, P. J. E., Sutton, S., Fanty, M., Wessels, L. F. A., de Villiers, J. H., Schalkwyk, J., Hansen, B., and Burnett, D. (1997). Experiments with a spoken dialogue system for taking the US census. *Speech Communication*, *23*, 243–260.

Cole, R. A. (1973). Listening for mispronunciations: A measure of what we hear during speech. *Perception and Psychophysics*, *13*, 153–156.

Cole, R. A. (Ed.). (1997). *Survey of the State of the Art in Human Language Technology*. Cambridge University Press, Cambridge.

Cole, R. A. and Jakimik, J. (1980). A model of speech perception. In Cole, R. A. (Ed.), *Perception and Production of Fluent Speech*, pp. 133–163. Lawrence Erlbaum, Hillsdale, NJ.

Cole, R. A., Novick, D. G., Burnett, D., Hansen, B., Sutton, S., and Fanty, M. (1994). Towards automatic collection of the U.S. census. In *IEEE ICASSP-94*, Adelaide, Australia, Vol. I, pp. 93–96. IEEE.

Cole, R. A., Novick, D. G., Fanty, M., Sutton, S., Hansen, B., and Burnett, D. (1993). Rapid prototyping of spoken language systems: The Year 2000 Census Project. In *Proceedings of the International Symposium on Spoken Dialogue*, Waseda University, Tokyo, Japan.

Cole, R. A. and Rudnicky, A. I. (1983). What's new in speech perception? The research and ideas of William Chandler Bagley. *Psychological Review*, *90*(1), 94–101.

Collins, M. J. (1996). A new statistical parser based on bigram lexical dependencies. In *ACL-96*, Santa Cruz, CA, pp. 184–191. ACL.

Collins, M. J. (1999). *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania, Philadelphia.

Collins, M. J., Hajič, J., Ramshaw, L. A., and Tillmann, C. (1999). A statistical parser for Czech. In *ACL-99*, College Park, MD, pp. 505–512. ACL.

Colmerauer, A. (1970). Les systèmes-q ou un formalisme pour analyser et synthétiser des phrase sur ordinateur. Internal publication 43, Département d'informatique de l'Université de Montréal†.

Colmerauer, A. (1975). Les grammaires de métamorphose GIA. Internal publication, Groupe Intelligence artificielle, Faculté des Sciences de Luminy, Université Aix-Marseille II, France, Nov 1975. English version, Metamorphosis grammars. In L. Bolc, (Ed.), *Natural Language Communication with Computers, Lecture Notes in Computer Science 63*, Springer Verlag, Berlin, 1978, pp. 133–189.

Colmerauer, A. and Roussel, P. (1996). The birth of Prolog. In Bergin Jr., T. J. and Gibson, Jr., R. G. (Eds.), *History of Programming Languages – II*, pp. 331–352. ACM Press/Addison-Wesley, New York.

Comrie, B. (1989). *Language Universals and Linguistic Typology*. Basil Blackwell, Oxford. Second edition.

Connine, C. M. (1990). Effects of sentence context and lexical knowledge in speech processing. In Altmann, G. T. M. (Ed.), *Cognitive Models of Speech Processing*, pp. 281–294. MIT Press, Cambridge, MA.

Connine, C. M. and Clifton, Jr., C. (1987). Interactive use of lexical information in speech perception. *Journal of Experimental Psychology: Human Perception and Performance*, *13*, 291–299.

Connolly, D., Burger, J. D., and Day, D. S. (1994). A machine learning approach to anaphoric reference. In *Proceedings of the International Conference on New Methods in Language Processing (NeMLaP)*. ACL.

Cooley, J. W. and Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, *19*(90), 297–301.

Cooper, A. (1995). *About Face: The essentials of user interface design*. IDG Books.

Copeland, J. (1993). *Artificial Intelligence: A Philosophical Introduction*. Blackwell, Oxford.

Copestake, A. and Briscoe, T. (1995). Semi-productive polysemy and sense extension. *Journal of Semantics*, *12*(1), 15–68.

Core, M., Ishizaki, M., Moore, J. D., Nakatani, C., Reithinger, N., Traum, D., and Tutiya, S. (1999). The report of the third workshop of the Discourse Resource Initiative, Chiba University and Kazusa Academia Hall. Tech. rep. No.3 CC-TR-99-1, Chiba Corpus Project, Chiba, Japan.

Cottrell, G. W. (1985). *A Connectionist Approach to Word Sense Disambiguation*. Ph.D. thesis, University of Rochester, Rochester, NY. Revised version published in the same title by Pitman in 1989.

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27.

Cover, T. M. and King, R. C. (1978). A convergent gambling estimate of the entropy of English. *IEEE Transactions on Information Theory*, *24*(4), 413–421.

Cover, T. M. and Thomas, J. A. (1991). *Elements of information theory*. Wiley, New York.

Cowie, J., Guthrie, J. A., and Guthrie, L. M. (1992). Lexical disambiguation using simulated annealing. In *COLING-92*, Nantes, France, pp. 359–365.

Cowper, E. A. (1976). *Constraints on Sentence Complexity: A Model for Syntactic Processing*. Ph.D. thesis, Brown University, Providence, RI†.

Crawley, R. A., Stevenson, R. J., and Kleinman, D. (1990). The use of heuristic strategies in the interpretation of pronouns. *Journal of Psycholinguistic Research*, *19*, 245–264.

Crestani, F., Lemas, M., Rijsbergen, C. J. V., and Campbell, I. (1998). "Is This Document Relevant? ...Probably": a survey of probabilistic models in information retrieval. *ACM Computing Surveys*, *30*(4), 528–552.

Croft, W. (1990). *Typology and Universals*. Cambridge University Press, Cambridge.

Croft, W. (1995). Intonation units and grammatical structure. *Linguistics*, *33*, 839–882.

Crouch, C. J. and Yang, B. (1992). Experiments in automatic statistical thesaurus construction. In *SIGIR-92*, Copenhagen, Denmark, pp. 77–88. ACM.

Cruse, D. A. (1986). *Lexical Semantics*. Cambridge University Press,, Cambridge.

Crystal, D. (1969). *Prosodic systems and intonation in English*. Cambridge University Press, Cambridge.

Cullingford, R. E. (1981). SAM. In Schank, R. C. and Riesbeck, C. K. (Eds.), *Inside Computer Understanding: Five Programs plus Miniatures*, pp. 75–119. Lawrence Erlbaum, Hillsdale, NJ.

Culy, C. (1985). The complexity of the vocabulary of Bambara. *Linguistics and Philosophy*, *8*, 345–351.

Cutler, A. (1986). Forbear is a homophone: Lexical prosody does not constrain lexical access. *Language and Speech*, *29*, 201–219.

Cutler, A. and Carter, D. M. (1987). The predominance of strong initial syllables in the English vocabulary. *Computer Speech and Language*, *2*, 133–142.

Cutler, A. and Norris, D. (1988). The role of strong syllables in segmentation for lexical access. *Journal of Experimental Psychology: Human Perception and Performance*, *14*, 113–121.

Cutting, D., Kupiec, J., Pedersen, J. O., and Sibun, P. (1992a). A practical part-of-speech tagger. In *Third Conference on Applied Natural Language Processing*, pp. 133–140. ACL.

Cutting, D., Karger, D. R., Pedersen, J. O., and Tukey, J. W. (1992b). Scatter/gather: A cluster-based approach to browsing large document collections. In *SIGIR-92*, Copenhagen, Denmark, pp. 318–329. ACM.

Daelemans, W., De Smedt, K., and Gazdar, G. (1992). Inheritance in natural language processing. *Computational Linguistics*, *18*(2), 205–218.

Daelemans, W., Gillis, S., and Durieux, G. (1994). The acquisition of stress: A data-oriented approach. *Computational Linguistics*, *20*(3), 421–451.

Dagan, I. and Church, K. W. (1997). Termight: Coordinating humans and machines in bilingual terminology acquisition. *Machine Translation*, *12*, 89–107.

Dale, R. (1992). *Generating Referring Expressions: Constructing Descriptions in a Domain of Objects and Processes*. MIT Press, Cambridge, MA.

Dale, R., Eugenio, B. D., and Scott, D. R. (1998a). Introduction to the special issue on natural language generation. *Computational Linguistics*, *24*(3), 345–353.

Dale, R., Oberlander, J., Milosavljevic, M., and Knott, A. (1998b). Integrating natural language generation and hypertext to produce dynamic documents. *Interacting with Computers*, *11*(2), 109–135.

Dalianis, H. (1999). Aggregation in natural language generation. *Computational Intelligence*, *15*(4).

Daly, N. A. and Zue, V. W. (1992). Statistical and linguistic analyses of F_0 in read and spontaneous speech. In *ICSLP-92*, Vol. 1, pp. 763–766.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, *7*(3), 171–176.

Damerau, F. J. and Mays, E. (1989). An examination of undetected typing errors. *Information Processing and Management*, *25*(6), 659–664.

Dang, H. T., Kipper, K., Palmer, M., and Rosenzweig, J. (1998). Investigating regular sense extensions based on intersective levin classes. In *COLING/ACL-98*, Montreal, pp. 293–299. ACL.

Danieli, M. and Gerbino, E. (1995). Metrics for evaluating dialogue strategies in a spoken language system. In *Proceedings of the 1995 AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, Stanford, CA, pp. 34–39. AAAI Press, Menlo Park, CA.

Davey, A. (1979). *Discourse Production: A Computer Model of Some Aspects of a Speaker*. Edinburgh University Press.

David, Jr., E. E. and Selfridge, O. G. (1962). Eyes and ears for computers. *Proceedings of the IRE (Institute of Radio Engineers)*, *50*, 1093–1101.

Davidson, D. (1967). The logical form of action sentences. In Rescher, N. (Ed.), *The Logic of Decision and Action*. University of Pittsburgh Press.

Davis, E. (1990). *Representations of Commonsense Knowledge*. Morgan Kaufmann, San Mateo, CA.

Davis, K. H., Biddulph, R., and Balashek, S. (1952). Automatic recognition of spoken digits. *Journal of the Acoustical Society of America*, *24*(6), 637–642.

de Marcken, C. (1996). *Unsupervised Language Acquisition*. Ph.D. thesis, MIT.

de Tocqueville, A. (1840). *Democracy in America*. Doubleday, New York. The 1966 translation by George Lawrence.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, *41*, 391–407.

DeJong, G. F. (1982). An overview of the FRUMP system. In Lehnert, W. G. and Ringle, M. H. (Eds.), *Strategies for Natural Language Processing*, pp. 149–176. Lawrence Erlbaum, Hillsdale, NJ.

DeJong, G. F. and Waltz, D. L. (1983). Understanding novel language. *Computers and Mathematics with Applications*, *9*.

Dell, G. S. (1986). A spreading activation theory of retrieval in sentence production. *Psychological Review*, *93*, 283–321.

Dell, G. S. (1988). The retrieval of phonological forms in production: Tests of predictions from a connectionist model. *Journal of Memory and Language*, *27*, 124–142.

Dell, G. S., Schwarts, M. F., Martin, N., Saffran, E., and Gagnon, D. A. (1997). Lexical access in aphasic and nonaphasic speakers. *Psychological Review*, *104*(4), 801–838.

Demetriou, G., Atwell, E., and Souter, C. (1997). Large-scale lexical semantics for speech recognition support. In *EUROSPEECH-97*, pp. 2755–2758.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the *EM* algorithm. *Journal of the Royal Statistical Society*, *39*(1), 1–21.

Denes, P. (1959). The design and operation of the mechanical speech recognizer at University College London. *Journal of the British Institution of Radio Engineers*, *19*(4), 219–234. Appears together with companion paper (Fry 1959).

Deng, L., Lennig, M., Seitz, F., and Mermelstein, P. (1990). Large vocabulary word recognition using context-dependent allophonic hidden Markov models. *Computer Speech and Language*, *4*, 345–357.

Dermatas, E. and Kokkinakis, G. (1995). Automatic stochastic tagging of natural language texts. *Computational Linguistics*, *21*(2), 137–164.

DeRose, S. J. (1988). Grammatical category disambiguation by statistical optimization. *Computational Linguistics*, *14*, 31–39.

Di Eugenio, B. (1990). Centering theory and the Italian pronominal system. In *COLING-90*, Helsinki, pp. 270–275.

Di Eugenio, B. (1996). The discourse functions of Italian subjects: A centering approach. In *COLING-96*, Copenhagen, pp. 352–357.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*(7), 1895–1924.

Dolan, W. B. (1994). Word sense ambiguation: Clustering related senses. In *COLING-94*, Kyoto, Japan, pp. 712–716. ACL.

Dorr, B. (1992). The use of lexical semantics in interlingual machine translation. *Journal of Machine Translation*, *7*(3), 135–193.

Dorr, B. (1993). *Machine Translation*. MIT Press, Cambridge, MA.

Dostert, L. (1955). The Georgetown-I.B.M. experiment. In *Machine Translation of Languages: Fourteen Essays*, pp. 124–135. MIT Press.

Downing, P. (1977). On the creation and use of English compound nouns. *Language*, *53*(4), 810–842.

Dowty, D. (1991). Thematic proto-roles and argument selection. *Language*, *67*(3), 547–619.

Dowty, D. R. (1979). *Word Meaning and Montague Grammar*. D. Reidel, Dordrecht.

Dowty, D. R., Wall, R. E., and Peters, S. (1981). *Introduction to Montague Semantics*. D. Reidel, Dordrecht.

Du Bois, J. W., Schuetze-Coburn, S., Cumming, S., and Paolino, D. (1983). Outline of discourse transcription. In Edwards, J. A. and Lampert, M. D. (Eds.), *Talking Data: Transcription and Coding in Discourse Research*, pp. 45–89. Lawrence Erlbaum, Hillsdale, NJ.

Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York.

Earley, J. (1968). *An Efficient Context-Free Parsing Algorithm*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.

Earley, J. (1970). An efficient context-free parsing algorithm. *Communications of the ACM*, *6*(8), 451–455. Reprinted in Grosz et al. (1986).

Eisner, J. (1997). Efficient generation in primitive optimality theory. In *ACL/EACL-97*, Madrid, Spain, pp. 313–320. ACL.

Ejerhed, E. I. (1988). Finding clauses in unrestricted text by finitary and stochastic methods. In *Second Conference on Applied Natural Language Processing*, pp. 219–227. ACL.

Elhadad, M. (1990). Types in functional unification grammars. In *ACL-90*, Pittsburgh, PA, pp. 157–164. ACL.

Elhadad, M. (1992). *Using Argumentation to Control Lexical Choice: A Functional Unification-Based Approach*. Ph.D. thesis, Columbia University.

Elhadad, M. (1993). FUF: The universal unifier — User Manual, version 5.2. Tech. rep., Ben Gurion University of the Negev.

Ellison, T. M. (1992). *The Machine Learning of Phonological Structure*. Ph.D. thesis, University of Western Australia.

Ellison, T. M. (1994). Phonological derivation in optimality theory. In *COLING-94*, Kyoto, pp. 1007–1013.

Emele, M. C. and Dorna, M. (1998). Ambiguity preserving machine translation using packed representations. In *COLING/ACL-98*, Montreal, pp. 365–371.

Evans, R. and Gazdar, G. (1996). DATR: A language for lexical knowledge representation. *Computational Linguistics*, *22*(2), 167–216.

Fasold, R. W. (1972). *Tense marking in Black English*. Center for Applied Linguistics, Washington, D.C.

Fass, D. (1988). *Collative Semantics: A Semantics for Natural Language*. Ph.D. thesis, New Mexico State University, Las Cruces, New Mexico. CRL Report No. MCCS-88-118.

Fass, D. (1991). met*: A method for discriminating metaphor and metonymy by computer. *Computational Linguistics*, *17*(1).

Fass, D. (1997). *Processing Metonymy and Metaphor*. Ablex Publishing, Greenwich, CT.

Fass, D., Martin, J. H., and Hinkelman, E. A. (Eds.). (1992). *Computational Intelligence: Special Issue on Non-Literal Language*, Vol. 8. Blackwell, Cambridge, MA.

Fauconnier, G. (1985). *Mental Spaces: Aspects of Meaning Construction in Natural Language*. MIT Press, Cambridge, MA.

Fellbaum, C. (Ed.). (1998). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Ferreira, F. and Henderson, J. M. (1991). How is verb information used during syntactic processing?. In Simpson, G. B. (Ed.), *Understanding Word and Sentence*, pp. 305–330. Elsevier Science.

Fikes, R. E. and Nilsson, N. J. (1971). Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, *2*, 189–208.

Fillmore, C. J. (1965). *Indirect Object Constructions in English and the Ordering of Transformations*. Mouton, The Hague.

Fillmore, C. J. (1968). The case for case. In Bach, E. W. and Harms, R. T. (Eds.), *Universals in Linguistic Theory*, pp. 1–88. Holt, Rinehart & Winston, New York.

Fillmore, C. J. (1977). Scenes-and-frames semantics. In Zampolli, A. (Ed.), *Linguistic Structures Processing*, pp. 55–79. North Holland, Amsterdam.

Fillmore, C. J. (1985). Frames and the semantics of understanding. *Quaderni di Semantica*, *VI*(2), 222–254.

Fillmore, C. J., Kay, P., and O'Connor, M. C. (1988). Regularity and idiomaticity in grammatical constructions: The case of Let Alone. *Language*, *64*(3), 510–538.

Finin, T. (1980). The semantic interpretation of nominal compounds. In *AAAI-80*, Stanford, CA, pp. 310–312.

Firbas, J. (1966). On defining the theme in functional sentence analysis. *Travaux Linguistiques de Prague*, *1*, 267–280.

Fisher, D., Soderland, S., McCarthy, J., Feng, F., and Lehnert, W. (1995). Description of the UMass system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Francisco, pp. 127–140. Morgan Kaufmann.

Fisher, M. F. K. (1968). *With Bold Knife and Fork*. Paragon Books, New York.

Flickinger, D., Pollard, C., and Wasow, T. (1985). Structure-sharing in lexical representation. In *ACL-85*, Chicago, IL, pp. 262–267. ACL.

Florian, R. and Yarowsky, D. (1999). Dynamic nonlocal language modeling via hierarchical topic-based adaptation. In *ACL-99*, College Park, MD, pp. 167–174. ACL.

Fodor, J. D. (1978). Parsing strategies and constraints on transformations. *Linguistic Inquiry*, *9*(3), 427–473.

Fodor, J. A. and Bever, T. G. (1965). The psychological reality of linguistic segments. *Journal of Verbal Learning and Verbal Behavior*, *4*, 414–420.

Foley, W. A. and van Valin, Jr., R. D. (1984). *Functional Syntax and Universal Grammar*. Cambridge University Press, Cambridge.

Ford, C., Fox, B., and Thompson, S. A. (1996). Practices in the construction of turns. *Pragmatics*, *6*, 427–454.

Ford, C. and Thompson, S. A. (1996). Interactional units in conversation: syntactic, intonational, and pragmatic resources for the management of turns. In Ochs, E., Schegloff, E. A., and Thompson, S. A. (Eds.), *Interaction and Grammar*, pp. 134–184. Cambridge University Press, Cambridge.

Ford, M. (1983). A method for obtaining measures of local parsing complexity through sentences. *Journal of Verbal Learning and Verbal Behavior*, *22*, 203–218.

Ford, M., Bresnan, J., and Kaplan, R. M. (1982). A competence-based theory of syntactic closure. In Bresnan, J. (Ed.), *The Mental Representation of Grammatical Relations*, pp. 727–796. MIT Press, Cambridge, MA.

Forney, Jr., G. D. (1973). The Viterbi algorithm. *Proceedings of the IEEE*, *61*(3), 268–278.

Fosler, E. (1996). On reversing the generation process in optimality theory. In *ACL-96*, Santa Cruz, CA, pp. 354–356. ACL.

Foster, D. W. (1989). *Elegy by W.W.: A Study in Attribution*. Associated University Presses, Cranbury, NJ.

Foster, D. W. (1996). Primary culprit. *New York*, 50–57. February 26.

Fox Tree, J. E. and Clark, H. H. (1997). Pronouncing "the" as "thee" to signal problems in speaking. *Cognition*, *62*, 151–167.

Frakes, W. B. and Baeza-Yates, R. (1992). *Information Retrieval: Data Structures and Algorithms*. Prentice Hall, Englewood Cliffs, NJ.

Francis, H. S., Gregory, M. L., and Michaelis, L. A. (1999). Are lexical subjects deviant?. In *CLS-99*. University of Chicago.

Francis, W. N. (1979). A tagged corpus – problems and prospects. In Greenbaum, S., Leech, G., and Svartvik, J. (Eds.), *Studies in English linguistics for Randolph Quirk*, pp. 192–209. Longman, London and New York.

Francis, W. N. and Kučera, H. (1982). *Frequency Analysis of English Usage*. Houghton Mifflin, Boston.

Frank, R. (1992). *Syntactic Locality and Tree Adjoining Grammar: Grammatical, Acquisition and Processing Perspectives*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

Frank, R. and Satta, G. (1999). Optimality theory and the generative complexity of constraint violability. *Computational Linguistics*. To appear.

Franz, A. (1996). *Automatic Ambiguity Resolution in Natural Language Processing*. Springer-Verlag, Berlin.

Franz, A. (1997). Independence assumptions considered harmful. In *ACL/EACL-97*, Madrid, Spain, pp. 182–189. ACL.

Fraser, N. M. and Gilbert, G. N. (1991). Simulating speech systems. *Computer Speech and Language*, *5*, 81–99.

Fraser, N. M. and Hudson, R. A. (1992). Inheritance in word grammar. *Computational Linguistics*, *18*(2), 133–158.

Frazier, L. and Clifton, Jr., C. (1996). *Construal*. MIT Press, Cambridge, MA.

Frederking, R., Rudnicky, A. I., and Hogan, C. (1997). Interactive speech translation in the DIPLOMAT project. In *Proceedings of the ACL-97 Spoken Language Translation Workshop*, Madrid, pp. 61–66. ACL.

Freitag, D. (1998). Multistrategy learning for information extraction. In *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI, pp. 161–169.

Friedl, J. E. F. (1997). *Master Regular Expressions*. O'Reilly, Cambridge.

Fromkin, V. and Ratner, N. B. (1998). Speech production. In Gleason, J. B. and Ratner, N. B. (Eds.), *Psycholinguistics*. Harcourt Brace, Fort Worth, TX.

Fry, D. B. (1959). Theoretical aspects of mechanical speech recognition. *Journal of the British Institution of Radio Engineers*, *19*(4), 211–218. Appears together with companion paper (Denes 1959).

Fujisaki, H. and Ohno, S. (1997). Comparison and assessment of models in the study of fundamental frequency contours of speech. In *ESCA workshop on Intonation: Theory Models and Applications*.

Fung, P. and McKeown, K. R. (1997). A technical word- and term-translation aid using noisy parallel corpora. *Machine Translation*, *12*, 53–87.

Gaizauskas, R., Wakao, T., Humphreys, K., Cunningham, H., and Wilks, Y. (1995). University of Sheffield: Description of the LaSIE system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Francisco, pp. 207–220. Morgan Kaufmann.

Gale, W. A. and Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational Linguistics*, *19*, 75–102.

Gale, W. A. and Church, K. W. (1994). What is wrong with adding one?. In Oostdijk, N. and de Haan, P. (Eds.), *Corpus-based Research into Language*, pp. 189–198. Rodopi, Amsterdam.

Gale, W. A., Church, K. W., and Yarowsky, D. (1992). Estimating upper and lower bounds on the performance of word-sense disambiguation programs. In *ACL-92*, Newark, DE, pp. 249–256. ACL.

Gale, W. A., Church, K. W., and Yarowsky, D. (1993). A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, *26*, 415–439.

Garnham, A. (1985). *Psycholinguistics*. Methuen, London.

Garnham, A., Traxler, M., Oakhill, J., and Gernsbacher, M. A. (1996). The locus of implicit causality effects in comprehension. *Journal of Memory and Language*, *35*, 517–534.

Garrett, M. F. (1975). The analysis of sentence production. In Bower, G. H. (Ed.), *The Psychology of Learning and Motivation*, Vol. 9. Academic, New York.

Garrod, S. C. and Sanford, A. J. (1994). Resolving sentences in a discourse context. In Gernsbacher, M. A. (Ed.), *Handbook of Psycholinguistics*, pp. 675–698. Academic Press, New York.

Garside, R. (1987). The CLAWS word-tagging system. In Garside, R., Leech, G., and Sampson, G. (Eds.), *The Computational Analysis of English*, pp. 30–41. Longman, London.

Garside, R., Leech, G., and McEnery, A. (1997). *Corpus Annotation*. Longman, London and New York.

Gazdar, G. (1981). Unbounded dependencies and coordinate structure. *Linguistic Inquiry*, *12*(2), 155–184.

Gazdar, G. (1982). Phrase structure grammar. In Jacobson, P. and Pullum, G. K. (Eds.), *The Nature of Syntactic Representation*, pp. 131–186. Reidel, Dordrecht.

Gazdar, G., Klein, E., Pullum, G. K., and Sag, I. A. (1985). *Generalized Phrase Structure Grammar*. Basil Blackwell, Oxford.

Gazdar, G. and Mellish, C. (1989). *Natural Language Processing in LISP*. Addison Wesley.

Gazdar, G., Pullum, G. K., Carpenter, B., Klein, E., Hukari, T. E., and Levine, R. D. (1988). Category structures. *Computational Linguistics*, *14*(1), 1–19.

Ge, N., Hale, J., and Charniak, E. (1998). A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora*. ACL.

Gentner, D. (1983). Structure mapping: A theoretical framework for analogy. *Cognitive Science*, *7*, 155–170.

Gentner, D. and France, I. M. (1988). The verb mutability effect: Studies of the combinatorial semantics of nouns and verbs. In *Lexical Ambiguity Resolution*, pp. 343–382. Morgan Kaufman, San Mateo, CA.

Gershman, A. V. (1977). Conceptual analysis of noun groups in English. In *IJCAI-77*, Cambridge, MA, pp. 132–138.

Gibson, E. (1991). *A Computational Theory of Human Linguistic Processing: Memory Limitations and Processing Breakdown*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.

Gibson, E. (1998). Linguistic complexity: Locality of syntactic dependencies. *Cognition*, *68*, 1–76.

Gilboy, E. and Sopena, J. M. (1996). Segmentation effects in the processing of complex NPs with relative clauses. In Carreiras, M., García-Albea, J. E., and Sebastián-Gallés, N. (Eds.), *Language Processing in Spanish*, pp. 191–206. Lawrence Erlbaum, Hillsdale, NJ.

Gildea, D. and Jurafsky, D. (1996). Learning bias and phonological rule induction. *Computational Linguistics*, *22*(4), 497–530.

Gildea, D., Jurafsky, D., Roland, D., and O'Connell, M. (2000). Automatic labeling of thematic roles. Submitted manuscript.

Givón, T. (1990). *Syntax: A functional typological introduction*. John Benjamins, Amsterdam.

Glennie, A. (1960). On the syntax machine and the construction of a universal compiler. Tech. rep. No. 2, Contr. NR 049-141, Carnegie Mellon University (at the time Carnegie Institute of Technology), Pittsburgh, PA†.

Godfrey, J., Holliman, E., and McDaniel, J. (1992). SWITCHBOARD: Telephone speech corpus for research and development. In *IEEE ICASSP-92*, pp. 517–520. IEEE.

Gold, B. and Morgan, N. (1999). *Speech and Audio Signal Processing*. Wiley Press.

Goldberg, A. E. (Ed.). (1995). *Constructions: A Construction Grammar approach to Argument Structure*. University of Chicago Press, Chicago.

Goldberg, E., Driedger, N., and Kittredge, R. (1994). Using natural-language processing to produce weather forecasts. *IEEE Expert*, *9*(2), 45–53.

Golding, A. R. and Roth, D. (1999). A winnow based approach to context-sensitive spelling correction. *Machine Learning*, *34*(1-3), 107–130. Special Issue on Machine Learning and Natural Language.

Golding, A. R. (1997). A bayesian hybrid method for context-sensitive spelling correction. In *Proceedings of the Third Workshop on Very Large Corpora*, Boston, MA, pp. 39–53. ACL.

Golding, A. R. and Schabes, Y. (1996). Combining trigram-based and feature-based methods for context-sensitive spelling correction. In *ACL-96*, Santa Cruz, CA, pp. 71–78. ACL.

Goldman, N. (1975). Conceptual generation. In Schank, R. C. (Ed.), *Conceptual Information Processing*, chap. 6. North-Holland.

Goldsmith, J. (1976). *Autosegmental Phonology*. Ph.D. thesis, MIT, Cambridge, MA.

Goldsmith, J. (1993). Harmonic phonology. In Goldsmith, J. (Ed.), *The Last Phonological Rule*, pp. 21–60. University of Chicago Press, Chicago.

Goldsmith, J. (Ed.). (1995). *The Handbook of Phonological Theory*. Basil Blackwell, Oxford.

Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, *40*, 16–264.

Good, M. D., Whiteside, J. A., Wixon, D. R., and Jones, S. J. (1984). Building a user-derived interface. *Communications of the ACM*, *27*(10), 1032–1043.

Goodman, J. (1997). Probabilistic feature grammars. In *Proceedings of the International Workshop on Parsing Technology*.

Goodwin, C. (1996). Transparent vision. In Ochs, E., Schegloff, E. A., and Thompson, S. A. (Eds.), *Interaction and Grammar*. Cambridge University Press, Cambridge.

Gordon, D. and Lakoff, G. (1971). Conversational postulates. In *CLS-71*, pp. 200–213. University of Chicago. Reprinted in Peter Cole and Jerry L. Morgan (Eds.), *Speech Acts: Syntax and Semantics Volume 3*, Academic, 1975.

Gordon, P. C., Grosz, B. J., and Gilliom, L. A. (1993). Pronouns, names, and the centering of attention in discourse. *Cognitive Science*, *17*(3), 311–347.

Gorin, A. L., Riccardi, G., and Wright, J. H. (1997). How may I help you?. *Speech Communication*, *23*(1), 113–127.

Götz, T., Meurers, W. D., and Gerdemann, D. (1997). The ConTroll manual. Tech. rep., Seminar für Sprachwissenschaft, Universität Tübingen.

Gould, J. D., Conti, J., and Hovanyecz, T. (1983). Composing letters with a simulated listening typewriter. *Communications of the ACM*, *26*(4), 295–308.

Gould, J. D. and Lewis, C. (1985). Designing for usability: Key principles and what designers think. *Communications of the ACM*, *28*(3), 300–311.

Graff, D. (1997). The 1996 Broadcast News speech and language-model corpus. In *Proceedings DARPA Speech Recognition Workshop*, Chantilly, VA, pp. 11–14. Morgan Kaufmann.

Graham, S. L., Harrison, M. A., and Ruzzo, W. L. (1980). An improved context-free recognizer. *ACM Transactions on Programming Languages and Systems*, *2*(3), 415–462.

Grainger, J., Colé, P., and Segui, J. (1991). Masked morphological priming in visual word recognition. *Journal of Memory and Language*, *30*, 370–384.

Green, B. F., Wolf, A. K., Chomsky, C., and Laughery, K. (1961). Baseball: An automatic question answerer. In *Proceedings of the Western Joint Computer Conference 19*, pp. 219–224. Reprinted in Grosz et al. (1986).

Green, G. M. (1974). *Semantics and Syntactic Regularity*. Indiana University Press, Bloomington.

Greenberg, S., Ellis, D., and Hollenback, J. (1996). Insights into spoken language gleaned from phonetic transcription of the Switchboard corpus. In *ICSLP-96*, Philadelphia, PA, pp. S24–27.

Greene, B. B. and Rubin, G. M. (1971). Automatic grammatical tagging of English. Department of Linguistics, Brown University, Providence, Rhode Island.

Gregory, M. L., Raymond, W. D., Bell, A., Fosler-Lussier, E., and Jurafsky, D. (1999). The effects of collocational strength and contextual predictability in lexical production. In *CLS-99*. University of Chicago.

Grice, H. P. (1957). Meaning. *Philosophical Review*, *67*, 377–388. Reprinted in *Semantics*, edited by D. D. Steinberg & L. A. Jakobovits (1971), Cambridge University Press, pages 53–59.

Grice, H. P. (1975). Logic and conversation. In Cole, P. and Morgan, J. L. (Eds.), *Speech Acts: Syntax and Semantics Volume 3*, pp. 41–58. Academic Press, New York.

Grice, H. P. (1978). Further notes on logic and conversation. In Cole, P. (Ed.), *Pragmatics: Syntax and Semantics Volume 9*, pp. 113–127. Academic Press, New York.

Grishman, R. and Sundheim, B. (1995). Design of the MUC-6 evaluation. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Francisco, pp. 1–11. Morgan Kaufmann.

Gropen, J., Pinker, S., Hollander, M., Goldberg, R., and Wilson, R. (1989). The learnability and acquisition of the dative alternation in English. *Language*, *65*(2), 203–257.

Grosjean, F. (1980). Spoken word recognition processes and the gating paradigm. *Perception and Psychophysics*, *28*, 267–283.

Grosz, B. and Hirschberg, J. (1992). Some intonational characteristics of discourse structure. In *ICSLP-92*, Vol. 1, pp. 429–432.

Grosz, B. J. (1974). The structure of task-oriented dialogs. In *Proceedings of the IEEE Symposium on Speech Recognition: Contributed Papers*, Pittsburgh, PA†, pp. 250–253.

Grosz, B. J. (1977a). The representation and use of focus in a system for understanding dialogs. In *IJCAI-77*, Cambridge, MA, pp. 67–76. Morgan Kaufmann. Reprinted in Grosz et al. (1986).

Grosz, B. J. (1977b). *The Representation and Use of Focus in Dialogue Understanding*. Ph.D. thesis, University of California, Berkeley.

Grosz, B. J., Jones, K. S., and Webber, B. L. (Eds.). (1986). *Readings in Natural Language Processing*. Morgan Kaufmann, Los Altos, Calif.

Grosz, B. J., Joshi, A. K., and Weinstein, S. (1983). Providing a unified account of definite noun phrases in English. In *ACL-83*, pp. 44–50. ACL.

Grosz, B. J., Joshi, A. K., and Weinstein, S. (1995). Centering: A framework for modelling the local coherence of discourse. *Computational Linguistics*, *21*(2).

Grosz, B. J. and Sidner, C. L. (1980). Plans for discourse. In Cohen, P. R., Morgan, J., and Pollack, M. E. (Eds.), *Intentions in Communication*, pp. 417–444. MIT Press, Cambridge, MA.

Grosz, B. J. and Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics*, *12*(3), 175–204.

Gruber, J. S. (1965). *Studies in Lexical Relations*. Ph.D. thesis, MIT, Cambridge, MA.

Grudin, J. T. (1983). Error patterns in novice and skilled transcription typing. In Cooper, W. E. (Ed.), *Cognitive Aspects of Skilled Typewriting*, pp. 121–139. Springer-Verlag, New York.

Guindon, R. and Kintsch, W. (1984). Priming macropropositions: Evidence for the primacy of macropropositions in the memory for text. *Journal of Verbal Learning and Verbal Behavior*, *23*, 508–518.

Guindon, R. (1988). A multidisciplinary perspective on dialogue structure in user-advisor dialogues. In Guindon, R. (Ed.), *Cognitive Science And Its Applications For Human-Computer Interaction*, pp. 163–200. Lawrence Erlbaum, Hillsdale, NJ.

Gundel, J. K., Hedberg, N., and Zacharski, R. (1993). Cognitive status and the form of referring expressions in discourse. *Language*, *69*(2), 274–307.

Gupta, V., Lennig, M., and Mermelstein, P. (1988). Fast search strategy in a large vocabulary word recognizer. *Journal of the Acoustical Society of America*, *84*(6), 2007–2017.

Gupta, V., Lennig, M., and Mermelstein, P. (1992). A language model for very large-vocabulary speech recognition. *Computer Speech and Language*, *6*, 331–344.

Guthrie, J. A., Guthrie, L., Wilks, Y., and Aidinejad, H. (1991). Subject-dependent co-occurrence and word sense disambiguation. In *ACL-91*, Berkeley, CA, pp. 146–152. ACL.

Hain, T., Woodland, P. C., Niesler, T. R., and Whittaker, E. W. D. (1999). The 1998 HTK system for transcription of conversational telephone speech. In *IEEE ICASSP-99*, pp. 57–60. IEEE.

Hajič, J. (1998). *Building a Syntactically Annotated Corpus: The Prague Dependency Treebank*, pp. 106–132. Karolinum, Prague/Praha.

Halliday, M. A. K. (1985). *An Introduction to Functional Grammar*. Edward Arnold, London.

Halliday, M. A. K. and Hasan, R. (1976). *Cohesion in English*. Longman, London. English Language Series, Title No. 9.

Hammond, M. (1997). Parsing in OT. Alternative title "Parsing syllables: Modeling OT computationally". Rutgers Optimality Archive ROA-222-1097.

Hankamer, J. (1986). Finite state morphology and left to right phonology. In *Proceedings of the Fifth West Coast Conference on Formal Linguistics*, pp. 29–34.

Hankamer, J. (1989). Morphological parsing and the lexicon. In Marslen-Wilson, W. (Ed.), *Lexical Representation and Process*, pp. 392–408. MIT Press, Cambridge, MA.

Hankamer, J. and Black, H. A. (1991). Current approaches to computational morphology. Unpublished manuscript.

Harris, Z. S. (1946). From morpheme to utterance. *Language*, *22*(3), 161–183.

Harris, Z. S. (1962). *String Analysis of Sentence Structure*. Mouton, The Hague.

Hartley, A. and Paris, C. (1997). Multilingual document production: From support for translating to support for authoring. *Machine Translation*, *12*, 109–128.

Haviland, S. E. and Clark, H. H. (1974). What's new? Acquiring new information as a process in comprehension. *Journal of Verbal Learning and Verbal Behaviour*, *13*, 512–521.

Hayes, E. and Bayer, S. (1991). Metaphoric generalization through sort coercion. In *ACL-91*, Berkeley, CA, pp. 222–228. ACL.

Hearst, M. A. (1991). Noun homograph disambiguation. In *Proceedings of the 7th Annual Conference of the University of Waterloo Centre for the New OED and Text Research*, Oxford, pp. 1–19.

Hearst, M. A. (1997). TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, *23*(1), 33–64.

Hearst, M. A. and Pedersen, J. O. (1996). Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *SIGIR-96*, Zurich, Switzerland, pp. 76–84. ACM.

Heeman, P. A. (1999). POS tags and decision trees for language modeling. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, College Park, MD, pp. 129–137. ACL.

Heeman, P. A. and Allen, J. (1999). Speech repairs, intonational phrases and discourse markers: Modeling speakers' utterances in spoken dialog. *Computational Linguistics*, *25*(4).

Heikkilä, J. (1995). A TWOL-based lexicon and feature system for English. In Karlsson, F., Voutilainen, A., Heikkilä, J., and Anttila, A. (Eds.), *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*, pp. 103–131. Mouton de Gruyter, Berlin.

Heim, I. and Kratzer, A. (1998). *Semantics in a Generative Grammar*. Blackwell Publishers, Malden, MA.

Hemphill, C. T., Godfrey, J., and Doddington, G. R. (1990). The ATIS spoken language systems pilot corpus. In *Proceedings DARPA Speech and Natural Language Workshop*, Hidden Valley, PA, pp. 96–101. Morgan Kaufmann.

Hermansky, H. (1990). Perceptual linear predictive (PLP) analysis of speech. *Journal of the Acoustical Society of America*, *87*(4), 1738–1752.

Hertz, J., Krogh, A., and Palmer, R. G. (1991). *Introduction to the Theory of Neural Computation*. Addison-Wesley Publishing Company.

Hindle, D. (1983). Deterministic parsing of syntactic non-fluencies. In *ACL-83*, Cambridge, MA, pp. 123–128. ACL.

Hindle, D. and Rooth, M. (1990). Structural ambiguity and lexical relations. In *Proceedings DARPA Speech and Natural Language Workshop*, Hidden Valley, PA, pp. 257–262. Morgan Kaufmann.

Hindle, D. and Rooth, M. (1991). Structural ambiguity and lexical relations. In *ACL-91*, Berkeley, CA, pp. 229–236. ACL.

Hinkelman, E. A. and Allen, J. (1989). Two constraints on speech act ambiguity. In *ACL-89*, Vancouver, Canada, pp. 212–219. ACL.

Hintikka, J. (1969). Semantics for propositional attitudes. In Davis, J. W., Hockney, D. J., and Wilson, W. K. (Eds.), *Philosophical Logic*, pp. 21–45. D. Reidel, Dordrecht, Holland.

Hirschberg, J. and Litman, D. J. (1993). Empirical studies on the disambiguation of cue phrases. *Computational Linguistics*, *19*(3), 501–530.

Hirschberg, J. and Nakatani, C. (1996). A prosodic analysis of discourse segments in direction-giving monologues. In *ACL-96*, Santa Cruz, CA, pp. 286–293. ACL.

Hirschberg, J. and Pierrehumbert, J. (1986). The intonational structuring of discourse. In *ACL-86*, New York, pp. 136–144. ACL.

Hirschman, L. and Pao, C. (1993). The cost of errors in a spoken language system. In *EUROSPEECH-93*, pp. 1419–1422.

Hirst, G. (1987). *Semantic Interpretation and the Resolution of Ambiguity*. Cambridge University Press, Cambridge.

Hirst, G. (1988). Resolving lexical ambiguity computationally with spreading activation and polaroid words. In Small, S. L., Cottrell, G. W., and Tanenhaus, M. K. (Eds.), *Lexical ambiguity resolution: Perspectives from psycholinguistics, neuropsychology, and artificial intelligence*, pp. 73–108. Morgan Kaufmann, San Mateo, CA.

Hirst, G. and Charniak, E. (1982). Word sense and case slot disambiguation. In *AAAI-82*, pp. 95–98.

Hobbs, J. R. (1977). 38 examples of elusive antecedents from published texts. Tech. rep. 77-2, Department of Computer Science, City University of New York.

Hobbs, J. R. (1978). Resolving pronoun references. *Lingua*, *44*, 311–338. Reprinted in Grosz et al. (1986).

Hobbs, J. R. (1979a). Coherence and coreference. *Cognitive Science*, *3*, 67–90.

Hobbs, J. R. (1979b). Metaphor, metaphor schemata, and selective inferencing. Tech. rep. Technical Note 204, SRI, San Mateo, CA.

Hobbs, J. R. (1990). *Literature and Cognition*. CSLI Lecture Notes 21.

Hobbs, J. R., Appelt, D., Bear, J., Israel, D., Kameyama, M., Stickel, M. E., and Tyson, M. (1997). FASTUS: A cascaded finite-state transducer for extracting information from natural-language text. In Roche, E. and Schabes, Y. (Eds.), *Finite-State Devices for Natural Language Processing*, pp. 383–406. MIT Press, Cambridge, MA.

Hobbs, J. R. and Shieber, S. M. (1987). An algorithm for generating quantifier scopings. *Computational Linguistics*, *13*(1), 47–55.

Hobbs, J. R., Stickel, M. E., Appelt, D. E., and Martin, P. (1993). Interpretation as abduction. *Artificial Intelligence*, *63*, 69–142.

Hofstadter, D. R. (1997). *Le Ton beau de Marot*. Basic Books, New York.

Holmes, D. I. (1994). Authorship attribution. *Computers and the Humanities*, *28*, 87–106.

Hopcroft, J. E. and Ullman, J. D. (1979). *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA.

Householder, F. W. (1995). Dionysius Thrax, the *technai*, and Sextus Empiricus. In Koerner, E. F. K. and Asher, R. E. (Eds.), *Concise History of the Language Sciences*, pp. 99–103. Elsevier Science, Oxford.

Hovy, E. and Radev, D. (Eds.). (1998). *Intelligent Text Summarization: Papers from the 1998 AAAI Symposium*, Menlo Park, CA. AAAI, AAAI Press.

Hovy, E. H. (1988a). *Generating Natural Language Under Pragmatic Constraints*. Lawrence Erlbaum, Hillsdale, NJ.

Hovy, E. H. (1988b). Planning coherent multisentential text. In *ACL-88*, Buffalo, NY, pp. 163–169. ACL.

Hovy, E. H. (1990). Parsimonious and profligate approaches to the question of discourse structure relations. In *Proceedings of the Fifth International Workshop on Natural Language Generation*, Dawson, PA, pp. 128–136.

Howes, D. (1957). On the relation between the intelligibility and frequency of occurrence of English words. *Journal of the Acoustical Society of America*, *29*, 296–305.

Hu, J., Brown, M. K., and Turin, W. (1996). HMM based on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(10), 1039–1045.

Huang, X. D. and Jack, M. A. (1989). Semi-continuous hidden Markov models for speech signals. *Computer Speech and Language*, *3*, 239–252.

Hudson, R. A. (1984). *Word Grammar*. Basil Blackwell, Oxford.

Huffman, D. A. (1954). The synthesis of sequential switching circuits. *Journal of the Franklin Institute*, *3*, 161–191. Continued in Volume 4.

Huffman, S. (1996). Learning information extraction patterns from examples. In Wertmer, S., Riloff, E., and Scheller, G. (Eds.), *Connectionist, Statistical, and Symbolic Approaches to Learning Natural Language Processing*, pp. 246–260. Springer, Berlin.

Hull, D. (1996). Stemming algorithms – a case study for detailed evaluation. *Journal of the American Society for Information Science*, *47*(1), 70–84.

Hull, J. J. and Srihari, S. N. (1982). Experiments in text recognition with binary *n*-gram and Viterbi algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-4*, 520–530.

Huls, C., Bos, E., and Classen, W. (1995). Automatic referent resolution of deictic and anaphoric expressions. *Computational Linguistics*, *21*(1), 59–79.

Hunt, A. J. and Black, A. W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. In *IEEE ICASSP-96*, Atlanta, GA, Vol. 1, pp. 373–376. IEEE.

Hutchins, J. (1997). From first conception to first demonstration: the nascent years of machine translation, 1947–1954. A chronology. *Machine Translation*, *12*, 192–252.

Hutchins, W. J. and Somers, H. L. (1992). *An Introduction to Machine Translation*. Academic Press.

Hutchins, W. J. (1986). *Machine Translation: Past, Present, Future*. Ellis Horwood, Chichester, England.

Huybregts, R. (1984). The weak inadequacy of context-free phrase structure grammars. In de Haan, G., Trommele, M., and Zonneveld, W. (Eds.), *Van Periferie naar Kern*. Foris, Dordrecht†. Cited in Pullum (1991).

Ide, N. M. and Veronis, J. (Eds.). (1998). *Computational Linguistics: Special Issue on Word Sense Disambiguation*, Vol. 24. MIT Press, Cambridge, MA.

Irons, E. T. (1961). A syntax directed compiler for ALGOL 60. *Communications of the ACM*, *4*, 51–55.

Itakura, F. (1975). Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *ASSP-32*, 67–72.

Iverson, E. and Helmreich, S. (1992). Metallel: An integrated approach to non-literal phrase interpretation. *Computational Intelligence*, *8*(3).

Iyer, R. and Ostendorf, M. (1997). Transforming out-of-domain estimates to improve in-domain language models. In *EUROSPEECH-97*, pp. 1975–1978.

Jackendoff, R. (1972). *Semantic Interpretation in Generative Grammar*. MIT Press, Cambridge, MA.

Jackendoff, R. (1975). Morphological and semantic regularities in the lexicon. *Language*, *51*(3), 639–671.

Jackendoff, R. (1983). *Semantics and Cognition*. MIT Press, Cambridge, MA.

Jackendoff, R. (1990). *Semantic Structures*. MIT Press, Cambridge, MA.

Jacobs, P. (1985). *A Knowledge-Based Approach to Language Generation*. Ph.D. thesis, University of California, Berkeley, CA. Available as University of California at Berkeley Computer Science Division Tech. rep. #86/254.

Jacobs, P. (1987). Knowledge-based natural language generation. *Artificial Intelligence*, *33*, 325–378.

Jacobs, P. and Rau, L. (1990). SCISOR: A system for extracting information from online news. *Communications of the ACM*, *33*(11), 88–97.

Jakobson, R. (1939). Observations sur le classement phonologique des consonnes. In Blancquaert, E. and Pée, W. (Eds.), *Proceedings of the Third International Congress of Phonetic Sciences*, Ghent, pp. 34–41.

Janssen, T. M. (1997). Compositionality. In van Benthem, J. and ter Meulen, A. (Eds.), *Handbook of Logic and Language*, chap. 7, pp. 417–473. North-Holland, Amsterdam.

Jardine, N. and van Rijsbergen, C. J. (1971). The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, *7*, 217–240.

Järvinen, T. and Tapanainen, P. (1997). A dependency parser for English. Tech. rep. TR-1, Department of General Linguistics, University of Helsinki, Helsinki.

Jefferson, G. (1984). Notes on a systematic deployment of the acknowledgement tokens 'yeah' and 'mm hm'. *Papers in Linguistics*, pp. 197–216.

Jeffreys, H. (1948). *Theory of Probability*. Clarendon Press, Oxford. 2nd edn Section 3.23.

Jelinek, F., Mercer, R. L., Bahl, L. R., and Baker, J. K. (1977). Perplexity – a measure of the difficulty of speech recognition tasks. *Journal of the Acoustical Society of America*, *62*, S63. Supplement 1.

Jelinek, F. (1969). A fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development*, *13*, 675–685.

Jelinek, F. (1990). Self-organized language modeling for speech recognition. In Waibel, A. and Lee, K.-F. (Eds.), *Readings in Speech Recognition*, pp. 450–506. Morgan Kaufmann, Los Altos. Originally distributed as IBM technical report in 1985.

Jelinek, F., Lafferty, J. D., Magerman, D. M., Mercer, R. L., Ratnaparkhi, A., and Roukos, S. (1994). Decision tree parsing using a hidden derivation model. In *ARPA Human Language Technologies Workshop*, Plainsboro, N.J., pp. 272–277. Morgan Kaufmann.

Jelinek, F. and Mercer, R. L. (1980). Interpolated estimation of Markov source parameters from sparse data. In Gelsema, E. S. and Kanal, L. N. (Eds.), *Proceedings, Workshop on Pattern Recognition in Practice*, pp. 381–397. North Holland, Amsterdam.

Jelinek, F. (1976). Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, *64*(4), 532–557.

Jelinek, F. (1997). *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA.

Jelinek, F. and Lafferty, J. D. (1991). Computation of the probability of initial substring generation by stochastic context-free grammars. *Computational Linguistics*, *17*(3), 315–323.

Jelinek, F., Mercer, R. L., and Bahl, L. R. (1975). Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, *IT-21*(3), 250–256.

Jing, H. and McKeown, K. R. (1998). Combining multiple, large-scale resources in a reusable lexicon for natural language generation. In *COLING/ACL-98*, Montreal, pp. 607–613. ACL.

Johnson, C. D. (1972). *Formal Aspects of Phonological Description*. Mouton, The Hague. Monographs on Linguistic Analysis No. 3.

Johnson, C. (1999). Syntactic and semantic principles of FrameNet annotation, version 1. Tech. rep. TR-99-018, ICSI, Berkeley, CA.

Johnson, K. (1997). *Acoustic and Auditory Phonetics*. Blackwell, Cambridge, MA.

Johnson, M. K., Bransford, J. D., and Solomon, S. K. (1973). Memory for tacit implications of sentences. *Journal of Experimental Psychology*, *98*, 203–205.

Johnson, M. (1984). A discovery procedure for certain phonological rules. In *COLING-84*, Stanford, CA, pp. 344–347.

Johnson, M. (1988). *Attribute-Value Logic and the Theory of Grammar*. CSLI Lecture Notes. Chicago University Press, Chicago.

Johnson, M. (1990). Expressing disjunctive and negative feature constraints with classical first-order logic. In *ACL-90*, Pittsburgh, PA, pp. 173–179. ACL.

Johnson, S. C. and Lesk, M. E. (1978). Language development tools. *Bell System Technical Journal*, *57*(6), 2155–2175.

Johnson-Laird, P. N. (1983). *Mental Models*. Harvard University Press, Cambridge, MA.

Jones, M. A. and Eisner, J. (1992). A probabilistic parser applied to software testing documents. In *AAAI-92*, San Jose, CA, pp. 322–328.

Jones, M. A. and McCoy, K. (1992). Transparently-motivated metaphor generation. In Dale, R., Hovy, E. H., Rösner, D., and Stock, O. (Eds.), *Aspects of Automated Natural Language Generation*, Lecture Notes in Artificial Intelligence 587, pp. 183–198. Springer Verlag, Berlin.

Jones, M. P. (1997). *Spoken Language Help of High Functionality Systems*. Ph.D. thesis, University of Colorado, Boulder, CO.

Jones, M. P. and Martin, J. H. (1997). Contextual spelling correction using latent semantic analysis. In *Proceedings of the 5th Conference on Applied Natural Language Processing (ANLP'97)*, Washington, D.C., pp. 166–173. ACL.

Joshi, A. K. (1985). Tree adjoining grammars: how much context-sensitivity is required to provide reasonable structural descriptions?. In Dowty, D. R., Karttunen, L., and Zwicky, A. (Eds.), *Natural Language Parsing*, pp. 206–250. Cambridge University Press, Cambridge.

Joshi, A. K. and Hopely, P. (1999). A parser from antiquity. In Kornai, A. (Ed.), *Extended Finite State Models of Language*, pp. 6–15. Cambridge University Press, Cambridge.

Joshi, A. K. and Kuhn, S. (1979). Centered logic: The role of entity centered sentence representation in natural language inferencing. In *IJCAI-79*, pp. 435–439.

Joshi, A. K. and Srinivas, B. (1994). Disambiguation of super parts of speech (or supertags): Almost parsing. In *COLING-94*, Kyoto, pp. 154–160.

Joshi, A. K. and Weinstein, S. (1981). Control of inference: Role of some aspects of discourse structure – centering. In *IJCAI-81*, pp. 385–387.

Juliano, C. and Tanenhaus, M. K. (1993). Contingent frequency effects in syntactic ambiguity resolution. In *COGSCI-93*, Boulder, CO, pp. 593–598.

Juola, P. (1999). Measuring linguistic complexity. Presented at the 4th Conference on Conceptual Structure, Discourse, and Language (CSDL-4), Georgia.

Jurafsky, D. (1992). *An On-line Computational Model of Human Sentence Interpretation: A Theory of the Representation and Use of Linguistic Knowledge*. Ph.D. thesis, University of California, Berkeley, CA. Available as University of California at Berkeley Computer Science Division Tech. rep. #92/676.

Jurafsky, D. (1996). A probabilistic model of lexical and syntactic access and disambiguation. *Cognitive Science*, *20*, 137–194.

Jurafsky, D., Bates, R., Coccaro, N., Martin, R., Meteer, M., Ries, K., Shriberg, E., Stolcke, A., Taylor, P., and Van Ess-Dykema, C. (1997). Automatic detection of discourse structure for speech recognition and understanding. In *Proceedings of the 1997 IEEE Workshop on Speech Recognition and Understanding*, Santa Barbara, pp. 88–95.

Jurafsky, D., Bell, A., Fosler-Lussier, E., Girand, C., and Raymond, W. D. (1998). Reduction of English function words in Switchboard. In *ICSLP-98*, Sydney, Vol. 7, pp. 3111–3114.

Jurafsky, D., Wooters, C., Tajchman, G., Segal, J., Stolcke, A., Fosler, E., and Morgan, N. (1994). The Berkeley restaurant project. In *ICSLP-94*, Yokohama, Japan, pp. 2139–2142.

Jurafsky, D., Wooters, C., Tajchman, G., Segal, J., Stolcke, A., Fosler, E., and Morgan, N. (1995). Using a stochastic context-free grammar as a language model for speech recognition. In *IEEE ICASSP-95*, pp. 189–192. IEEE.

Kameyama, M. (1986). A property-sharing constraint in centering. In *ACL-86*, New York, pp. 200–206. ACL.

Kamm, C. A. (1994). User interfaces for voice applications. In Roe, D. B. and Wilpon, J. G. (Eds.), *Voice Communication Between Humans and Machines*, pp. 422–442. National Academy Press, Washington, D.C.

Kamp, H. (1981). A theory of truth and semantic representation. In Groenendijk, J. A. G., Janssen, T. M. V., and Stokhof, M. B. J. (Eds.), *Formal Methods in the Study of Language*, Vol. 1, pp. 277–322. Mathematisch Centrum, Amsterdam.

Kaplan, R. M. (1987). Three seductions of computational psycholinguistics. In Whitelock, P., Wood, M. M., Somers, H. L., Johnson, R., and Bennett, P. (Eds.), *Linguistic Theory and Computer Applications*, pp. 149–188. Academic Press, London.

Kaplan, R. M. and Bresnan, J. (1982). Lexical-functional grammar: A formal system for grammatical representation. In Bresnan, J. (Ed.), *The Mental Representation of Grammatical Relations*, pp. 173–281. MIT Press, Cambridge, MA.

Kaplan, R. M. and Kay, M. (1981). Phonological rules and finite-state transducers. Paper presented at the Annual meeting of the Linguistics Society of America. New York.

Kaplan, R. M. and Kay, M. (1994). Regular models of phonological rule systems. *Computational Linguistics*, *20*(3), 331–378.

Karlsson, F., Voutilainen, A., Heikkilä, J., and Anttila, A. (Eds.). (1995). *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*. Mouton de Gruyter, Berlin.

Karttunen, L. (1969). Pronouns and variables. In *CLS-69*, pp. 108–116. University of Chicago.

Karttunen, L. (1983). Kimmo: A general morphological processor. In *Texas Linguistics Forum 22*, pp. 165–186.

Karttunen, L. (1993). Finite-state constraints. In Goldsmith, J. (Ed.), *The Last Phonological Rule*, pp. 173–194. University of Chicago Press.

Karttunen, L. (1998). The proper treatment of optimality in computational phonology. In *Proceedings of FSMNLP'98: International Workshop on Finite-State Methods in Natural Language Processing*, Bilkent University. Ankara, Turkey, pp. 1–12.

Karttunen, L. (1999). Comments on Joshi. In Kornai, A. (Ed.), *Extended Finite State Models of Language*, pp. 16–18. Cambridge University Press, Cambridge.

Karttunen, L. and Kay, M. (1985). Structure sharing with binary trees. In *ACL-85*, Chicago, pp. 133–136. ACL.

Kasami, T. (1965). An efficient recognition and syntax analysis algorithm for context-free languages. Tech. rep. AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA†.

Kashyap, R. L. and Oommen, B. J. (1983). Spelling correction using probabilistic methods. *Pattern Recognition Letters*, *2*, 147–154.

Kasper, R. T. (1988). An experimental parser for systemic grammars. In *COLING-88*, Budapest, pp. 309–312.

Kasper, R. T. and Rounds, W. C. (1986). A logical semantics for feature structures. In *ACL-86*, New York, pp. 257–266. ACL.

Katz, J. J. and Fodor, J. A. (1963). The structure of a semantic theory. *Language*, *39*, 170–210.

Katz, S. M. (1987). Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *35*(3), 400–401.

Kawamoto, A. H. (1988). Distributed representations of ambiguous words and their resolution in connectionist networks. In Small, S. L., Cottrell, G. W., and Tanenhaus, M. (Eds.), *Lexical Ambiguity Resolution*, pp. 195–228. Morgan Kaufman, San Mateo, CA.

Kay, M. (1967). Experiments with a powerful parser. In *Proc. 2eme Conference Internationale sur le Traitement Automatique des Langues*, Grenoble.

Kay, M. (1973). The MIND system. In Rustin, R. (Ed.), *Natural Language Processing*, pp. 155–188. Algorithmics Press, New York.

Kay, M. (1979). Functional grammar. In *BLS-79*, Berkeley, CA, pp. 142–158.

Kay, M. (1980/1997). The proper place of men and machines in language translation. *Machine Translation*, *12*, 3–23. First appeared as a Xerox PARC Working paper in 1980.

Kay, M. (1984). Functional unification grammar: A formalism for machine translation. In *COLING-84*, Stanford, CA, pp. 75–78.

Kay, M. (1985). Parsing in functional unification grammar. In Dowty, D. R., Karttunen, L., and Zwicky, A. (Eds.), *Natural Language Parsing*, pp. 251–278. Cambridge University Press, Cambridge.

Kay, M. (1987). Nonconcatenative finite-state morphology. In *Proceedings of the Third Conference of the European Chapter of the ACL (EACL-87)*, Copenhagen, Denmark, pp. 2–10. ACL.

Kay, M., Gawron, J. M., and Norvig, P. (1992). *Verbmobil: A Translation System for Face-to-Face Dialog*. CSLI.

Kay, M. and Röscheisen, M. (1993). Text-translation alignment. *Computational Linguistics*, *19*, 121–142.

Kay, P. and Fillmore, C. J. (1999). Grammatical constructions and linguistic generalizations: The What's X Doing Y? construction. *Language*, *75*(1), 1–33.

Keating, P. A., Byrd, D., Flemming, E., and Todaka, Y. (1994). Phonetic analysis of word and segment variation using the TIMIT corpus of American English. *Speech Communication*, *14*, 131–142.

Keenan, J. M., Potts, G. R., Golding, J. M., and Jennings, T. M. (1990). Which elaborative inferences ar drawn during reading? A question of methodologies. In Balota, D. A., d'Arcais, G. B. F., and Rayner, K. (Eds.), *Comprehension processes in reading*, pp. 377–402. Lawrence Erlbaum, Hillsdale, NJ.

Kehler, A. (1993). The effect of establishing coherence in ellipsis and anaphora resolution. In *ACL-93*, Columbus, Ohio, pp. 62–69. ACL.

Kehler, A. (1994a). Common topics and coherent situations: Interpreting ellipsis in the context of discourse inference. In *ACL-94*, Las Cruces, New Mexico, pp. 50–57. ACL.

Kehler, A. (1994b). Temporal relations: Reference or discourse coherence?. In *ACL-94*, Las Cruces, New Mexico, pp. 319–321. ACL.

Kehler, A. (1997a). Current theories of centering for pronoun interpretation: A critical evaluation. *Computational Linguistics*, *23*(3), 467–475.

Kehler, A. (1997b). Probabilistic coreference in information extraction. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing (EMNLP-97)*, Providence, RI, pp. 163–173.

Kehler, A. (2000). *Coherence, Reference, and the Theory of Grammar*. CSLI Publications.

Kelley, L. G. (1979). *The True Interpreter: A History of Translation Theory and Practice in the West*. St. Martin's Press, New York.

Kelly, E. F. and Stone, P. J. (1975). *Computer Recognition of English Word Senses*. North-Holland, Amsterdam.

Kennedy, C. and Boguraev, B. (1996). Anaphora for everyone: Pronominal anaphora resolution without a parser. In *COLING-96*, Copenhagen, pp. 113–118.

Kernighan, M. D., Church, K. W., and Gale, W. A. (1990). A spelling correction program base on a noisy channel model. In *COLING-90*, Helsinki, Vol. II, pp. 205–211.

Khudanpur, S. and Wu, J. (1999). A maximum entropy language model integrating n-grams and topic dependencies for conversational speech recognition. In *IEEE ICASSP-99*, pp. 553–556. IEEE.

Kiefer, B., Krieger, H.-U., Carroll, J., and Malouf, R. (1999). A bag of useful techniques for efficient and robust parsing. In *ACL-99*, College Park, MD, pp. 473–480.

Kilgarriff, A. and Palmer, M. (Eds.). (2000). *Computing and the Humanities: Special Issue on SENSEVAL*, Vol. 34. Kluwer.

Kilgarriff, A. and Rosenzweig, J. (2000). Framework and results for English SENSEVAL. *Computers and the Humanities*, *34*(1-2).

King, J. and Just, M. A. (1991). Individual differences in syntactic processing: The role of working memory. *Journal of Memory and Language*, *30*, 580–602.

King, P. (1989). *A Logical Formalism for Head-Driven Phrase Structure Grammar*. Ph.D. thesis, University of Manchester†. Cited in Carpenter (1992)).

Kintsch, W. (1974). *The Representation of Meaning in Memory*. Wiley, New York.

Kintsch, W. (1988). The role of knowledge in discourse comprehension: A construction-integration model. *Psychological Review*, *95*(2), 163–182.

Kintsch, W. and van Dijk, T. A. (1978). Toward a model of text comprehension and production. *Psychological Review*, *85*, 363–394.

Kiraz, G. A. (1997). Compiling regular formalisms with rule features into finite-state automata. In *ACL/EACL-97*, Madrid, Spain, pp. 329–336. ACL.

Kisseberth, C. W. (1969). On the abstractness of phonology: The evidence from Yawelmani. *Papers in Linguistics*, *1*, 248–282.

Kisseberth, C. W. (1970). On the functional unity of phonological rules. *Linguistic Inquiry*, *1*(3), 291–306.

Kita, K., Fukui, Y., Nagata, M., and Morimoto, T. (1996). Automatic acquisition of probabilistic dialogue models. In *ICSLP-96*, Philadelphia, PA, Vol. 1, pp. 196–199.

Klatt, D. H. (1977). Review of the ARPA speech understanding project. *Journal of the Acoustical Society of America*, *62*(6), 1345–1366.

Klavans, J. (Ed.). (1995). *Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity and Generativity*. AAAI Press, Menlo Park, CA. AAAI Technical Report SS-95-01.

Kleene, S. C. (1951). Representation of events in nerve nets and finite automata. Tech. rep. RM-704, RAND Corporation. RAND Research Memorandum†.

Kleene, S. C. (1956). Representation of events in nerve nets and finite automata. In Shannon, C. and McCarthy, J. (Eds.), *Automata Studies*, pp. 3–41. Princeton University Press, Princeton, NJ.

Klein, S. and Simmons, R. F. (1963). A computational approach to grammatical coding

of English words. *Journal of the Association for Computing Machinery*, *10*(3), 334–347.

Klovstad, J. W. and Mondshein, L. F. (1975). The CASPERS linguistic analysis system. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *ASSP-23*(1), 118–123.

Kneser, R. (1996). Statistical language modeling using a variable context length. In *ICSLP-96*, Philadelphia, PA, Vol. 1, pp. 494–497.

Kneser, R. and Ney, H. (1993). Improved clustering techniques for class-based statistical language modelling. In *EUROSPEECH-93*, pp. 973–976.

Knight, K. (1989). Unification: A multidisciplinary survey. *ACM Computing Surveys*, *21*(1), 93–124.

Knight, K. (1997). Automating knowledge acquisition for machine translation. *AI Magazine*, *18*(4), 81–96.

Knight, K. and Al-Onaizan, Y. (1998). Translation with finite-state devices. In Farwell, D., Gerber, L., and Hovy, E. H. (Eds.), *Machine Translation and the Information Soup*, pp. 421–437. Springer.

Knight, K., Chander, I., Haines, M., Hatzivassiloglou, V., Hovy, E. H., Iida, M., Luk, S. K., Okumura, A., Whitney, R., and Yamada, K. (1994). Integrating knowledge bases and statistics in MT. In *Proceedings of the Conference of the Association for Machine Translation in the Americas*, Columbia, MD.

Knill, K. and Young, S. J. (1997). Hidden Markov Models in speech and language processing. In Young, S. J. and Bloothooft, G. (Eds.), *Corpus-based Methods in Language and Speech Processing*, pp. 27–68. Kluwer, Dordrecht.

Knott, A. and Dale, R. (1994). Using linguistic phenomena to motivate a set of coherence relations. *Discourse Processes*, *18*(1), 35–62.

Knuth, D. E. (1968). Semantics of context-free languages. *Mathematical Systems Theory*, *2*(2), 127–145.

Knuth, D. E. (1973). *Sorting and Searching: The Art of Computer Programming Volume 3*. Addison-Wesley, Reading, MA.

Koenig, J.-P. and Jurafsky, D. (1995). Type underspecification and on-line type construction in the lexicon. In Aranovich, R., Byrne, W., Preuss, S., and Senturia, M. (Eds.), *West Coast Conference on Formal Linguistics (WCCFL13)*, pp. 270–285.

Koenig, W., Dunn, H. K., , and Lacy, L. Y. (1946). The sound spectrograph. *Journal of the Acoustical Society of America*, *18*, 19–49.

Koerner, E. F. K. and Asher, R. E. (Eds.). (1995). *Concise History of the Language Sciences*. Elsevier Science, Oxford.

Kogure, K. (1990). Strategic lazy incremental copy graph unification. In *COLING-90*, Helsinki, pp. 223–228.

Kompe, R., Kießling, A., Kuhn, T., Mast, M., Niemann, H., Nöth, E., Ott, K., and Batliner, A. (1993). Prosody takes over: A prosodically guided dialog system. In *EUROSPEECH-93*, Berlin, Vol. 3, pp. 2003–2006.

Kornai, A. (1991). *Formal Phonology*. Ph.D. thesis, Stanford University, Stanford, CA†.

Koskenniemi, K. (1983). Two-level morphology: A general computational model of word-form recognition and production. Tech. rep. Publication No. 11, Department of General Linguistics, University of Helsinki.

Koskenniemi, K. and Church, K. W. (1988). Complexity, two-level morphology, and Finnish. In *COLING-88*, Budapest, pp. 335–339.

Krieger, H.-U. and Nerbonne, J. (1993). Feature-based inheritance networks for computational lexicons. In Briscoe, T., de Paiva, V., and Copestake, A. (Eds.), *Inheritance, Defaults, and the Lexicon*, pp. 90–136. Cambridge University Press, Cambridge.

Krieger, H.-U. and Schäfer, U. (1994). TDL — a type description language for HPSG. Part 1: Overview. Tech. rep. RR-94-37, DFKI, Saarbrücken.

Krippendorf, K. (1980). *Content Analysis: An Introduction to its Methodology*. Sage Publications, Beverly Hills, CA.

Krovetz, R. (1993). Viewing morphology as an inference process. In *SIGIR-93*, pp. 191–202. ACM.

Krovetz, R. and Croft, W. B. (1992). Lexical ambiguity and information retrieval. *ACM Transactions on Information Systems*, *10*(2), 115–141.

Kruskal, J. B. (1983). An overview of sequence comparison. In Sankoff, D. and Kruskal, J. B. (Eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, pp. 1–44. Addison-Wesley, Reading, MA.

Kučera, H. and Francis, W. N. (1967). *Computational analysis of present-day American English*. Brown University Press, Providence, RI.

Kuhn, R. and de Mori, R. (1990). A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(6), 570–583.

Kukich, K. (1988). Fluency in natural language reports. In McDonald, D. D. and Bolc, L. (Eds.), *Natural Language Generation Systems*, pp. 280–311. Springer-Verlag.

Kukich, K. (1992). Techniques for automatically correcting words in text. *ACM Computing Surveys*, *24*(4), 377–439.

Kuno, S. (1972). Functional sentence perspective: A case study from Japanese and English. *Linguistic Inquiry*, *3*(3), 269–320.

Kuno, S. (1965). The predictive analyzer and a path elimination technique. *Communications of the ACM*, *8*(7), 453–462.

Kuno, S. (1987). *Functional Syntax — Anaphora, Discourse and Empathy*. University of Chicago Press, Chicago.

Kuno, S. and Oettinger, A. G. (1963). Multiple-path syntactic analyzer. In Popplewell, C. M. (Ed.), *Information Processing 1962: Proceedings of the IFIP Congress 1962*, Munich, pp. 306–312. North-Holland. Reprinted in Grosz et al. (1986).

Kupiec, J. (1992). Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6, 225–242.

Kučera, H. (1992). The mathematics of language. In *The American Heritage Dictionary of the English Language*, pp. xxxi–xxxiii. Houghton Mifflin, Boston.

Labov, W. (1966). *The Social Stratification of English in New York City*. Center for Applied Linguistics, Washington, D.C.

Labov, W. (1969). Contraction, deletion, and inherent variability of the English copula. *Language*, *45*(4), 715–762.

Labov, W. (1972). The internal evolution of linguistic rules. In Stockwell, R. P. and Macaulay, R. K. S. (Eds.), *Linguistic Change and Generative Theory*, pp. 101–171. Indiana University Press, Bloomington.

Labov, W. (1975). *The quantitative study of linguistic structure*. Pennsylvania Working Papers on Linguistic Change and Variation v.1 no. 3. U.S. Regional Survey, Philadelphia, PA.

Labov, W. (1994). *Principles of linguistic change: internal factors*. Basil Blackwell, Oxford.

Labov, W. and Fanshel, D. (1977). *Therapeutic Discourse*. Academic Press, New York.

Ladd, D. R. (1996). *Intonational Phonology*. Cambridge Studies in Linguistics. Cambridge University Press.

Ladefoged, P. (1993). *A Course in Phonetics*. Harcourt Brace Jovanovich, Inc. Third Edition.

Ladefoged, P. (1996). *Elements of Acoustic Phonetics*. University of Chicago, Chicago, IL. Second Edition.

Lafferty, J. D., Sleator, D., and Temperley, D. (1992). Grammatical trigrams: A probabilistic model of link grammar. In *Proceedings of the 1992 AAAI Fall Symposium on Probabilistic Approaches to Natural Language*.

Lakoff, G. (1965). *On the Nature of Syntactic Irregularity*. Ph.D. thesis, Indiana University. Published as *Irregularity in Syntax*. Holt, Rinehart, and Winston, New York, 1970.

Lakoff, G. (1972). Linguistics and natural logic. In Davidson, D. and Harman, G. (Eds.), *Semantics for Natural Language*, pp. 545–665. D. Reidel, Dordrecht.

Lakoff, G. (1987). *Women, Fire, and Dangerous Things*. University of Chicago Press, Chicago.

Lakoff, G. (1993). Cognitive phonology. In Goldsmith, J. (Ed.), *The Last Phonological Rule*, pp. 117–145. University of Chicago Press, Chicago.

Lakoff, G. and Johnson, M. (1980). *Metaphors We Live By*. University of Chicago Press, Chicago, IL.

Lambek, J. (1958). The mathematics of sentence structure. *American Mathematical Monthly*, *65*(3), 154–170.

Lambrecht, K. (1994). *Information Structure and Sentence Form*. Cambridge University Press, Cambridge.

Landauer, T. K. (Ed.). (1995). *The Trouble With Computers: Usefulness, Usability, and Productivity*. MIT Press, Cambridge, MA.

Landauer, T. K. and Dumais, S. T. (1997). A solution to Plato's problem: The Latent Semantic Analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, *104*, 211–240.

Landauer, T. K., Laham, D., Rehder, B., and Schreiner, M. E. (1997). How well can passage meaning be derived without using word order: A comparison of latent semantic analysis and humans. In *COGSCI-97*, Stanford, CA, pp. 412–417. Lawrence Erlbaum.

Landes, S., Leacock, C., and Tengi, R. I. (1998). Building semantic concordances. In Fellbaum, C. (Ed.), *WordNet: An Electronic Lexical Database*, pp. 199–216. MIT Press, Cambridge, MA.

Langendoen, D. T. (1975). Finite-state parsing of phrase-structure languages and the status of readjustment rules in the grammar. *Linguistic Inquiry*, *6*(4), 533–554.

Langkilde, I. and Knight, K. (1998). The practical value of n-grams in generation. In *Proceedings of the Ninth International Workshop on Natural Language Generation*, Ontario, Canada, pp. 248–255.

Lapata, M. and Brew, C. (1999). Using subcategorization to resolve verb class ambiguity. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, College Park, MD, pp. 266–274.

Lappin, S. and Leass, H. (1994). An algorithm for pronominal anaphora resolution. *Computational Linguistics*, *20*(4), 535–561.

Lari, K. and Young, S. J. (1991). Applications of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech and Language*, *5*, 237–257.

Lascarides, A. and Asher, N. (1993). Temporal interpretation, discourse relations, and common sense entailment. *Linguistics and Philosophy*, *16*(5), 437–493.

Lascarides, A. and Copestake, A. (1997). Default representation in constraint-based frameworks. *Computational Linguistics*, *25*(1), 55–106.

Lauer, M. (1995). Corpus statistics meet the noun compound. In *ACL-95*, Cambridge, MA, pp. 47–54.

Lavoie, B. and Rambow, O. (1997). A fast and portable realizer for text generation systems. In *Proceedings of the Fifth Conference on Applied Natural Language Processing, Washington DC*, pp. 265–268.

Lavoie, B., Rambow, O., and Reiter, E. (1997). Customizable descriptions of object-oriented models. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, Washington, DC, pp. 265–268.

LDC (1993). *LDC Catalog: CSR-I (WSJ0) Complete*. University of Pennsylvania. `www.ldc.upenn.edu/Catalog/LDC93S6A.html`.

LDC (1995). COMLEX English Pronunciation Dictionary Version 0.2 (COMLEX 0.2). Linguistic Data Consortium.

LDC (1998). *LDC Catalog: Hub4 project*. University of Pennsylvania. `www.ldc.upenn.edu/Catalog/LDC98S71.html` or `www.ldc.upenn.edu/Catalog/Hub4.html`.

LDC (1999). *LDC Catalog: Hub5-LVCSR project*. University of Pennsylvania. `www.ldc.upenn.edu/ldc/about/chenglish.html` or `www.ldc.upenn.edu/Catalog/Hub5-LVCSR.html`.

Leech, G., Garside, R., and Bryant, M. (1994). Claws4: The tagging of the British National Corpus. In *COLING-94*, Kyoto, pp. 622–628.

Lees, R. (1970). Problems in the grammatical analysis of English nominal compounds. In Bierwitsch, M. and Heidolph, K. E. (Eds.), *Progress in Linguistics*, pp. 174–187. Mouton, The Hague.

Lehnert, W. (1977). A conceptual theory of question answering. In *IJCAI-77*, Cambridge, MA, pp. 158–164. Morgan Kaufmann.

Lehnert, W. G., Cardie, C., Fisher, D., Riloff, E., and Williams, R. (1991). Description of the CIRCUS system as used for MUC-3. In Sundheim, B. (Ed.), *Proceedings of the Third Message Understanding Conference*, pp. 223–233. Morgan Kaufmann.

Lehrer, A. (1974). *Semantic Fields and Lexical Structure*. North-Holland, Amsterdam.

Lehrer, A. and Kittay, E. (Eds.). (1992). *Frames, Fields and Contrasts: New Essays in Semantic and Lexical Organization*. Lawrence Erlbaum, Hillsdale, NJ.

Lenat, D. B. and Guha, R. V. (1991). *Building Large Knowledge-Based Systems: Representation and Inference in CYC*. Addison-Wesley, Reading, MA.

Lerner, A. J. (1978). *The Street Where I Live*. Da Capo Press, New York.

Lesk, M. E. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the Fifth International Conference on Systems Documentation*, Toronto, CA, pp. 24–26. ACM.

Lester, J. and Porter, B. (1997). Developing and empirically evaluating robust explanation generators: The KNIGHT experiments. *Computational Linguistics*, *23*(2), 65–101.

Levelt, W. J. M., Roelofs, A., and Meyer, A. S. (1999). A theory of lexical access in speech production. *Behavioral and Brain Science*, *22*, 1–75.

Levelt, W. J. M. (1970). A scaling approach to the study of syntactic relations. In d'Arcais, G. B. F. and Levelt, W. J. M. (Eds.), *Advances in psycholinguistics*, pp. 109–121. North-Holland, Amsterdam.

Levelt, W. J. M. (1983). Monitoring and self-repair in speech. *Cognition*, *14*, 41–104.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Cybernetics and Control Theory*, *10*(8), 707–710. Original in *Doklady Akademii Nauk SSSR* 163(4): 845–848 (1965).

Levesque, H. J., Cohen, P. R., and Nunes, J. H. T. (1990). On acting together. In *AAAI-90*, Boston, MA, pp. 94–99. Morgan Kaufmann.

Levi, J. (1978). *The Syntax and Semantics of Complex Nominals*. Academic Press, New York.

Levin, B. (1993). *English Verb Classes and Alternations*. University of Chicago Press, Chicago.

Levin, L., Gates, D., Lavie, A., and Waibel, A. (1998). An interlingua based on domain actions for machine translation of task-oriented dialogues. In *ICSLP-98*, Sydney, pp. 1155–1158.

Levinson, S. C. (1983). *Pragmatics*. Cambridge University Press, Cambridge.

Levinson, S. E. (1995). Structural methods in automatic speech recognition. *Proceedings of the IEEE*, *73*(11), 1625–1650.

Lewis, D. (1972). General semantics. In Davidson, D. and Harman, G. (Eds.), *Natural Language Semantics*, pp. 169–218. D. Reidel, Dordrecht.

Lewis, D. D. and Hayes, P. J. (Eds.). (1994). *ACM Transactions on Information Systems: Special Issue on Text Categorization*, Vol. 12. ACM Press.

Lewis, H. and Papadimitriou, C. (1981). *Elements of the Theory of Computation*. Prentice-Hall, Englewood Cliffs, NJ.

Liberman, M. and Church, K. W. (1992). Text analysis and word pronunciation in text-to-speech synthesis. In Furui, S. and Sondhi, M. M. (Eds.), *Advances in Speech Signal Processing*, pp. 791–832. Marcel Dekker, New York.

Liberman, M. and Prince, A. (1977). On stress and linguistic rhythm. *Linguistic Inquiry*, *8*, 249–336.

Liberman, M. and Sproat, R. (1992). The stress and structure of modified noun phrases in English. In Sag, I. A. and Szabolcsi, A. (Eds.), *Lexical Matters*, pp. 131–181. CSLI, Stanford University.

Lin, D. (1995). A dependency-based method for evaluating broad-coverage parsers. In *IJCAI-95*, Montreal, pp. 1420–1425.

Lindsey, R. (1963). Inferential memory as the basis of machines which understand natural language. In Feigenbaum, E. and Feldman, J. (Eds.), *Computers and Thought*, pp. 217–233. McGraw Hill.

Litman, D. J. (1985). *Plan Recognition and Discourse Analysis: An Integrated Approach for Understanding Dialogues*. Ph.D. thesis, University of Rochester, Rochester, NY.

Litman, D. J. and Allen, J. F. (1987). A plan recognition model for subdialogues in conversation. *Cognitive Science*, *11*, 163–200.

Lochbaum, K. E. (1998). A collaborative planning model of intentional structure. *Computational Linguistics*, *24*(4), 525–572.

Lochbaum, K. E., Grosz, B. J., and Sidner, C. L. (1990). Models of plans to support communication: An initial report. In *AAAI-90*, Boston, MA, pp. 485–490. Morgan Kaufmann.

Longacre, R. E. (1983). *The Grammar of Discourse*. Plenum Press.

Lopresti, D. and Zhou, J. (1997). Using consensus sequence voting to correct OCR errors. *Computer Vision and Image Understanding*, *67*(1), 39–47.

Losiewicz, B. L. (1992). *The Effect of Frequency on Linguistic Morphology*. Ph.D. thesis, University of Texas, Austin, TX.

Lowe, J. B., Baker, C. F., and Fillmore, C. J. (1997). A frame-semantic approach to semantic annotation. In *Proceedings of ACL SIGLEX Workshop on Tagging Text with Lexical Semantics*, Washington, D.C., pp. 18–24. ACL.

Lowerre, B. T. (1968). *The Harpy Speech Recognition System*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.

Luce, P. A., Pisoni, D. B., and Goldfinger, S. D. (1990). Similarity neighborhoods of spoken words. In Altmann, G. T. M. (Ed.), *Cognitive Models of Speech Processing*, pp. 122–147. MIT Press, Cambridge, MA.

Luhn, H. P. (1957). A statistical approach to the mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, *1*(4), 309–317.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, *2*(2), 159–165.

Lyons, J. (1977). *Semantics*. Cambridge University Press, New York.

MacDonald, M. C. (1993). The interaction of lexical and syntactic ambiguity. *Journal of Memory and Language*, *32*, 692–715.

MacDonald, M. C. (1994). Probabilistic constraints and syntactic ambiguity resolution. *Language and Cognitive Processes*, *9*(2), 157–201.

Macleod, C., Grishman, R., and Meyers, A. (1998). COMLEX Syntax Reference Manual Version 3.0. Linguistic Data Consortium.

MacWhinney, B. (1977). Starting points. *Language*, *53*, 152–168.

MacWhinney, B. (1982). Basic syntactic processes. In Kuczaj, S. (Ed.), *Language Acquisition: Volume 1, Syntax and Semantics*, pp. 73–136. Lawrence Erlbaum, Hillsdale, NJ.

MacWhinney, B. (1987). The competition model. In MacWhinney, B. (Ed.), *Mechanisms of Language Acquisition*, pp. 249–308. Lawrence Erlbaum, Hillsdale, NJ.

MacWhinney, B. and Csaba Pléh (1988). The processing of restrictive relative clauses in Hungarian. *Cognition*, *29*, 95–141.

MacWhinney, B. and Leinbach, J. (1991). Implementations are not conceptualizations: Revising the verb learning model. *Cognition*, *40*, 121–157.

Madhu, S. and Lytel, D. (1965). A figure of merit technique for the resolution of non-grammatical ambiguity. *Mechanical Translation*, *8*(2), 9–13.

Magerman, D. M. (1995). Statistical decision-tree models for parsing. In *ACL-95*, Cambridge, MA, pp. 276–283. ACL.

Magerman, D. M. and Marcus, M. P. (1991). Pearl: A probabilistic chart parser. In *Proceedings of the 6th Conference of the European Chapter of the ACL*, Berlin, Germany.

Main, M. G. and Benson, D. B. (1983). Denotational semantics for natural language question-answering programs. *American Journal of Computational Linguistics*, *9*(1), 11–21.

Makkai, A. (1972). *Idiom Structure in English*. Mouton, The Hague.

Mangu, L. and Brill, E. (1997). Automatic rule acquisition for spelling correction. In *Proceedings of the 14th International Conference on Machine Learning (ICML-97)*, Nashville, TN, pp. 187–194. Morgan Kaufmann.

Mann, W. C. and Moore, J. D. (1981). Computer generation of multiparagraph text. *Computational Linguistics*, *7*(1), 17–29.

Mann, W. C. (1983). An overview of the PENMAN text generation system. In *Proceedings of the National Conference on Artificial Intelligence*, pp. 261–265.

Mann, W. C. and Thompson, S. A. (1986). Relational propositions in discourse. *Discourse Processes*, *9*(1), 57–90.

Mann, W. C. and Thompson, S. A. (1987). Rhetorical structure theory: A theory of text organization. Tech. rep. RS-87-190, Information Sciences Institute.

Manning, C. D. (1993). Automatic acquisition of a large subcategorization dictionary from corpora. In *ACL-93*, Columbus, Ohio, pp. 235–242. ACL.

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

Marcu, D. (1998). Improving summarization through rhetorical parsing tuning. In *Proceedings of the Sixth Workshop on Very Large Corpora (WVLC–6)*, Montreal, Canada, pp. 206–215.

Marcus, G. F., Brinkman, U., Clahsen, H., Wiese, R., and Pinker, S. (1995). German inflection: The exception that proves the rule. *Cognitive Psychology*, *29*, 189–256.

Marcus, M. P. (1990). Summary of session 9: Automatic acquisition of linguistic structure. In *Proceedings DARPA Speech and Natural Language Workshop*, Hidden Valley, PA, pp. 249–250. Morgan Kaufmann.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, *19*(2), 313–330.

Markey, K. and Ward, W. (1997). Lexical tuning based on triphone confidence estimation. In *EUROSPEECH-97*, Vol. 5, pp. 2479–2482.

Markov, A. A. (1913). Essai d'une recherche statistique sur le texte du roman "Eugene Onegin" illustrant la liaison des epreuve en chain ('Example of a statistical investigation of the text of "Eugene Onegin" illustrating the dependence between samples in chain'). *Izvistia Imperatorskoi Akademii Nauk (Bulletin de l'Académie Impériale des Sciences de St.-Pétersbourg)*, *7*, 153–162. English translation by Morris Halle, 1956.

Marshall, I. (1983). Choice of grammatical word-class without global syntactic analysis: Tagging words in the LOB corpus. *Computers and the Humanities*, *17*, 139–150.

Marshall, I. (1987). Tag selection using probabilistic methods. In Garside, R., Leech, G., and Sampson, G. (Eds.), *The Computational Analysis of English*, pp. 42–56. Longman, London.

Marslen-Wilson, W. and Welsh, A. (1978). Processing interactions and lexical access during word recognition in continuous speech. *Cognitive Psychology*, *10*, 29–63.

Marslen-Wilson, W., Tyler, L. K., Waksler, R., and Older, L. (1994). Morphology and meaning in the English mental lexicon. *Psychological Review*, *101*(1), 3–33.

Marslen-Wilson, W. D. (1973). Linguistic structure and speech shadowing at very short latencies. *Nature*, *244*, 522–523.

Martin, J. H. (1986). The acquisition of polysemy. In *Proceedings of the Fourth International Conference on Machine Learning*, Irvine, CA, pp. 198–204.

Martin, J. H. (1990). *A Computational Model of Metaphor Interpretation*. Perspectives in Artificial Intelligence. Academic Press, San Diego, CA.

Martin, J. H. (1996). Computational approaches to figurative language. *Metaphor and Symbolic Activity, 11*(1), 85–100.

Massaro, D. W. (1998). *Perceiving Talking Faces: From Speech Perception to a Behavioral Principle*. MIT Press.

Massaro, D. W. and Cohen, M. M. (1983). Evaluation and integration of visual and auditory information in speech perception. *Journal of Experimental Psychology: Human Perception and Performance, 9*, 753–771.

Mast, M., Kompe, R., Harbeck, S., Kießling, A., Niemann, H., Nöth, E., Schukat-Talamazzini, E. G., and Warnke, V. (1996). Dialog act classification with the help of prosody. In *ICSLP-96*, Philadelphia, PA, Vol. 3, pp. 1732–1735.

Masterman, M. (1957). The thesaurus in syntax and semantics. *Mechanical Translation, 4*(1), 1–2.

Mather, L. (1998). *Enhancing Cluster-Based Retrieval through Linear Algebra*. Ph.D. thesis, University of Colorado, Boulder, CO.

Matthews, A. and Chodorow, M. S. (1988). Pronoun resolution in two-clause sentences: Effects of ambiguity, antecedent location, and depth of embedding. *Journal of Memory and Language, 27*, 245–260.

Mays, E., Damerau, F. J., and Mercer, R. L. (1991). Context based spelling correction. *Information Processing and Management, 27*(5), 517–522.

Mazuka, R. and Itoh, K. (1995). Can Japanese speakers be led down the garden path?. In Mazuka, R. and Nagai, N. (Eds.), *Japanese Sentence Processing*, pp. 295–330. Lawrence Erlbaum, Hillsdale, NJ.

McCarthy, J. J. (1981). A prosodic theory of non-concatenative morphology. *Linguistic Inquiry, 12*, 373–418.

McCarthy, J. F. and Lehnert, W. G. (1995). Using decision trees for coreference resolution. In *IJCAI-95*, Montreal, Canada, pp. 1050–1055.

McCawley, J. D. (1968). The role of semantics in a grammar. In Bach, E. W. and Harms, R. T. (Eds.), *Universals in Linguistic Theory*, pp. 124–169. Holt, Rinehart & Winston, New York, NY.

McCawley, J. D. (1978). Where you can shove infixes. In Bell, A. and Hooper, J. B. (Eds.), *Syllables and Segments*, pp. 213–221. North-Holland, Amsterdam.

McCawley, J. D. (1993). *Everything that Linguists have Always Wanted to Know about Logic* (2nd edition). University of Chicago Press, Chicago, IL.

McCawley, J. D. (1998). *The Syntactic Phenomena of English*. University of Chicago Press, Chicago.

McClelland, J. L. and Elman, J. L. (1986). Interactive processes in speech perception: The TRACE model. In McClelland, J. L., Rumelhart, D. E., and the PDP Research Group (Eds.), *Parallel Distributed Processing Volume 2: Psychological and Biological Models*, pp. 58–121. MIT Press, Cambridge, MA.

McCoy, K. F. (1985). *Correcting Object-Related Misconceptions*. Ph.D. thesis, University of Pennsylvania.

McCoy, K. F., Pennington, C. A., and Badman, A. L. (1998). Compansion: From research prototype to practical integration. *Natural Language Engineering, 4*(1), 73–95.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, pp. 115–133. Reprinted in *Neurocomputing: Foundations of Research, ed. by J. A. Anderson and E Rosenfeld. MIT Press 1988.*

McDermott, D. (1976). Artificial intelligence meets natural stupidity. *SIGART Newsletter, 57*.

McDonald, D. B. (1982). *Understanding Noun Compounds*. Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA. CMU Technical Report CS-82-102.

McDonald, D. D. (1980). *Natural Language Production as a Process of Decision Making*. Ph.D. thesis, MIT, Cambridge, MA.

McDonald, D. D. (1988). Modularity in natural language generation: Methodological issues. In *Proceedings of the AAAI Workshop on Text Planning and Realization*, pp. 91–98.

McDonald, D. D. (1992). Natural-language generation. In Shapiro, S. C. (Ed.), *Encyclopedia of Artificial Intelligence* (2nd edition)., pp. 642–655. John Wiley, New York.

McKeown, K. R. (1985). *Text Generation*. Cambridge University Press, Cambridge.

McKeown, K. R., Elhadad, M., Fukumoto, Y., Lim, J., Lombardi, C., Robin, J., and Smadja, F. (1990). Natural language generation in COMET. In Dale, R., Mellish, C., and Zock, M. (Eds.), *Current Research in Natural Language Generation*, chap. 5, pp. 103–139. Academic Press, London.

McKeown, K. R. and Swartout, W. R. (1988). Language generation and explanation. In Zock, M. and Sabah, G. (Eds.), *Advances in Natural Language Generation — An Interdisciplinary Perspective*, Vol. 1, chap. 1, pp. 1–51. Ablex, Norwood, NJ.

McKoon, G. and Ratcliff, R. (1992). Inferences during reading. *Psychological Review*, *99*, 440–466.

McRoy, S. (1992). Using multiple knowledge sources for word sense discrimination. *Computational Linguistics*, *18*(1), 1–30.

McTear, M. (1998). Modelling spoken dialogues with state transition diagrams: Experiences with the CSLU toolkit. In *ICSLP-98*, Sydney, Vol. 4, pp. 1223–1226.

Mealy, G. H. (1955). A method for synthesizing sequential circuits. *Bell System Technical Journal*, *34*(5), 1045–1079.

Melamed, I. D. (1999). Bitext maps and alignment via pattern recognition. *Computational Linguistics*, *25*(1), 107–130.

Melamed, I. D. (1999 to appear). Word-to-word models of translational equivalence. *Computational Linguistics*.

Mel'čuk, I. A. (1979). *Studies in dependency syntax*. Karoma Publishers, Ann Arbor.

Mel'čuk, I. A. (1988). *Dependency Syntax: Theory and Practice*. SUNY Series in Linguistics, Mark Aronoff, series editor. State University of New York Press, Albany.

Merialdo, B. (1994). Tagging English text with a probabilistic model. *Computational Linguistics*, *20*(2), 155–172.

Merton, R. K. (1961). Singletons and multiples in scientific discovery. *American Philosophical Society Proceedings*, *105*(5), 470–486.

Meteer, M. and Iyer, R. (1996). Modeling conversational speech for speech recognition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, University of Pennsylvania, pp. 33–47. ACL.

Meteer, M. W. (1992). *Expressibility and the Problem of Efficient Text Planning*. Pinter, London.

Meurers, W. D. and Minnen, G. (1997). A computational treatment of lexical rules in HPSG as covariation in lexical entries. *Computational Linguistics*, *23*(4), 543–568.

Miller, C. A. (1998). Pronunciation modeling in speech synthesis. Tech. rep. IRCS 98–09, University of Pennsylvania Institute for Research in Cognitive Science, Philadephia, PA.

Miller, G. A. and Chomsky, N. (1963). Finitary models of language users. In Luce, R. D., Bush, R. R., and Galanter, E. (Eds.), *Handbook of Mathematical Psychology*, Vol. II, pp. 419–491. John Wiley, New York.

Miller, G. A. and Selfridge, J. A. (1950). Verbal context and the recall of meaningful material. *American Journal of Psychology*, *63*, 176–185.

Miller, J. L. (1994). On the internal structure of phonetic categories: a progress report. *Cognition*, *50*, 271–275.

Milosavljevic, M. (1997). Content.selection in comparison generation. In *Proceedings of the 6th European Workshop on Natural Language Generation*, Duisburg, Germany, 24–26 March, pp. 72–81.

Mitamura, T. and Nyberg, E. H. (1995). Controlled English for knowledge-based MT: Experience with the KANT system. In *6th International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium.

Mitchell, D. C., Cuetos, F., Corley, M. M. B., and Brysbaert, M. (1995). Exposure-based models of human parsing: Evidence for the use of coarse-grained (nonlexical) statistical records. *Journal of Psycholinguistic Research*, *24*(6), 469–488.

Mitchell, T. M. (1981). Generalization as search. In Webber, B. L. and Nilsson, N. J. (Eds.), *Readings in Artificial Intelligence*, pp. 517–542. Morgan Kaufmann, Los Altos.

Mitkov, R. and Boguraev, B. (Eds.). (1997). *Proceedings of the ACL-97 Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, Madrid, Spain. ACL.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, *23*(2), 269–312.

Montague, R. (1973). The proper treatment of quantification in ordinary English. In Thomason, R. (Ed.), *Formal Philosophy: Selected Papers of Richard Montague*, pp. 247–270. Yale University Press, New Haven, CT.

Mooney, R. J. (1995). Encouraging experimental results on learning CNF. *Machine Learning*, *19*(1), 79–92.

Mooney, R. J. (1996). Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-96)*, Philadelphia, PA, pp. 82–91.

Moore, E. F. (1956). Gedanken-experiments on sequential machines. In Shannon, C. and McCarthy, J. (Eds.), *Automata Studies*, pp. 129–153. Princeton University Press, Princeton, NJ.

Moore, J. D. and Paris, C. L. (1993). Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Computational Linguistics*, *19*(4), 651–694.

Moore, J. D. and Pollack, M. E. (1992). A problem for RST: The need for multi-level discourse analysis. *Computational Linguistics*, *18*(4), 537–544.

Moore, R. (1977). Reasoning about knowledge and action. In *IJCAI-77*, pp. 223–227.

Moore, R., Appelt, D., Dowding, J., Gawron, J. M., and Moran, D. (1995). Combining linguistic and statistical knowledge sources in natural-language processing for ATIS. In *Proceedings of the January 1995 ARPA Spoken Language Systems Technology Workshop*, Austin, TX, pp. 261–264. Morgan Kaufmann.

Morris, J. and Hirst, G. (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, *17*(1), 21–48.

Morris, W. (Ed.). (1985). *American Heritage Dictionary* (2nd College Edition edition). Houghton Mifflin.

Moshier, M. A. (1988). *Extensions to Unification Grammar for the Description of Programming Languages*. Ph.D. thesis, University of Michigan, Ann Arbor, MI.

Mosteller, F. and Wallace, D. L. (1964). Inference and Disputed Authorship: The Federalist. Springer-Verlag, New York. 2nd Edition appeared in 1984 and was called *Applied Bayesian and Classical Inference*.

Mostow, J. and Aist, G. (1999). Reading and pronunciation tutor.. U.S. Patent 5,920,838.

Munoz, M., Punyakanok, V., Roth, D., and Zimak, D. (1999). A learning approach to shallow parsing. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, College Park, MD, pp. 168–178. ACL.

Murata, T. (1989). Petri nets: Properties, analysis, and applications. *Proceedings of the IEEE*, *77*(4), 541–576.

Murveit, H., Butzberger, J. W., Digalakis, V. V., and Weintraub, M. (1993). Large-vocabulary dictation using SRI's decipher speech recognition system: Progressive-search techniques. In *IEEE ICASSP-93*, Vol. 2, pp. 319–322. IEEE.

Myers, J. L., Shinjo, M., and Duffy, S. A. (1987). Degree of causal relatedness and memory. *Journal of Verbal Learning and Verbal Behavior*, *26*, 453–465.

Nádas, A. (1984). Estimation of probabilities in the language model of the ibm speech recognition system. *IEEE Transactions on Acoustics, Speech, Signal Processing*, *32*(4), 859–861.

Nagata, M. and Morimoto, T. (1994). First steps toward statistical modeling of dialogue to predict the speech act type of the next utterance. *Speech Communication*, *15*, 193–203.

Narayanan, S. (1997a). *Knowledge-based Action Representations for Metaphor and Aspect (KARMA)*. Ph.D. thesis, University of California, Berkeley.

Narayanan, S. (1997b). Talking the talk *is* like walking the walk: A computational model of verbal aspect. In *COGSCI-97*, Stanford, CA, pp. 548–553.

Narayanan, S. and Jurafsky, D. (1998). Bayesian models of human sentence processing. In *COGSCI-98*, Madison, WI, pp. 752–757. LEA.

Naur, P., Backus, J. W., Bauer, F. L., Green, J., Katz, C., McCarthy, J., Perlis, A. J., Rutishauser, H., Samelson, K., Vauquois, B., Wegstein, J. H., van Wijnagaarden, A., and Woodger, M. (1960). Report on the algorithmic language ALGOL 60. *Communications of the ACM*, *3*(5), 299–314. Revised in CACM 6:1, 1-17, 1963.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino-acid sequence of two proteins. *Journal of Molecular Biology*, *48*, 443–453.

Nespor, M. and Vogel, I. (1986). *Prosodic phonology*. Foris, Dordrecht.

Neu, H. (1980). Ranking of constraints on /t,d/ deletion in American English: A statistical analysis. In Labov, W. (Ed.), *Locating Language in Time and Space*, pp. 37–54. Academic, New York.

Newell, A., Langer, S., and Hickey, M. (1998). The rôle of natural language processing in alternative and augmentative communication. *Natural Language Engineering*, *4*(1), 1–16.

Newman, S. (1944). *Yokuts Language of California*. Viking Fund Publications in Anthropology 2, New York.

Ney, H., Essen, U., and Kneser, R. (1994). On structuring probabilistic dependencies in stochastic language modelling. *Computer Speech and Language*, *8*, 1–38.

Ney, H., Haeb-Umbach, R., Tran, B.-H., and Oerder, M. (1992). Improvements in beam search for 10000-word continuous speech recognition. In *IEEE ICASSP-92*, San Francisco, CA, pp. I.9–12. IEEE.

Ney, H. (1991). Dynamic programming parsing for context-free grammars in continuous speech recognition. *IEEE Transactions on Signal Processing*, *39*(2), 336–340.

Ng, H. T. and Lee, H. B. (1996). Integrating multiple knowledge sources to disambiguate word senses: An exemplar-based approach. In *ACL-96*, Santa Cruz, CA, pp. 40–47. ACL.

Ng, H. T. and Zelle, J. (1997). Corpus-based approaches to semantic interpretation in NLP. *AI Magazine*, *18*(4), 45–64.

Nguyen, L. and Schwartz, R. (1999). Single-tree method for grammar-directed search. In *IEEE ICASSP-99*, pp. 613–616. IEEE.

Nichols, J. (1986). Head-marking and dependent-marking grammar. *Language*, *62*(1), 56–119.

Nida, E. A. (1975). *Componential Analysis of Meaning: An Introduction to Semantic Structures*. Mouton, The Hague.

Nielsen, J. (1992). The usability engineering life cycle. *IEEE Computer*, 12–22.

Niesler, T. R. and Woodland, P. C. (1996). A variable-length category-based n-gram language model. In *IEEE ICASSP-96*, Atlanta, GA, Vol. I, pp. 164–167. IEEE.

Niesler, T. R. and Woodland, P. C. (1999). Modelling word-pair relations in a category-based language model. In *IEEE ICASSP-99*, pp. 795–798. IEEE.

Nilsson, N. J. (1980). *Principles of Artificial Intelligence*. Morgan Kaufmann, Los Altos, CA.

Nirenburg, S., Lesser, V., and Nyberg, E. H. (1989). Controlling a language generation planner. In *IJCAI-89*, pp. 1524–1530.

Nirenburg, S., Carbonell, J., Tomita, M., and Goodman, K. (1992). *Machine Transla-*

tion: A Knowledge-based Approach. Morgan Kaufmann.

Norman, D. A. and Rumelhart, D. E. (1975). *Explorations in Cognition*. Freeman, San Francisco, CA.

Norvig, P. (1987). *A Unified Theory of Inference for Text Understanding*. Ph.D. thesis, University of California, Berkeley, CA. Available as University of California at Berkeley Computer Science Division Tech. rep. #87/339.

Norvig, P. (1991). Techniques for automatic memoization with applications to context-free parsing. *Computational Linguistics*, *17*(1), 91–98.

Oard, D. W. (1997). Alternative approaches for cross-language text retrieval. In *AAAI Spring Symposium on Cross-Language Text and Speech Retrieval*.

Odell, M. K. and Russell, R. C. (1918/1922). U.S. Patents 1261167 (1918), 1435663 (1922)†. Cited in Knuth (1973).

Oden, G. C. and Massaro, D. W. (1978). Integration of featural information in speech perception. *Psychological Review*, *85*, 172–191.

O'Donnell, M. J. (1994). *Sentence Analysis and generation: A Systemic Perspective*. Ph.D. thesis, University of Sydney.

Oehrle, R. T., Bach, E., and Wheeler, D. (Eds.). (1988). *Categorial Grammars and Natural Language Structures*. D. Reidel, Dordrecht.

Oflazer, K. (1993). Two-level description of Turkish morphology. In *Proceedings, Sixth Conference of the European Chapter of the ACL*.

Ogburn, W. F. and Thomas, D. S. (1922). Are inventions inevitable? a note on social evolution. *Political Science Quarterly*, *37*, 83–98.

Oncina, J., García, P., and Vidal, E. (1993). Learning subsequential transducers for pattern recognition tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*, 448–458.

Oppenheim, A., Schafer, R., and Stockham, T. J. (1968). Nonlinear filtering of multiplied and convolved signals. *Proceedings of the IEEE*, *56*(8), 1264–1291.

Orgun, O. (1995). A declarative theory of phonology-morphology interleaving. Unpublished manuscript, U. of California-Berkeley, Department of Linguistics.

Ortony, A. (Ed.). (1993). *Metaphor* (2nd edition). Cambridge University Press, Cambridge.

Ostendorf, M. and Veilleux, N. (1994). A hierarchical stochastic model for automatic prediction of prosodic boundary location. *Computational Linguistics*, *20*(1).

Oviatt, S., Cohen, P. R., Wang, M. Q., and Gaston, J. (1993). A simulation-based research strategy for designing complex NL sysems. In *Proceedings DARPA Speech and Natural Language Workshop*, Princeton, NJ, pp. 370–375. Morgan Kaufmann.

Oviatt, S., MacEachern, M., and Levow, G.-A. (1998). Predicting hyperarticulate speech during human-computer error resolution. *Speech Communication*, *24*, 87–110.

Packard, D. W. (1973). Computer-assisted morphological analysis of ancient Greek. In Zampolli, A. and Calzolari, N. (Eds.), *Computational and Mathematical Linguistics: Proceedings of the International Conference on Computational Linguistics*, Pisa, pp. 343–355. Leo S. Olschki.

Palmer, M. and Finin, T. (1990). Workshop on the evaluation of natural language processing systems. *Computational Linguistics*, *16*(3), 175–181.

Paris, C. (1993). *User Modelling in Text Generation*. Pinter, London.

Paris, C. et al. (1995). A support tool for writing multilingual instructions. In *IJCAI-95*, pp. 1398–1404.

Paris, C. and Vander Linden, K. (1996). Drafter: An interactive support tool for writing multilingual instructions. *IEEE Computer*, *29*(7), 49–56.

Paris, C., Vander Linden, K., and Lu, S. (1998). Automatic document creation from software specifications. In Kay, J. and Milosavljevic, M. (Eds.), *Proceedings of the 3rd Australian Document Computing Symposium (ADCS-98), Sydney, August*, pp. 26–31.

Parsons, T. (1990). *Events in the Semantics of English*. MIT Press, Cambridge, MA.

Partee, B. H. (Ed.). (1976). *Montague Grammar*. Academic Press, New York.

Partee, B. H., ter Meulen, A., and Wall, R. E. (1990). *Mathematical Methods in Linguistics*. Kluwer, Dordrecht.

Passonneau, R. and Litman, D. J. (1993). Intention-based segmentation: Human reliability and correlation with linguistic cues. In *ACL-93*, Columbus, Ohio, pp. 148–155. ACL.

Patten, T. (1988). *Systemic Text Generation as Problem Solving*. Cambridge University Press.

Paul, D. B. (1991). Algorithms for an optimal A* search and linearizing the search in the stack decoder. In *IEEE ICASSP-91*, Vol. 1, pp. 693–696. IEEE.

Pearl, J. (1984). *Heuristics*. Addison-Wesley, Reading, MA.

Pearlmutter, N. J. and MacDonald, M. C. (1992). Plausibility and syntactic ambiguity resolution. In *COGSCI-92*, pp. 498–503.

Pedersen, T. and Bruce, R. (1997). Distinguishing word senses in untagged text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-97)*, Providence, RI.

Percival, W. K. (1976). On the historical source of immediate constituent analysis. In McCawley, J. D. (Ed.), *Syntax and Semantics Volume 7, Notes from the Linguistic Underground*, pp. 229–242. Academic Press, New York.

Pereira, F. (1985). A structure-sharing representation for unification-based grammar formalisms. In *ACL-85*, Chicago, pp. 137–144.

Pereira, F., Riley, M. D., and Sproat, R. (1994). Weighted rational transductions and their applications to human language processing. In *ARPA Human Language Technology Workshop*, Plainsboro, NJ, pp. 262–267. Morgan Kaufmann.

Pereira, F. and Shieber, S. M. (1984). The semantics of grammar formalisms seen as computer languages. In *COLING-84*, Stanford, CA, pp. 123–129.

Pereira, F. and Shieber, S. M. (1987). *Prolog and Natural-Language Analysis*, Vol. 10 of *CSLI Lecture Notes*. Chicago University Press, Chicago.

Pereira, F. and Warren, D. H. D. (1980). Definite clause grammars for language analysis— a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, *13*(3), 231–278.

Pereira, F. and Wright, R. N. (1997). Finite-state approximation of phrase-structure grammars. In Roche, E. and Schabes, Y. (Eds.), *Finite-State Devices for Natural Language Processing*, pp. 149–174. MIT Press, Cambridge, MA.

Perrault, C. R. and Allen, J. (1980). A plan-based analysis of indirect speech acts. *American Journal of Computational Linguistics*, *6*(3-4), 167–182.

Peterson, J. L. (1986). A note on undetected typing errors. *Communications of the ACM*, *29*(7), 633–637.

Picone, J. (1993). Signal modeling techniques in speech recognition. *Proceedings of the IEEE*, *81*(9), 1215–1247.

Pierce, C. S. (1955). Abduction and induction. In Buchler, J. (Ed.), *Philosophical Writings of Pierce*, pp. 150–156. Dover Books, New York.

Pierre, I. (1984). Another look at nominal compounds. In *COLING-84*, Stanford, CA, pp. 509–516.

Pierrehumbert, J. and Hirschberg, J. (1990). The meaning of intonational contours in the interpretation of discourse. In Cohen, P. R., Morgan, J., and Pollack, M. (Eds.), *Intentions in Communication*, pp. 271–311. MIT Press, Cambridge, MA.

Pierrehumbert, J. (1980). *The Phonology and Phonetics of English Intonation*. Ph.D. thesis, MIT.

Pinker, S. (1989). *Learnability and Cognition: The acquisition of argument structure*. MIT Press, Cambridge.

Pinker, S. and Prince, A. (1988). On language and connectionism: Analysis of a Parallel Distributed Processing model of language acquisition. *Cognition*, *28*, 73–193.

Placeway, P., Schwartz, R., Fung, P., and Nguyen, L. (1993). The estimation of powerful language models from small and large corpora. In *IEEE ICASSP-93*, Vol. 2, pp. 33–36. IEEE.

Plunkett, K. and Marchman, V. (1991). U-shaped learning and frequency effects in a multi-layered perceptron: Implications for child language acquisition. *Cognition, 38*, 43–102.

Poesio, M. and Vieira, R. (1998). A corpus-based investigation of definite description use. *Computational Linguistics, 24*(2), 183–216.

Polanyi, L. (1988). A formal model of the structure of discourse. *Journal of Pragmatics, 12*.

Polifroni, J., Hirschman, L., Seneff, S., and Zue, V. (1992). Experiments in evaluating interactive spoken language systems. In *Proceedings DARPA Speech and Natural Language Workshop*, Harriman, New York, pp. 28–33. Morgan Kaufmann.

Pollard, C. and Moshier, M. A. (1990). Unifying partial descriptions of sets. In Hanson, P. P. (Ed.), *Information, Language, and Cognition*, pp. 285–322. University of British Columbia Press, Vancouver.

Pollard, C. and Sag, I. A. (1987). *Information-Based Syntax and Semantics: Volume 1: Fundamentals*. University of Chicago Press, Chicago.

Pollard, C. and Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–127.

Power, R. (1979). The organization of purposeful dialogs. *Linguistics, 17*, 105–152.

Price, P., Fisher, W., Bernstein, J., and Pallet, D. (1988). The DARPA 1000-word resource management database for continuous speech recognition. In *IEEE ICASSP-88*, New York, Vol. 1, pp. 651–654. IEEE.

Prince, A. and Smolensky, P. (1993). Optimality theory: Constraint interaction in generative grammar. Tech. rep. CU-CS-696-93, Department of Computer Science, University of Colorado at Boulder, and RuCCs Tech. rep. TR-2, Cognitive Science Center, Rutgers University. [to appear, MIT Press, Cambridge, MA].

Prince, E. (1981). Toward a taxonomy of given-new information. In Cole, P. (Ed.), *Radical Pragmatics*, pp. 223–255. Academic Press, New York, New York.

Prince, E. (1992). The ZPG letter: Subjects, definiteness, and information-status. In Thompson, S. and Mann, W. (Eds.), *Discourse Description: Diverse Analyses of a Fundraising Text*, pp. 295–325. John Benjamins, Philadelphia/Amsterdam.

Pritchett, B. (1988). Garden path phenomena and the grammatical basis of language processing. *Language, 64*(3), 539–576.

Procter, P. (Ed.). (1978). *Longman Dictionary of Contemporary English*. Longman Group, Essex, England.

Prüst, H. (1992). *On Discourse Structuring, VP Anaphora, and Gapping*. Ph.D. thesis, University of Amsterdam.

Pullum, G. K. and Gazdar, G. (1982). Natural languages and context-free languages. *Linguistics and Philosophy, 4*, 471–504.

Pullum, G. K. (1991). *The Great Eskimo Vocabulary Hoax*. University of Chicago, Chicago, IL.

Pullum, G. K. and Ladusaw, W. A. (1996). *Phonetic Symbol Guide*. University of Chicago, Chicago, IL. Second Edition.

Pustejovsky, J. (1995). *The Generative Lexicon*. MIT Press, Cambridge, MA.

Pustejovsky, J. and Bergler, S. (Eds.). (1992). *Lexical Semantics and Knowledge Representation*. Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin.

Quillian, M. R. (1968). Semantic memory. In Minsky, M. (Ed.), *Semantic Information Processing*, pp. 227–270. MIT Press, Cambridge, MA.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1*, 81–106.

Quirk, R., Greenbaum, S., Leech, G., and Svartvik, J. (1985). *A Comprehensive Grammar of the English Language*. Longman, London.

Rabin, M. O. and Scott, D. (1959). Finite automata and their decision problems.

IBM Journal of Research and Development, *3*(2), 114–125.

Rabiner, L. R. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286.

Rabiner, L. R. and Juang, B. (1993). *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ.

Radford, A. (1988). *Transformational Grammar: A First Course*. Cambridge University Press, Cambridge.

Ramshaw, L. A. and Marcus, M. P. (1995). Text chunking using transformation-based learning. In *Proceedings of the Third Annual Workshop on Very Large Corpora*, pp. 82–94. ACL.

Raphael, B. (1968). SIR: A computer program for semantic information retrieval. In Minsky, M. (Ed.), *Semantic Information Processing*, pp. 33–145. MIT Press.

Ratnaparkhi, A. (1996). A maximum entropy part-of-speech tagger. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, University of Pennsylvania, pp. 133–142. ACL.

Ratnaparkhi, A. (1997). A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, Providence, Rhode Island, pp. 1–10. ACL.

Ratnaparkhi, A., Reynar, J., and Roukos, S. (1994). A Maximum Entropy model for prepositional phrase attachment. In *ARPA Human Language Technologies Workshop*, Plainsboro, N.J., pp. 250–255.

Ravishankar, M. (1996). *Efficient Algorithms for Speech Recognition*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh. Available as CMU CS tech report CMU-CS-96-143.

Reeves, B. and Nass, C. (1996). *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places*. Cambridge University Press, Cambridge.

Regier, T. (1996). *The Human Semantic Potential*. MIT Press, Cambridge, MA.

Reichenbach, H. (1947). *Elements of Symbolic Logic*. Macmillan, New York.

Reichert, T. A., Cohen, D. N., and Wong, A. K. C. (1973). An application of information theory to genetic mutations and the matching of polypeptide sequences. *Journal of Theoretical Biology*, *42*, 245–261.

Reichman, R. (1985). *Getting Computers to Talk Like You and Me*. MIT Press, Cambridge, MA.

Reiter, E. (1990). A new model for lexical choice for open-class words. In McKeown, K. R., Moore, J. D., and Nirenburg, S. (Eds.), *Proceedings of the Fifth International Workshop on Natural Language Generation*, Dawson, PA, pp. 23–30.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge. To appear.

Reiter, E., Robertson, R., and Osman, L. (1999). Types of knowledge required to personalise smoking cessation letters. In *Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making*. Springer-Verlag.

Reiter, R. (1980). A logic for default reasoning. *Artificial Intelligence*, *13*, 81–132.

Reithinger, N., Engel, R., Kipp, M., and Klesen, M. (1996). Predicting dialogue acts for a speech-to-speech translation system. In *ICSLP-96*, Philadelphia, PA, Vol. 2, pp. 654–657.

Reithinger, N. and Klesen, M. (1997). Dialogue act classification using language models. In *EUROSPEECH-97*, Vol. 4, pp. 2235–2238.

Resnik, P. (1992). Probabilistic tree-adjoining grammar as a framework for statistical natural language processing. In *COLING-92*, Nantes, France, pp. 418–424.

Resnik, P. (1997). Selectional preference and sense disambiguation. In *Proceedings of ACL SIGLEX Workshop on Tagging Text with Lexical Semantics*, Washington, D.C., pp. 52–57. ACL.

Resnik, P. (1998). Wordnet and class-based probabilities. In Fellbaum, C. (Ed.), *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Rhodes, F. L. (1929). *Beginnings of telephony*. Harper and Brothers, New York.

Rhodes, R. A. (1992). Flapping in American English. In Dressler, W. U., Prinzhorn, M., and Rennison, J. (Eds.), *Proceedings of the 7th International Phonology Meeting*, pp. 217–232. Rosenberg and Sellier.

Riesbeck, C. K. (1975). Conceptual analysis. In Schank, R. C. (Ed.), *Conceptual Information Processing*, pp. 83–156. American Elsevier, New York.

Riesbeck, C. K. (1986). From conceptual analyzer to direct memory access parsing: An overview. In *Advances in Cognitive Science 1*, pp. 236–258. Ellis Horwood, Chichester.

Riley, M. D. (1991). A statistical model for generating pronunciation networks. In *IEEE ICASSP-91*, pp. 737–740. IEEE.

Riley, M. D. (1992). Tree-based modelling for speech synthesis. In Bailly, G. and Beniot, C. (Eds.), *Talking Machines: Theories, Models and Designs*. North Holland, Amsterdam.

Riloff, E. (1993). Automatically constructing a dictionary for information extraction tasks. In *AAAI-93*, Washington, D.C., pp. 811–816.

Riloff, E. and Schmelzenbach, M. (1998). An empirical approach to conceptual case frame acquisition. In *Proceedings of the Sixth Workshop on Very Large Corpora*, Montreal, Canada, pp. 49–56.

Rivest, R. L. (1987). Learning decision lists. *Machine Learning*, *2*(3), 229–246.

Robins, R. H. (1967). *A Short History of Linguistics*. Indiana University Press, Bloomington.

Robinson, J. A. (1965). A machine-oriented logic based on the resolution principle. *Journal of the Association for Computing Machinery*, *12*, 23–41.

Robinson, J. J. (1975). Performance grammars. In Reddy, D. R. (Ed.), *Speech Recognition: Invited Paper Presented at the 1974 IEEE Symposium*, pp. 401–427. Academic Press, New York.

Robinson, S. E. and Sparck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, *27*, 129–146.

Rocchio, J. J. (1971). Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Indexing*, pp. 324–336. Prentice Hall, Englewood Cliffs, NJ.

Roche, E. and Schabes, Y. (1997a). Deterministic part-of-speech tagging with finite-state transducers. In Roche, E. and Schabes, Y. (Eds.), *Finite-State Devices for Natural Language Processing*, pp. 205–239. MIT Press, Cambridge, MA.

Roche, E. and Schabes, Y. (1997b). Introduction. In Roche, E. and Schabes, Y. (Eds.), *Finite-State Devices for Natural Language Processing*, pp. 1–65. MIT Press, Cambridge, MA.

Roelofs, A. (1997). The WEAVER model of word-form encoding in speech production. *Cognition*, *64*, 249–284.

Roland, D. and Jurafsky, D. (1998). How verb subcategorization frequencies are affected by corpus choice. In *COLING/ACL-98*, Montreal, pp. 1122–1128. ACL.

Rosenfeld, R. (1996). A maximum entropy approach to adaptive statistical language modeling. *Computer Speech and Language*, *10*, 187–228.

Roth, D. (1998). Learning to resolve natural language ambiguities: A unified approach. In *AAAI-98*, pp. 806–813.

Roth, D. (1999). Learning in natural language. In *IJCAI-99*, pp. 898–904.

Roth, D. and Zelenko, D. (1998). Part of speech tagging using a network of linear separators. In *COLING/ACL-98*, Montreal, pp. 1136–1142. ACL.

Rounds, W. C. and Kasper, R. T. (1986). A complete logical calculus for record structures representing linguistic information. In *Proceedings of the 1st Annual IEEE Symposium on Logic in Computer Science*, pp. 38–43.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E. and McClelland, J. L. (Eds.), *Parallel Distributed Processing*, Vol. 2, pp. 318–362. MIT Press.

Rumelhart, D. E. and McClelland, J. L. (1986). On learning the past tense of English verbs. In Rumelhart, D. E. and McClelland, J. L. (Eds.), *Parallel Distributed Processing*, Vol. 2, pp. 216–271. MIT Press.

Russell, S. and Norvig, P. (1995). *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ.

Russell, S. W. (1976). Computer understanding of metaphorically used verbs. *American Journal of Computational Linguistics*, 2. Microfiche 44.

Ryder, M. E. (1994). *Ordered Chaos: The Interpretation of English Noun-Noun Compounds*. University of California Press, Berkeley.

Sacks, H., Schegloff, E. A., and Jefferson, G. (1974). A simplest systematics for the organization of turn-taking for conversation. *Language*, *50*(4), 696–735.

Saffran, J. R., Aslin, R. N., and Newport, E. L. (1996). Statistical cues in language acquisition: Word segmentation by infants. In *COGSCI-96*, pp. 376–380.

Sag, I. A. and Liberman, M. (1975). The intonational disambiguation of indirect speech acts. In *CLS-75*, pp. 487–498. University of Chicago.

Sag, I. A. and Wasow, T. (Eds.). (1999). *Syntactic Theory: A Formal Introduction*. CSLI Publications, Stanford, CA.

Saint-Dizier, P. and Viegas, E. (Eds.). (1995). *Computational Lexical Semantics*. Cambridge University Press, New York.

Sakoe, H. and Chiba, S. (1971). A dynamic programming approach to continuous speech recognition. In *Proceedings of the Seventh International Congress on Acoustics, Budapest*, Budapest, Vol. 3, pp. 65–69. Akadémiai Kiadó.

Sakoe, H. and Chiba, S. (1984). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *ASSP-26*(1), 43–49.

Salasoo, A. and Pisoni, D. B. (1985). Interaction of knowledge sources in spoken word identification. *Journal of Memory and Language*, *24*, 210–231.

Salomaa, A. (1969). Probabilistic and weighted grammars. *Information and Control*, *15*, 529–544.

Salton, G. (1971). *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice Hall, Englewood Cliffs, NJ.

Salton, G., Allan, J., and Buckley, C. (1993). Approaches to passage retrieval in full text information systems. In *SIGIR-93*, Pittsburgh, PA, pp. 49–58. ACM.

Salton, G. and Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, *24*, 513–523. Also available in Sparck Jones and Willett (1997).

Salton, G. and Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Information Processing and Management*, *41*, 288–297.

Salton, G. and McGill, M. J. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, NY.

Sampson, G. (1987). Alternative grammatical coding systems. In Garside, R., Leech, G., and Sampson, G. (Eds.), *The Computational Analysis of English*, pp. 165–183. Longman, London and New York.

Sampson, G. (1996). *Evolutionary Language Understanding*. Cassell, London.

Samuel, A. G. (1981). Phonemic restoration: Insights from a new methodology. *Journal of Experimental Psychology: General*, *110*, 474–494.

Samuel, K., Carberry, S., and Vijay-Shanker, K. (1998a). Computing dialogue acts from features with transformation-based learning. In Chu-Carroll, J. and Green, N. (Eds.), *Applying Machine Learning to Discourse Processing. Papers from the 1998 AAAI Spring Symposium*, pp. 90–97. Technical Report SS-98-01.

Samuel, K., Carberry, S., and Vijay-Shanker, K. (1998b). Dialogue act tagging with transformation-based learning. In *COLING/ACL-98*, Montreal, Vol. 2, pp. 1150–1156. ACL.

Samuelsson, C. and Reichl, W. (1999). A class-based language model for large-vocabulary speech recognition extracted

from part-of-speech statistics. In *IEEE ICASSP-99*, pp. 537–540. IEEE.

Sanders, T. J. M., Spooren, W. P. M., and Noordman, L. G. M. (1992). Toward a taxonomy of coherence relations. *Discourse Processes*, *15*, 1–35.

Sanderson, M. (1994). Word sense disambiguation and information retrieval. In *SIGIR-94*, Dublin, Ireland, pp. 142–151. ACM.

Sanfilippo, A. (1993). LKB encoding of lexical knowledge. In Briscoe, T., de Paiva, V., and Copestake, A. (Eds.), *Inheritance, Defaults, and the Lexicon*, pp. 190–222. Cambridge University Press, Cambridge.

Sankoff, D. (1972). Matching sequences under deletion-insertion constraints. *Proceedings of the Natural Academy of Sciences of the U.S.A.*, *69*, 4–6.

Sankoff, D. and Kruskal, J. B. (Eds.). (1983). *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, Reading, MA.

Sato, S. and Nagao, M. (1990). Toward memory-based translation. In *COLING-90*, Helsinki, pp. 247–252.

Scha, R. and Polanyi, L. (1988). An augmented context free grammar for discourse. In *COLING-88*, Budapest, pp. 573–577.

Schabes, Y. (1990). *Mathematical and Computational Aspects of Lexicalized Grammars*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA†.

Schabes, Y. (1992). Stochastic lexicalized tree-adjoining grammars. In *COLING-92*, Nantes, France, pp. 426–433.

Schabes, Y., Abeillé, A., and Joshi, A. K. (1988). Parsing strategies with 'lexicalized' grammars: Applications to Tree Adjoining Grammars. In *COLING-88*, Budapest, pp. 578–583.

Schachter, P. (1985). Parts-of-speech systems. In Shopen, T. (Ed.), *Language Typology and Syntactic Description, Volume 1*, pp. 3–61. Cambridge University Press.

Schank, R. C. (1972). Conceptual dependency: A theory of natural language processing. *Cognitive Psychology*, *3*, 552–631.

Schank, R. C. and Albelson, R. P. (1977). *Scripts, Plans, Goals and Understanding*. Lawrence Erlbaum, Hillsdale, NJ.

Schank, R. C. and Riesbeck, C. K. (Eds.). (1981). *Inside Computer Understanding: Five Programs plus Miniatures*. Lawrence Erlbaum, Hillsdale, NJ.

Schegloff, E. A. (1968). Sequencing in conversational openings. *American Anthropologist*, *70*, 1075–1095.

Schegloff, E. A. (1982). Discourse as an interactional achievement: Some uses of 'uh huh' and other things that come between sentences. In Tannen, D. (Ed.), *Analyzing Discourse: Text and Talk*, pp. 71–93. Georgetown University Press, Washington, D.C.

Schönkfinkel, M. (1924). Über die Bausteine der mathematischen Logik. *Mathematische Annalen*, *92*, 305–316. English Translation appears in *From Frege to Gödel: A Source Book in Mathematical Logic*, Harvard University Press, Cambridge, MA, 1967.

Schubert, L. K. and Pelletier, F. J. (1982). From English to logic: Context-free computation of 'conventional' logical translation. *American Journal of Computational Linguistics*, *8*(1), 27–44.

Schütze, H. (1992). Dimensions of meaning. In *Proceedings of Supercomputing '92*, pp. 787–796. IEEE, IEEE Press.

Schütze, H. (1997). *Ambiguity Resolution in Language Learning: Computational and Cognitive Models*. CSLI Publications, Stanford, CA.

Schütze, H. (1998). Automatic word sense discrimination. *Computational Linguistics*, *24*(1), 97–124.

Schütze, H. and Pedersen, J. (1995). Information retrieval based on word senses. In *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, pp. 161–175.

Schütze, H. and Singer, Y. (1994). Part-of-speech tagging using a variable memory Markov model. In *ACL-94*, Las Cruces, NM, pp. 181–187. ACL.

Schwartz, R. and Chow, Y.-L. (1990). The N-best algorithm: An efficient and exact procedure for finding the N most likely sentence hypotheses. In *IEEE ICASSP-90*, Vol. 1, pp. 81–84. IEEE.

Schwartz, R., Chow, Y.-L., Kimball, O., Roukos, S., Krasnwer, M., and Makhoul, J. (1985). Context-dependent modeling for acoustic-phonetic recognition of continuous speech. In *IEEE ICASSP-85*, Vol. 3, pp. 1205–1208. IEEE.

Scott, D. R. and Souza, C. (1990). Getting the message across in RST-based text generation. In Dale, R., Mellish, C., and Zock, M. (Eds.), *Current Research in Natural Language Generation*, chap. 3. Academic Press.

Searle, J. R. (1975a). Indirect speech acts. In Cole, P. and Morgan, J. L. (Eds.), *Speech Acts: Syntax and Semantics Volume 3*, pp. 59–82. Academic Press, New York.

Searle, J. R. (1975b). A taxonomy of illocutionary acts. In Gunderson, K. (Ed.), *Language, Mind and Knowledge, Minnesota Studies in the Philosophy of Science*, Vol. VII, pp. 344–369. University of Minnesota Press, Amsterdam. Also appears in John R. Searle, *Expression and Meaning: Studies in the Theory of Speech Acts*, Cambridge University Press, 1979.

Searle, J. R. (1980). Minds, brains, and programs. *Behavioral and Brain Sciences*, *3*, 417–457.

Selkirk, E. (1986). On derived domains in sentence phonology. *Phonology Yearbook*, *3*, 371–405.

Seymore, K., Chen, S., and Rosenfeld, R. (1998). Nonlinear interpolation of topic models for language model adaptation. In *ICSLP-98*, Sydney, Vol. 6, p. 2503.

Seymore, K. and Rosenfeld, R. (1997). Using story topics for language model adaptation. In *EUROSPEECH-97*, pp. 1987–1990.

Shannon, C. E. (1938). A symbolic analysis of relay and switching circuits. *Transactions of the American Institute of Electrical Engineers*, *57*, 713–723.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, *27*(3), 379–423. Continued in following volume.

Shannon, C. E. (1951). Prediction and entropy of printed English. *Bell System Technical Journal*, *30*, 50–64.

Shaw, W. M., Burgin, R., and Howell, P. (1996). Performance standards and evaluations in ir test collections: Cluster-based retrieval models. *Information Processing and Management*, *33*(1), 1–14.

Sheil, B. A. (1976). Observations on context free parsing. *SMIL: Statistical Methods in Linguistics*, *1*, 71–109.

Shieber, S. M. (1985a). Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, *8*, 333–343.

Shieber, S. M. (1985b). Using restriction to extend parsing algorithms for complex-feature-based formalisms. In *ACL-85*, Chicago, pp. 145–152.

Shieber, S. M. (1986). *An Introduction to Unification-Based Approaches to Grammar*. University of Chicago Press, Chicago.

Shieber, S. M. (1994). Lessons from a restricted Turing test. *Communications of the ACM*, *37*(6), 70–78.

Shinghal, R. and Toussaint, G. T. (1979). Experiments in text recognition with the modified Viterbi algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-1*, 184–193.

Shoup, J. E. (1980). Phonological aspects of speech recognition. In Lea, W. A. (Ed.), *Trends in Speech Recognition*, pp. 125–138. Prentice-Hall, Englewood Cliffs, NJ.

Shriberg, E. (1994). *Preliminaries to a Theory of Speech Disfluencies*. Ph.D. thesis, University of California, Berkeley, CA.

Shriberg, E., Bates, R., Taylor, P., Stolcke, A., Jurafsky, D., Ries, K., Coccaro, N., Martin, R., Meteer, M., and Ess-Dykema, C. V. (1998). Can prosody aid the automatic classification of dialog acts in conversational speech?. *Language and Speech (Special Issue on Prosody and Conversation)*, *41*(3-4), 439–487.

Shriberg, E., Wade, E., and Price, P. (1992). Human-machine problem solving using spoken language systems (SLS): Factors

affecting performance and user satisfaction. In *Proceedings DARPA Speech and Natural Language Workshop*, Harriman, New York, pp. 49–54. Morgan Kaufmann.

Sidner, C. (1979). Towards a computational theory of definite anaphora comprehension in English discourse. Tech. rep. 537, MIT Artificial Intelligence Laboratory, Cambridge, MA.

Sidner, C. (1983). Focusing in the comprehension of definite anaphora. In Brady, M. and Berwick, R. C. (Eds.), *Computational Models of Discourse*, pp. 267–330. MIT Press, Cambridge, MA.

Siegel, S. and Castellan, Jr., N. J. (1988). *Nonparametric Statistics for the Behavioral Sciences* (Second edition). McGraw-Hill, New York.

Sills, D. L. and Merton, R. K. (Eds.). (1991). *Social Science Quotations*. MacMillan, New York.

Silverman, K., Beckman, M. E., Pitrelli, J., Ostendorf, M., Wightman, C., Price, P., Pierrehumbert, J., and Hirschberg, J. (1992). ToBI: a standard for labelling English prosody. In *ICSLP-92*, Vol. 2, pp. 867–870.

Silverstein, C., Henzinger, M., Marais, H., and Moricz, M. (1998). Analysis of a very large AltaVista query log. Tech. rep. 1998-014, Digital Systems Research Center.

Simmons, R. and Slocum, J. (1972). Generating English discourse from semantic networks. *Communications of the ACM*, *15*(10), 891–905.

Simmons, R. F. (1965). Answering English questions by computer: A survey. *Communications of the ACM*, *8*(1), 53–70.

Simmons, R. F. (1973). Semantic networks: Their computation and use for understanding English sentences. In Schank, R. C. and Colby, K. M. (Eds.), *Computer Models of Thought and Language*, pp. 61–113. W.H. Freeman and Co., San Francisco.

Simmons, R. F. (1978). Rule-based computations on English. In Waterman, D. A. and Hayes-Roth, F. (Eds.), *Pattern-Directed Inference Systems*. Academic Press, New York.

Simmons, R. F. (1983). *Computations from the English*. Prentice Hall, Englewood Cliffs.

Singer, M. (1979). Processes of inference during sentence encoding. *Memory & Cognition*, *7*(3), 192–200.

Singer, M. (1980). The role of case-filling inferences in the coherence of brief passages. *Discourse Processes*, *3*, 185–201.

Singer, M. (1994). Discourse inference processes. In Gernsbacher, M. A. (Ed.), *Handbook of Psycholinguistics*, pp. 479–515. Academic Press, New York.

Sleator, D. and Temperley, D. (1993). Parsing English with a link grammar. In *Proceeedings, Third International Workshop on Parsing Technologies*, Tilburg, The Netherlands/Durbuy, Belgium.

Slobin, D. I. (1996). Two ways to travel. In Shibatani, M. and Thompson, S. A. (Eds.), *Grammatical Constructions: Their Form and Meaning*, pp. 195–220. Clarendon Press, Oxford.

Small, S. L., Cottrell, G. W., and Tanenhaus, M. (Eds.). (1988). *Lexical Ambiguity Resolution*. Morgan Kaufman, San Mateo, CA.

Small, S. L. and Rieger, C. (1982). Parsing and comprehending with Word Experts. In Lehnert, W. G. and Ringle, M. H. (Eds.), *Strategies for Natural Language Processing*, pp. 89–147. Lawrence Erlbaum, Hillsdale, NJ.

Smith, R. W. and Gordon, S. A. (1997). Effects of variable initiative on linguistic behavior in human-computer spoken natural language dialogue. *Computational Linguistics*, *23*(1), 141–168.

Smith, V. L. and Clark, H. H. (1993). On the course of answering questions. *Journal of Memory and Language*, *32*, 25–38.

Smyth, R. (1994). Grammatical determinants of ambiguous pronoun resolution. *Journal of Psycholinguistic Research*, *23*, 197–229.

Soderland, S., Fisher, D., Aseltine, J., and Lehnert, W. G. (1995). CRYSTAL: Inducing a conceptual dictionary. In *IJCAI-95*, Montreal, pp. 1134–1142.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, *28*(1), 11–21.

Sparck Jones, K. (1986). *Synonymy and Semantic Classification*. Edinburgh University Press, Edinburgh. Republication of 1964 PhD Thesis.

Sparck Jones, K. (1997). Summarization. In Cole, R. (Ed.), *Survey of the State of the Art in Human Language Technology*, pp. 232–235. Cambridge University Press, Cambridge.

Sparck Jones, K. and Willett, P. (Eds.). (1997). *Readings in Information Retrieval*. Morgan Kaufmann, San Francisco, CA.

Sproat, R. (1993). *Morphology and Computation*. MIT Press, Cambridge.

Sproat, R. (1994). English noun-phrase prediction for text-to-speech. *Computer Speech and Language*, *8*, 79–94.

Sproat, R. (1998a). Further issues in text analysis. In Sproat, R. (Ed.), *Multilingual Text-To-Speech Synthesis: The Bell Labs Approach*, pp. 89–114. Kluwer, Dordrecht.

Sproat, R. (Ed.). (1998b). *Multilingual Text-To-Speech Synthesis: The Bell Labs Approach*. Kluwer, Dordrecht.

Sproat, R., Möbius, B., Maeda, K., and Tzoukermann, E. (1998). Multilingual text analysis. In Sproat, R. (Ed.), *Multilingual Text-To-Speech Synthesis: The Bell Labs Approach*, pp. 31–86. Kluwer, Dordrecht.

Sproat, R. and Riley, M. D. (1996). Compilation of weighted finite-state transducers from decision trees. In *ACL-96*, Santa Cruz, CA, pp. 215–222. ACL.

Sproat, R., Shih, C., Gale, W. A., and Chang, N. (1996). A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics*, *22*(3), 377–404.

Stalnaker, R. C. (1978). Assertion. In Cole, P. (Ed.), *Pragmatics: Syntax and Semantics Volume 9*, pp. 315–332. Academic Press, New York.

Stanners, R. F., Neiser, J., Hernon, W. P., and Hall, R. (1979). Memory representation for morphologically related words. *Journal of Verbal Learning and Verbal Behavior*, *18*, 399–412.

Stede, M. (1998). A generative perspective on verb alternations. *Computational Linguistics*, *24*(3), 401–430.

Steedman, M. J. (1989). Constituency and coordination in a combinatory grammar. In Baltin, M. R. and Kroch, A. S. (Eds.), *Alternative Conceptions of Phrase Structure*, pp. 201–231. University of Chicago, Chicago.

Steiner, G. (1975). *After Babel*. Oxford University Press, Oxford.

Stetina, J. and Nagao, M. (1997). Corpus based PP attachment ambiguity resolution with a semantic dictionary. In Zhou, J. and Church, K. W. (Eds.), *Proceedings of the Fifth Workshop on Very Large Corpora*, Beijing, China, pp. 66–80. ACL.

Stifelman, L. J., Arons, B., Schmandt, C., and Hulteen, E. A. (1993). VoiceNotes: A speech interface for a hand-held voice notetaker. In *Human Factors in Computing Systems: INTERCHI '93 Conference Proceedings*, Amsterdam, pp. 179–186. ACM.

Stolcke, A., Shriberg, E., Bates, R., Coccaro, N., Jurafsky, D., Martin, R., Meteer, M., Ries, K., Taylor, P., and Van Ess-Dykema, C. (1998). Dialog act modeling for conversational speech. In Chu-Carroll, J. and Green, N. (Eds.), *Applying Machine Learning to Discourse Processing. Papers from the 1998 AAAI Spring Symposium*. Tech. rep. SS-98-01, Stanford, CA, pp. 98–105. AAAI Press.

Stolcke, A. (1995). An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, *21*(2), 165–202.

Stolcke, A. and Shriberg, E. (1996a). Automatic linguistic segmentation of conversational speech. In *ICSLP-96*, Philadelphia, PA, pp. 1005–1008.

Stolcke, A. and Shriberg, E. (1996b). Statistical language modeling for speech disfluencies. In *IEEE ICASSP-96*, Atlanta, GA, Vol. 1, pp. 405–408. IEEE.

Stolz, W. S., Tannenbaum, P. H., and Carstensen, F. V. (1965). A stochastic approach to the grammatical coding of English. *Communications of the ACM*, *8*(6), 399–405.

Streeter, L. (1978). Acoustic determinants of phrase boundary perception. *Journal of the Acoustical Society of America*, *63*, 1582–1592.

Strube, M. and Hahn, U. (1996). Functional centering. In *ACL-96*, Santa Cruz, CA, pp. 270–277. ACL.

Suhm, B. and Waibel, A. (1994). Toward better language models for spontaneous speech. In *ICSLP-94*, Vol. 2, pp. 831–834.

Sumita, E. and Iida, H. (1991). Experiments and prospects of example-based machine translation. In *ACL-91*, Berkeley, CA, pp. 185–192. ACL.

Sundheim, B. (Ed.). (1991). *Proceedings of the Third Message Understanding Conference*, San Mateo, CA. Morgan Kaufmann.

Sundheim, B. (Ed.). (1992). *Proceedings of the Fourth Message Understanding Conference*. Morgan Kaufmann.

Sundheim, B. (Ed.). (1993). *Proceedings, Fifth Message Understanding Conference (MUC-5), Baltimore, MD*. Morgan Kaufmann, San Mateo, CA.

Sundheim, B. (1995a). Overview of results of the MUC-6 evaluation. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Columbia, MD, pp. 13–31.

Sundheim, B. (Ed.). (1995b). *Proceedings of the Sixth Message Understanding Conference*. Morgan Kaufmann.

Sweet, H. (1877). *A Handbook of Phonetics*. Clarendon Press, Oxford.

Tabor, W., Juliano, C., and Tanenhaus, M. K. (1997). Parsing in a dynamical system. *Language and Cognitive Processes*, *12*, 211–272.

Tajchman, G., Fosler, E., and Jurafsky, D. (1995). Building multiple pronunciation models for novel words using exploratory computational phonology. In *EUROSPEECH-95*, pp. 2247–2250.

Talmy, L. (1985). Lexicalization patterns: Semantic structure in lexical forms. In Shopen, T. (Ed.), *Language Typology and Syntactic Description, Volume 3*. Cambridge University Press. Originally appeared as UC Berkeley Cognitive Science Program Report No. 30, 1980.

Talmy, L. (1991). Path to realization: a typology of event conflation. In *BLS-91*, Berkeley, CA, pp. 480–519.

Tappert, C. C., Suen, C. Y., and Wakahara, T. (1990). The state of the art in on-line handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *12*(8), 787–808.

Taylor, P. (2000). Analysis and synthesis of intonation using the Tilt model. *Journal of the Acoustical Society of America*. To appear.

Taylor, P. and Black, A. W. (1998). Assigning phrase breaks from part of speech sequences. *Computer Speech and Language*, *12*, 99–117.

Taylor, P., King, S., Isard, S., and Wright, H. (1998). Intonation and dialog context as constraints for speech recognition. *Language and Speech*, *41*(3-4), 489–508.

ter Meulen, A. (1995). *Representing Time in Natural Language*. MIT Press, Cambridge, MA.

Tesar, B. (1995). *Computational Optimality Theory*. Ph.D. thesis, University of Colorado, Boulder.

Tesar, B. (1996). Computing optimal descriptions for optimality theory grammars with context-free position structures. In *ACL-96*, Santa Cruz, CA, pp. 101–107. ACL.

Tesar, B. and Smolensky, P. (1993). The learnability of optimality theory: An algorithm and some basic complexity results. Tech. rep. CU-CS-678-93, University of Colorado at Boulder Department of Computer Science.

Tesnière, L. (1959). *Eléments de Syntaxe Structurale*. Librairie C. Klincksieck, Paris.

Tetreault, J. R. (1999). Analysis of syntax-based pronoun resolution methods. In *ACL-99*, College Park, MD, pp. 602–605.

Thompson, H. (1977). Strategy and tactics: A model for language production. In *CLS-77*, pp. 651–668.

Thompson, K. (1968). Regular expression search algorithm. *Communications of the ACM*, *11*(6), 419–422.

Tomabechi, H. (1991). Quasi-destructive graph unification. In *ACL-91*, Berkeley, CA, pp. 315–322.

Touretzky, D. S., Elvgren, III, G., and Wheeler, D. W. (1990). Phonological rule induction: An architectural solution. In *COGSCI-90*, pp. 348–355.

Traum, D. R. and Allen, J. (1994). Discourse obligations in dialogue processing. In *ACL-94*, Las Cruces, NM, pp. 1–8. ACL.

Trubetskoi, N. S. (1939). *Grundzüge der Phonologie*, Vol. 7 of *Travaux du cercle linguistique de Prague*. Available in 1969 English translation by Christiane A. M. Baltaxe as *Principles of Phonology*, University of California Press.

Trueswell, J. C. and Tanenhaus, M. K. (1994). Toward a lexicalist framework for constraint-based syntactic ambiguity resolution. In Clifton, Jr., C., Frazier, L., and Rayner, K. (Eds.), *Perspectives on Sentence Processing*, pp. 155–179. Lawrence Erlbaum, Hillsdale, NJ.

Trueswell, J. C., Tanenhaus, M. K., and Garnsey, S. M. (1994). Semantic influences on parsing: Use of of thematic role information in syntactic ambiguity resolution. *Journal of Memory and Language*, *33*, 285–318.

Trueswell, J. C., Tanenhaus, M. K., and Kello, C. (1993). Verb-specific constraints in sentence processing: Separating effects of lexical preference form garden-paths. *Journal of Experimental Psychology: Learning, Memory and Cognition*, *19*(3), 528–553.

Tsujii, J. (1986). Future directions of machine translation. In *COLING-86*, Bonn, pp. 655–668.

Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, *42*, 230–265. Read to the Society in 1936, but published in 1937. Correction in volume 43, 544–546.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, *59*, 433–460.

Tyler, L. K. (1984). The structure of the initial cohort: Evidence from gating. *Perception & Psychophysics*, *36*(5), 417–427.

Usher, A. P. (1954). *A History of Mechanical Inventions*. Harvard University Press, Cambridge, MA.

Uszkoreit, H. (1986). Categorial unification grammars. In *COLING-86*, Bonn, pp. 187–194.

Uszkoreit, H. (Ed.). (1996). *Language Generation*, chap. 4. available at: `http://cslu.cse.ogi.edu/HLTsurvey/`.

van Benthem, J. and ter Meulen, A. (Eds.). (1997). *Handbook of Logic and Language*. MIT Press, Cambridge, MA.

Van Deemter, K. and Odijk, J. (1997). Context modeling and the generation of spoken discourse. *Speech Communication*, *21*(1/2), 101–121.

van Dijk, T. A. and Kintsch, W. (1983). *Strategies of Discourse Comprehension*. Academic Press, New York.

van Lehn, K. (1978). Determining the scope of English quantifiers. Master's thesis, MIT, Cambridge, MA. MIT Technical Report AI-TR-483.

van Rijsbergen, C. J. (1975). *Information Retrieval*. Butterworths, London.

van Santen, J. and Sproat, R. (1998). Methods and tools. In Sproat, R. (Ed.), *Multilingual Text-To-Speech Synthesis: The Bell Labs Approach*, pp. 7–30. Kluwer, Dordrecht.

van Valin, Jr., R. D. (1999). Introduction to Syntax. Unpublished textbook draft.

Vander Linden, K. and Martin, J. H. (1995). Expressing local rhetorical relations in instructional text: A case-study of the purpose relation. *Computational Linguistics*, *21*(1), 29–57.

Vanderwende, L. (1994). Algorithm for the automatic interpretation of noun sequences. In *COLING-94*, Kyoto, pp. 782–788.

Veale, T. and Keane, M. T. (1992). Conceptual scaffolding: A spatially founded meaning representation for metaphor comprehension. *Computational Intelligence*, *8*(3), 494–519.

Veblen, T. (1899). *Theory of the Leisure Class*. Macmillan Company, New York.

Velichko, V. M. and Zagoruyko, N. G. (1970). Automatic recognition of 200 words. *International Journal of Man-Machine Studies*, *2*, 223–234.

Vendler, Z. (1967). *Linguistics in Philosophy*. Cornell University Press, Ithaca, NY.

Veronis, J. and Ide, N. M. (1990). Word sense disambiguation with very large neural networks extracted from machine readable dictionaries. In *COLING-90*, Helsinki, Finland, pp. 389–394.

Vintsyuk, T. K. (1968). Speech discrimination by dynamic programming. *Cybernetics*, *4*(1), 52–57. Russian Kibernetika 4(1):81-88 (1968).

Vitale, T. (1991). An algorithm for high accuracy name pronunciation by parametric speech synthesizer. *Computational Linguistics*, *17*(3), 257–276.

Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, *IT-13*(2), 260–269.

von Neumann, J. (1963). *Collected Works: Volume V*. Macmillan Company, New York.

Voorhees, E. M. (1998). Using WordNet for text retrieval. In Fellbaum, C. (Ed.), *WordNet: An Electronic Lexical Database*, pp. 285–303. MIT Press, Cambridge, MA.

Voorhees, E. M. and Harman, D. (Eds.). (1998). *The Seventh Text Retrieval Conference (TREC-7)*. National Institute of Standards and Technology. NIST Special Publication 500-242. Also available on the Web at NIST's TREC Web page.

Voutilainen, A. (1995). Morphological disambiguation. In Karlsson, F., Voutilainen, A., Heikkilä, J., and Anttila, A. (Eds.), *Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text*, pp. 165–284. Mouton de Gruyter, Berlin.

Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the Association for Computing Machinery*, *21*, 168–173.

Wahlster, W. (1989). One word says more than a thousand pictures. On the automatic verbalization of the results of image sequence analysis systems. *Computers and Artificial Intelligence*, *8*, 479–492.

Wahlster, W., André, E., Finkler, W., Profitlich, H.-J., and Rist, T. (1993). Plan-based Integration of Natural Language and Graphics Generation. *Artificial Intelligence*, *63*(1–2), 387–428.

Waibel, A. (1988). *Prosody and Speech Recognition*. Morgan Kaufmann, San Mateo, CA.

Wald, B. and Shopen, T. (1981). A researcher's guide to the sociolinguistic variable (ING). In Shopen, T. and Williams, J. M. (Eds.), *Style and Variables in English*, pp. 219–249. Winthrop Publishers, Cambridge, MA.

Walker, M. A. (1989). Evaluating discourse processing algorithms. In *ACL-89*, Vancouver, Canada, pp. 251–260. ACL.

Walker, M. A., Iida, M., and Cote, S. (1994). Japanese discourse and the process of centering. *Computational Linguistics*, *20*(2).

Walker, M. A., Joshi, A. K., and Prince, E. (Eds.). (1998). *Centering in Discourse*. Oxford University Press.

Walker, M. A., Litman, D. J., Kamm, C. A., and Abella, A. (1997). Paradise: A framework for evaluating spoken dialogue agents. In *ACL/EACL-97*, Madrid, Spain, pp. 271–280. ACL.

Walker, M. A., Maier, E., Allen, J., Carletta, J., Condon, S., Flammia, G., Hirschberg, J., Isard, S., Ishizaki, M., Levin, L., Luperfoy, S., Traum, D., and Whittaker, S. (1996). Penn multiparty standard coding scheme: Draft annotation manual. `www.cis.upenn.edu/~ircs/discourse-tagging/newcoding.html`.

Walker, M. A. and Whittaker, S. (1990). Mixed initiative in dialogue: An investigation into discourse segmentation. In *ACL-90*, Pittsburgh, PA, pp. 70–78. ACL.

Wang, M. Q. and Hirschberg, J. (1992). Automatic classification of intonational phrasing boundaries. *Computer Speech and Language*, *6*(2), 175–196.

Wanner, E. and Maratsos, M. (1978). An ATN approach to comprehension. In Halle, M., Bresnan, J., and Miller, G. A. (Eds.),

Linguistic Theory and Psychological Reality, pp. 119–161. MIT Press, Cambridge, MA.

Ward, N. (1994). *A Connectionist Language Generator*. Ablex.

Warnke, V., Kompe, R., Niemann, H., and Nöth, E. (1997). Integrated dialog act segmentation and classification using prosodic features and language models. In *EUROSPEECH-97*, Vol. 1, pp. 207–210.

Warren, R. M. (1970). Perceptual restoration of missing speech sounds. *Science*, *167*, 392–393.

Waugh, L. R. (1976). The semantics and paradigmatics of word order. *Language*, *52*(1), 82–107.

Weaver, W. (1949/1955). Translation. In Locke, W. N. and Boothe, A. D. (Eds.), *Machine Translation of Languages*, pp. 15–23. MIT Press, Cambridge, MA. Reprinted from a memorandum written by Weaver in 1949.

Webber, B., Knott, A., Stone, M., and Joshi, A. (1999). Discourse relations: A structural and presuppositional account using lexicalised TAG. In *ACL-99*, College Park, MD, pp. 41–48. ACL.

Webber, B. L. (1978). *A Formal Approach to Discourse Anaphora*. Ph.D. thesis, Harvard University.

Webber, B. L. (1983). So what can we talk about now?. In Brady, M. and Berwick, R. C. (Eds.), *Computational Models of Discourse*, pp. 331–371. The MIT Press, Cambridge, MA. Reprinted in Grosz et al. (1986).

Webber, B. L. (1991). Structure and ostension in the interpretation of discourse deixis. *Language and Cognitive Processes*, *6*(2), 107–135.

Webber, B. L. and Baldwin, B. (1992). Accommodating context change. In *ACL-92*, Newark, DE, pp. 96–103. ACL.

Weber, D. J., Black, H. A., and McConnel, S. R. (1988). Ample: A tool for exploring morphology. Tech. rep. Occasional Publications in Academic Computing No. 12, Summer Institute of Linguistics, Dallas.

Weber, D. J. and Mann, W. C. (1981). Prospects for computer-assisted dialect adaptation. *American Journal of Computational Linguistics*, *7*, 165–177. Abridged from Summer Instititute of Linguistics *Notes on Linguistics* Special Publication 1, 1979.

Weber, E. G. (1993). *Varieties of Questions in English Conversation*. John Benjamins, Amsterdam.

Weischedel, R. (1995). BBN: Description of the PLUM system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Francisco, pp. 55–70. Morgan Kaufmann.

Weischedel, R., Meteer, M., Schwartz, R., Ramshaw, L. A., and Palmucci, J. (1993). Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics*, *19*(2), 359–382.

Weizenbaum, J. (1966). ELIZA – A computer program for the study of natural language communication between man and machine. *Communications of the ACM*, *9*(1), 36–45.

Weizenbaum, J. (1976). *Computer Power and Human Reason: From Judgement to Calculation*. W.H. Freeman and Company, San Francisco.

Wells, J. C. (1982). *Accents of English*. Cambridge University Press.

Wells, J. C. (1990). *Pronunciation Dictionary*. Longman, London.

Whitelock, P. (1992). Shake-and-bake translation. In *COLING-92*, Nantes, France, pp. 784–791.

Whittemore, G., Ferrara, K., and Brunner, H. (1990). Empirical study of predictive powers of simple attachment schemes for post-modifier prepositional phrases. In *ACL-90*, Pittsburgh, PA, pp. 23–30. ACL.

Wierzbicka, A. (1996). *Semantics: Primes and Universals*. Oxford University Press, New York.

Wilensky, R. (1983). *Planning and Understanding*. Addison-Wesley, Reading, MA.

Wilensky, R. and Arens, Y. (1980). PHRAN: A knowledge-based natural language understander. In *ACL-80*, Philadelphia, PA, pp. 117–121. ACL.

Wilks, Y. (1975a). An intelligent analyzer and understander of English. *Communications of the ACM*, *18*(5), 264–274.

Wilks, Y. (1975b). Preference semantics. In Keenan, E. L. (Ed.), *The Formal Semantics of Natural Language*, pp. 329–350. Cambridge Univ. Press, Cambridge.

Wilks, Y. (1975c). A preferential, pattern-seeking, semantics for natural language inference. *Artificial Intelligence*, *6*(1), 53–74.

Wilks, Y. (1978). Making preferences more active. *Artificial Intelligence*, *11*(3), 197–223.

Wilks, Y., Slator, B. M., and Guthrie, L. M. (1996). *Electric Words: Dictionaries, Computers, and Meanings*. MIT Press, Cambridge, MA.

Willett, P. (1988). Recent trends in hierarchic document clustering: A critical review. *Information Processing and Management*, *24*(5), 577–597.

Winograd, T. (1972a). Understanding natural language. *Cognitive Psychology*, *3*(1). Reprinted as a book by Academic Press, 1972.

Winograd, T. (1972b). *Understanding Natural Language*. Academic Press, New York.

Withgott, M. M. and Chen, F. R. (1993). *Computational Models of American Speech*. Center for the Study of Language and Information.

Witten, I. H. and Bell, T. C. (1991). The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, *37*(4), 1085–1094.

Witten, I. H. (Ed.). (1982). *Principles of Computer Speech*. Academic Press, New York.

Wolfram, W. A. (1969). *A Sociolinguistic Description of Detroit Negro Speech*. Center for Applied Linguistics, Washington, D.C.

Woods, W. A. (1967). *Semantics for a Question-Answering System*. Ph.D. thesis, Harvard University.

Woods, W. A. (1973). Progress in natural language understanding. In *Proceedings of AFIPS National Conference*, pp. 441–450.

Woods, W. A. (1975). What's in a link: Foundations for semantic networks. In Bobrow, D. G. and Collins, A. M. (Eds.), *Representation and Understanding: Studies in Cognitive Science*, pp. 35–82. Academic Press, New York.

Woods, W. A. (1977). Lunar rocks in natural English: Explorations in natural language question answering. In Zampolli, A. (Ed.), *Linguistic Structures Processing*, pp. 521–569. North Holland, Amsterdam.

Woods, W. A. (1978). Semantics and quantification in natural language question answering. In Yovits, M. (Ed.), *Advances in Computers*, Vol. 17, pp. 2–87. Academic Press, New York.

Woods, W. A., Kaplan, R. M., and Nash-Webber, B. (1972). The Lunar Sciences Natural Language Information System: Final report. Tech. rep. 2378, Bolt, Beranek, and Newman, Inc., Cambridge, MA.

Wooters, C. and Stolcke, A. (1994). Multiple-pronunciation lexical modeling in a speaker-independent speech understanding system. In *ICSLP-94*, Yokohama, Japan, pp. 1363–1366.

Woszczyna, M. and Waibel, A. (1994). Inferring linguistic structure in spoken language. In *ICSLP-94*, Yokohama, Japan, pp. 847–850.

Wu, D. (1992). *Automatic Inference: A Probabilistic Basis for Natural Language Interpretation*. Ph.D. thesis, University of California, Berkeley, Berkeley, CA. UCB/CSD 92-692.

Wu, D. and Wong, H. (1998). Machine translation with a stochastic grammatical channel. In *COLING/ACL-98*, Montreal, pp. 1408–1414. ACL.

Wundt, W. (1900). *Völkerpsychologie: eine Untersuchung der Entwicklungsgesetze von Sprache, Mythus, und Sitte*. W. Engelmann, Leipzig. Band II: Die Sprache, Zweiter Teil.

Yaeger, L. S., Webb, B. J., and Lyon, R. F. (1998). Combining neural networks and context-driven search for online, printed handwriting recognition in the NEWTON. *AI Magazine*, *19*(1), 73–89.

Yankelovich, N., Levow, G.-A., and Marx, M. (1995). Designing SpeechActs: Issues in speech user interfaces. In *Human Factors in Computing Systems: CHI '95 Conference Proceedings*, Denver, CO, pp. 369–376. ACM.

Yarowsky, D. (1994). Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *ACL-94*, Las Cruces, NM, pp. 88–95. ACL.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised methods. In *ACL-95*, Cambridge, MA, pp. 189–196. ACL.

Yarowsky, D. (1996). Homograph disambiguation in text-to-speech synthesis. In van Santen, J., Sproat, R., Olive, J., and Hirschberg, J. (Eds.), *Progress in Speech Synthesis*, pp. 159–175. Springer-Verlag, New York.

Yeh, C.-L. and Mellish, C. (1997). An empirical study on the generation of anaphora in Chinese. *Computational Linguistics*, *23*(1), 169–190.

Yngve, V. H. (1955). Syntax and the problem of multiple meaning. In Locke, W. N. and Booth, A. D. (Eds.), *Machine Translation of Languages*, pp. 208–226. MIT Press, Cambridge, MA.

Yngve, V. H. (1960). A model and an hypothesis for language structure. *Proceedings of the American Philosophical Society*, *104*, 444–466.

Yngve, V. H. (1970). On getting a word in edgewise. In *CLS-70*, pp. 567–577. University of Chicago.

Young, M. and Rounds, W. C. (1993). A logical semantics for nonmonotonic sorts. In *ACL-93*, Columbus, OH, pp. 209–215. ACL.

Young, S. J. and Woodland, P. C. (1994). State clustering in HMM-based continuous speech recognition. *Computer Speech and Language*, *8*(4), 369–394.

Younger, D. H. (1967). Recognition and parsing of context-free languages in time n^3. *Information and Control*, *10*, 189–208.

Zappa, F. and Zappa, M. U. (1982). Valley girl. From Frank Zappa album *Ship Arriving Too Late To Save A Drowning Witch*.

Zavrel, J. and Daelemans, W. (1997). Memory-based learning: Using similarity for smoothing. In *ACL/EACL-97*, Madrid, Spain, pp. 436–443. ACL.

Zechner, K. and Waibel, A. (1998). Using chunk based partial parsing of spontaneous speech in unrestricted domains for reducing word error rate in speech recognition. In *COLING/ACL-98*, Montreal, pp. 1453–1459. ACL.

Zernik, U. (1987). *Strategies in Language Acquisition: Learning Phrases from Examples in Context*. Ph.D. thesis, University of California, Los Angeles, Computer Science Department, Los Angeles, CA.

Zernik, U. (1991). Train1 vs. train2: Tagging word senses in corpus. In *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pp. 91–112. Lawrence Erlbaum, Hillsdale, NJ.

Zhou, G. and Lua, K. (1998). Word association and MI-trigger-based language modelling. In *COLING/ACL-98*, Montreal, pp. 1465–1471. ACL.

Zipf, G. (1949). *Human Behavior and the Principle of Least Effort*. Addison–Wesley, Cambridge, MA.

Zue, V., Glass, J., Goodine, D., Leung, H., Phillips, M., Polifroni, J., and Seneff, S. (1989). Preliminary evaluation of the VOYAGER spoken language system. In *Proceedings DARPA Speech and Natural Language Workshop*, Cape Cod, MA, pp. 160–167. Morgan Kaufmann.

Zue, V., Glass, J., Goodine, D., Leung, H., Phillips, M., Polifroni, J., and Seneff, S. (1991). Integration of speech recognition and natural language processing in the MIT VOYAGER system. In *IEEE ICASSP-91*, pp. I.713–716. IEEE.

Zwicky, A. (1972). On Casual Speech. In *CLS-72*, pp. 607–615. University of Chicago.

Zwicky, A. and Sadock, J. (1975). Ambiguity tests and how to fail them. In Kimball, J. (Ed.), *Syntax and Semantics 4*, pp. 1–36. Academic Press, New York.

术 语 表

l-expressions　l 表达式
l-notation　l 符号
l-reduction　l 化简
A* decoder　A* 解码器
A* evaluation function　A* 评估函数
abduction　溯因推理
accented syllable　重音音节
accepting state　接收状态
accomplishment expression　完成体表示
accuracy　正确度
achievement expression　达成体表示
ACL　计算语言学会
acronymsw　首字母缩写词
acoustic model　声学模型
acoustic phonetics　声学语音学
action schema　行为方案
add-one smoothing　加 1 平滑
ad hoc retrieval　特定检索
adjacency pair　毗邻对
adjective　形容词
adjective phrase　形容词短语
advanced operators　高级算符
affix　词缀
agent　施事
agglomerative clustering　合并聚类
agglutinative languages　黏着语
aggregation　集结
agreement　一致关系
alignment　对齐
allophonic　音位
allophonic variation　音位变异
alveolar　齿龈音
ambiguity　歧义
ambiguity resolution　歧义判定
amplitude　振幅
anaphora　复指
anaphoric　复指语
anchors　锚号
ANLP　应用自然语言处理
antecedent　前提
antecedent　先行词
anti-modularist　反模块主义者
appropriateness condition　适切性条件
approximant　半元音
argument　论元
argument　主项
article　冠词
articulatory phonetics　发音语音学
artificial agent　智能机器人
Artificial Intelligence (AI)　人工智能
Artificial Neural Networks (ANN)　人工神经网络
aspect　体
assertives　断言语
assimilation　同化
attachment ambiguity　附着歧义
attentional state　关注状态
attribute-value matrix (AVM)　特征 – 值矩阵
attribute grammar　属性语法
augmentative and alternative communication　增强交替通信
automatic speech recognition　自动语音识别
automatic speech understanding　自动语音理解
automaton intersection　自动机交合
auxiliary　助动词
back vowels　后元音
backchannel　反输
backoff　回退
backoff and discounting　回退与打折
backward chaining　自后向前链接
bag of word model　单词包模型

Baum-Welch algorithm Baum-Welch 算法，又称向前向后算法
baseline 下界底线
basic phrases 基本短语
Bays inference Bays 推理
BDI model BDI（belief-desire-intention）模型
beam search 定向束搜索
best first search 最佳优先搜索
best-first 最佳优先搜索算法
bidirectional grammar 双向语法
bigram model 二元模型
belief network 信念网络
bilingual corpra 双语语料库
binary tree 二叉树
bitmap 位图
body 体
bootstrapped 自举
bootstrapping approach 自举方法
bottom-up 自底向上
bottom-up parsing 自底向上的剖析
boundary tone 边界声调
breadth-first research 广度优先搜索
Brill tagger 标注算法
British National Corpus(BNC) 英国国家语料库
broad transcription 宽式音标
business meeting summarization 商业会议摘要
canned text 罐装文本
canonical form 规范形式
Cartesian product 笛卡儿积
cascades 层叠式
case role analysis 格角色分析
cardinal number 基数词
cataphora 提前指代
category lexicon 范畴词表
category grammar 范畴语法
ceiling 上界顶线
center-embedded sentences 中心嵌入句
central noun 中心名词
cepstral coefficients 倒谱系数
chart entry 线图项目
Chomsky hierarchy Chomsky 层级
Chomsky normal form Chomsky 范式
circumfix 位缀
class-based *N*-grams 基于类的 *N* 元语法
closed class 封闭类
closure 闭包
cluster(clustering) 聚类
cluster hypothesis 聚类假设
co-articulation 协同发音
cochlea 耳蜗
cognition 认知
cognitive errors 认知错误
coherence 连贯
coherence relation 连贯关系
cohesive device 衔接语
collection 文档集
combination rules 结合规则
combinatory categorical grammar 组合范畴语法
combinatory rules 组合规则
commissives 承诺语
common ground 共同基础
common noun 普通名词
complementation 补语化
complementizers 标补语
complexity 计算复杂性
complex-term 复杂项
componential analysis 成分分析
composition 组合
compositional semantics 组合语义学
computational linguistics 计算语言学
computational phonology 计算音系学
computational psycholiguistics 计算心理语言学
computer-aided human translation 机助人译
concatenation 毗连
concatenation cost 毗连成本
concatenative synthesis 毗连合成
conceptual dependency 概念依存
confusion matrix 含混矩阵
consequent 结论
consonant 辅音
constant 常量
constituency 组成性

constituent 成分
Constraint Grammar 约束语法
constraint-based formalism 基于约束的形式化方法
Construction Grammar 构造语法
content selection 内容选择
context 上下文
context-free grammar(CFG) 上下文无关语法
context-sensitive grammar 上下文有关语法
context-sensitive spelling error correction 上下文有关的错拼更正
contingency table 列联表
continued attention 继续关注
continuer 接续
continuity hypothesis 连续性假说
contrastive knowledge 对比知识
control information 控制信息
controlled language 受控语言
conversational agent 会话代理
conversational implicature 会话隐涵
conversational move 会话行动
Conversational Structure 会话结构
coordinating conjunction 并列连接词
coordinating relation 并列关系
coordination ambiguity 并列连接歧义
corefer(coreference) 同指
corpora 语料库
correction of spelling errors 拼写错误校正
correction subdialogue 修正子对话
count noun 可数名词
coverage 覆盖面
cross entropy 交叉熵
cross-language information retrieval 跨语言信息检索
cross-serial dependencies 交叉连续依存关系
cross-serial dependencies 交叉序列依存
cross-validation 交叉确认
cue model 提示模型
cue word 线索词
CYK(Cocke-Younger-Kasami) algorithm CYK 算法
data-directed search 数据制导的搜索
data-driven 数据驱动
dative alternation 与格交替
decision tree 决策树
declaration 宣告语
declarative constraint 陈述式限制
declarative structure 陈述式结构
decoding 解码
decoding problem 解码问题
decoding stage 解码阶段
deduction 演绎
deep case 深层格
deep role 深层角色
default pair 默认偶对
definite noun phrase 有定名词短语
deleted interpolation 删除插值法
demarcated adverbial PP 可区分的状语介词短语
demonstrative 指示词
dental 齿音
dependency grammar 依存语法
depth-first search 深度优先搜索
depth-first-strategy 深度优先策略
dereferencing 解除参照
derivation 派生
Derivational Morphology 派生形态学
deterministic 确定性
deterministic finite-state automation 有限状态自动机
DFSA 确定的自动机
diacritics 附加符号
dialect variation 方言变异
dialogue act 对话行为
dialogue obligation 对话约束
diphones 双音子
direct MT 直接机器翻译
directed acyclic graph 非成圈图
directed graph 有向图
directive 指令语
disambiguation 歧义消解
disambiguation 歧义消解
disambiguation 排歧

discontinuous set 不连续集
discounting 打折
discourse context 话语上下文
discourse convention 话语规约
discourse interpretation 话语解释
discourse model 话语模型
discourse model paradigm 话语模型范型
discourse planner 话语规划器
discourse segment 话语片断
discourse structure 话语结构
disjunction 析取
dissimilation 异化
document 文档
dominance 支配
dominance 支配关系
dotted rule 点规则
draft translation 草稿翻译
duration 音长
duration 音延
dynamic programming 动态规划
dynamic programming alignment 动态规划对齐
dynamic programming invariant 动态规划恒定
dynamic time warping 动态时间偏移
Earley algorithm Earley 算法
edit cost 编辑耗费
embedded training 嵌入训练
empiricist tradition 经验主义传统
entity 实体
entropy 熵
entropy rate 熵率
epenthesis 增音
error back-propagation 错误反向传播
evaluation 评测
event time 事件时间
Example-based Machine Translation 基于实例的机器翻译
existential predicate nominal 存在谓词性名词
existential quantifier 存在量词
Expectation-Maximization(EM) algorithm 期望最大算法或 EM 算法
experiencer 体验者
expressiveness 表达能力
expressives 表情语
extraction 抽取
fail state 失败状态
F-measure F 系数
fallout 判误率
faithfulness in MT 机器翻译的忠实性
Fast Fourier Transform(FFT) 快速 Fourier 变换
feasible pair 可行偶对
feature 特征
feature co-occurrence restriction 特征共现限制
feature extraction 特征抽取
feature extraction stage 特征抽取阶段
feature path 特征路径
feature structure 特征结构
feature structure unification 特征结构合一
feature vector 特征矢量
feature-augmented grammars 特征增益语法
filled pause 有声停顿
filter transducer 过滤转录机
filtering 过滤
final state 最后状态
finite-state automaton 有限状态自动机
finite-state transducers 有限状态转录机
First In First Out 先进先出
first order logic 一阶逻辑
First Order Predicate Calculus 一阶谓词演算
flapping 闪音化
fluency 流畅性
formal language 形式语言
formal meaning representation 形式意义表示
formal rule system 形式规则系统
formal syntax 形式句法
formants 共振峰
Forward-backward algorithm 向前向后算法
forward algorithm 向前算法
forward chaining 自前向后链接
fragment 片断
frame 框架
frame-based representation 基于框架的表示
frames 框架

frequency 频度
fricative 擦音
front vowels 前元音
full listing 完全枚举法
function 函数
function word 虚词
Functional Unification Grammar(FUG) 功能合一语法
functors 函子
fusion language 溶合语
garden path sentence 花园幽径句
generalization 一般化
Generalized Phrase Structure Grammar(SPSG) 广义短语结构语法
generation 生成
generative grammar 生成语法
generative power 生成能力
generator 生成器
generic 类属
genitive 所属格
global focus 全局焦点
glottal 喉音
goal-directed search 目标制导的搜索
Gold Standard 黄金标准
grammar 语法
grammar checking 语法检查
grammatical 合语法的
grammatical heads 语法中心语
grammatical relation 语法关系
greedy algorithm 贪心算法
grounding （对话的）共同基础
head 中心语
Head-Driven Phrase Structure Grammar(HPSG) 中心语驱动的短语结构语法
head feature 中心语特征
head transducer 中心语转录机
helping verbs 辅助动词
high vowels 高元音
HMM 隐 Markov 模型
homograph 同形异义词
homomorphism 同态
homonymy 同形关系
homophone 同音异义词
hypernym 上义词
hyponymy 上下位关系
ideational meta-function 概念元功能
ill-formedness 非良构
illocutionary act 示意言行
illocutionary force 示意语力
immediate focus 紧邻焦点
immediate-constituent analysis 直接成分分析法
imperative structure 命令式结构
implicature 对话的隐涵
indefinite noun phrase 不定名词短语
independence assumption 独立性假设
indeterminacy 不确定性
indirect object or oblique 间接宾语或旁格
inferrable 推理对象
infix 中缀
inflection 屈折
Inflectional Morphology 屈折形态学
information extraction 信息抽取
information retrieval 信息检索
informational structure 信息结构
information-sharing subdialogue 信息共享子对话
Inside-Outside algorithm 向内 - 向外算法
intentional approach 意图方法
intentional structure 意图结构
interlingua 中间语
intermediate representations 中间表示
intermediate tape 中间带子
interpersonal meta-function 人际元功能
interruption point 停顿点
intonation phrase 语调短语
intonational unit 语调单位
intransitive 不及物动词
inverse document frequency 倒排文档频率
isolated-word speech recognition 孤立单词语音识别
isolating language 孤立语
isomorphic 同构
joint probability 联合概率

knowledge precondition 知识前提
labial 唇音
language model 语言模型
large vocabulary continuous speech recognition 大词汇量连续语音识别
Last In First Out 后进先出
Latent Semantic Analysis 潜在语义分析
Latent Semantic Indexing(LSI) 潜在语义索引
lemma 词目
lenient composition operator 宽容组合算符
lexeme 词位
lexical bias effect 词汇倾向效应
lexical dependency 概率词汇依存关系
lexical disambiguation 词汇排歧
Lexical Functional Grammar(LFG) 词汇功能语法
lexical gap 词汇空白
lexical head 词汇中心语
lexical level 词汇层
lexical rule 词汇规则
lexical selection 词汇选择
lexical semantic processing 词汇语义处理
lexical semantics 词汇语义学
lexical stress 词重音
lexical subcategorization preference 词汇次范畴化的优先性
lexical transfer 词汇转换
lexical variation 词汇变异
lexicalized grammar 词汇化语法
Lexicalized Tree Adjoining Grammar 词汇化的树邻接语法
lexicon 词表
likelihood 似然度
Linguistic Data Consortium(LDC) 语言数据联盟
linguistic structure 语言结构
Link Grammar 链语法
Linking theory 关联理论
literal meaning 字面意义
local ambiguity 局部歧义
localization 本地化
locutionary act 发话言行
logical connectives 逻辑连词
logic-based paradigm 基于逻辑的范型
loglinear model 对数线性模型
logographic 表意文字
long-distance-dependencies 长距离依存关系
Longman's Dictionary of Contemporary English 现代英语朗文词典
low vowels 低元音
Machine learning 机器学习
machine readable dictionary 机器可读词典
machine translation 机器翻译
machine-state 机器状态
manner of articulation 发音方法
mass noun 物质名词
maxim of manner 方式准则
maxim of quality 质量准则
maxim of quantity 数量准则
maxim of relevance 相关准则
maximum entropy models 最大熵模型
Maximum Likelihood Estimation 最大似然估计
maximum probability alignment 最大概率对齐
meaning postulates 意义假设
meaning representation 意义表示
meaning representation language 意义表示语言
meaning structure of language 语言的意义结构
metaphor 隐喻
microgrammar 微语法
microplanning 微规划
microworld 微世界
mid vowels 中元音
minimum description length 最小描述长度
minimum edit distance 最小编辑距离
minimum edit distance algorithm 最小编辑距离算法
minimum redundancy 最小羡余法
mirror image 镜像
modal logic 模态逻辑
modal operator 模态算符
modal verb 情态动词
modularist 模块主义者
modus ponens 取式推理
morpheme 语素

morpheme boundary 语素边界
morphological parsing 形态剖析
morphological rules 形态规则
morphotactics 形态顺序规则
multiple words 多项词
multiple-pass decoding 多遍解码
mutual information 互信息
narrow transcription 严式音标
nasal(nasal sounds) 鼻音
nasal tract 鼻腔
Natural Language Generation(NLG) 自然语言生成
Natural Language Processing(NLP) 自然语言处理
Natural Language Understanding(NLU) 自然语言理解
negotiation subdialogue 磋商子对话
neighborhood 邻近效应
nested sentences 嵌套句
neural net models 神经网络模型
node 结点
noisy channel 噪声信道
nominal 名词性成分
nominal value 标称值
nominalization 名词化
non-adverbial emphasis 非状语强调
non-concatenative morphology 非毗连性语素
non-deterministic finite-state automata 非确定的有限状态自动机
non-terminals 非终极符号
normal form 范式
normalizing 归一化
noun group 名词词组
noun phrase 名词短语
noun-phrase bracketing ambiguity 名词短语层次歧义
NP-completeness NP 完全问题
nucleus 核心
number agreement 数的一致
number of sense 涵义数
numeral value 数值
object 宾语
object hierarchy 对象层次
observation likelihood probabilities 观察似然概率
observation sequence 观察序列
observation symbols 观察符号
one sense per collocation 一个搭配一个涵义
one-anaphora 单个复指
one-pass decoding 一遍解码
on-line handwriting recognition 联机手写体识别
ontology 本体知识体系
open class 开放类
operation list 操作表
operator 算符
operator precedence hierarchy 算符优先层级
Optical Character Recognition(OCR) 光学字符识别
optional constituent 随选成分
oral tract 口腔
ordinal number 序数词
orthographic form 正字形
orthographic rule 正词法规则
overgeneration 过度生成
pairing 偶对
palatal 上腭音
parallelism 平行性
parameter tying 参数约束
parsing 剖析
part of speech 词类
partial credit 部分可信度
partial parse 局部剖析
particle 小品词
part-of-speech tagger 词类标注器
part-of-speech tagging 词类标注
patient 受事
pattern 模式
Penn Treebank 宾州树库
perceptual linear predictive 感知线性预测
performative sentence 施行句
perlocutionary act 取效言行
perplexity 困惑度
person and case agreement 人称和格的一致
personal pronoun 人称代词
phone 音子

phone set 音子集
phone stage 音子阶段
phoneme 音位
phonetic 语音学
phonological form 音素形
phonological learning 音位学习
phonological rule 音位规则
phonology 音系学
phrase structure grammar 短语结构语法
pipe symbol 析取符
pitch 音高
pitch accent 音高重音
place of articulation 发音部位
plan inference 计划推理
plan-based 基于计划
plural 复数
polysemy 多义关系
polysynthetic language 多式综合语
post-editing 译后编辑
post-head modifiers 后修饰语
posterior probability 后验概率
postposition 后置词
power 音强
practical semantic analysis 实用的语义分析
pragmatic force 语用语力
pragmatics 语用学
precision 精度
precision 准确率
precondition 前提
predicate 谓词
predicate 谓词演算
predicate-argument structure 谓词论元结构
pre-editing 译前编辑
prefixes 前缀
prehead modifiers 前修饰语
primitive decomposition 基元分解
primitives 基原
principle of compositionality 组合性原则
Prior probability 先验概率
priority union 优先合并
Probabilistic Context-Free Grammar(PCFG) 概率上下文无关语法
probabilistic parsing 概率剖析
probabilistic rule 概率规则
probabilistic transduction 概率转移
probability density function 概率强度函数
probability theory 概率论
production system 产生式系统
prominence 突显度
pronoun 代词
pronunciation 发音
pronunciation dictionaries 发音词典
pronunciation variation 发音变异
proper noun 专有名词
prosodic phrasing 韵律短语
Prosody 韵律
pseudocode 伪代码
pumping lemma 抽吸引理
quantifiers 量词
quantization 量化
query 查询条件
query expansion 查询扩展
Question Answering(QA) task 问答任务
queue 队列
reading lip 唇读
realization statement 实现语句
real-word error detection 真词错误更正
real-word errors 真词错误
real-word spelling error 更正真词错拼
recall 召回率
recognizer 识别器
recursion 递归
recursive enumerable languages 递归可枚举语言
Recursive Transition Network 递归转移网络
reductionist 还原主义者
reentrant structure 重入结构
reference 所指
reference point 参照点
reference resolution 参照消解
reference transcription 参照转写

referent 所指的对象
referential opaque 参照不透明
referential transparent 参照透明
referring expression 所指表达
referring expression generation 所指表达生成
register 语域
regressive assimilation 回退同化
regular expression 正则表达式
regular grammar 正则语法
regular language 正则语言
regular relation 正则关系
relater 关联器
relative frequency 相对频度
relevance 相关性
relevance feedback 相关反馈
relevant next contribution 相关邻接贡献
reparandum 待修正片断
repeated phoneme bias 重复音位倾向效应
repetition prime experiment 重复优先实验
repetition priming 重复优先
residual collection 剩余文档集
resolve 消解
resonating cavities 共鸣腔
rheme 述位
rhetorical relation 修辞关系
rhetorical relation planning 修辞关系规划
right-linear 右线性的
right-recursive rule 右递归规则
root 词根
root-and-pattern morphology 词根与模式语素
rough translation 粗略翻译
routing 分发
rule-based tagger 基于规则的标注算法
rule-based tagging 基于规则的标注
rule-to-rule 规则到规则
rule-to-rule hypothesis 规则到规则假设
salience factor 显著因子
sampling 抽样
Sapir-Whorf hypothesis 萨丕尔 – 沃尔夫假说
satellite 外围
satisfaction-precedence 满足 – 优先关系
search-state 搜索状态
segmentation 切分
selectional association 选择关联系数
selectional restriction 选择限制
selectional restriction-based 基于选择限制
self-reference 自引
semantic analysis 语义分析
semantic analyzer 语义分析器
semantic attachment 语义附着
semantic coherence 语义连贯
semantic composition 语义合成
semantic field 语义场
semantic grammar 语义语法
semantic networks 语义网络
semantic restrictions 语义限制
semantic roles 语义角色
send verb 传送动词
sense 涵义
sentence alignment 句子对齐
sentence segmentation 句子切分
sentential complement 句子补语
series of phones 音子序列
shallow parse 浅层剖析
signal analysis 信号分析
signal stage 信号处理阶段
single initiative 单一主动
single-error misspelling 单个错误的错拼
sink state 吸收状态
situational context 情境上下文
slight generalization 轻度泛化
slips of the tongue 言语失误
slot-filler 槽填充
smoothing 平滑
smoothing algorithm 平滑算法
sociolinguistic variation 社会语言变异
sound inference 可靠推理
SOV language SOV 语言
speaker-independent 不依赖于说话人
specialization 特殊化

spectral feature 声谱特征
spectral peak 声谱峰
spectrogram 频谱
speech act 言语行为
speech act interpretation 言语行为解释
speech recognition 语音识别
speech synthesis 语音合成
spelling correction 拼写校正
spelling rules 拼写规则
start state 初始状态
start symbol 初始符号
state machine 状态机器
state-space search 状态空间搜索
state-transition table 状态转移表
static cognition 静态认知
statistical model of translation 统计翻译模型
stative expression 静态体表示
stemmer 词干处理器
stemming 词干还原
stochastic 随机派
Stochastic Context-Free Grammar(SCFG) 随机上下文无关语法
stochastic paradigm 随机范型
stochastic tagger 随机标注算法
stochastic tagging 随机标注
stop 塞音
stop list 禁用词表
string distance 符号串距离
strong equivalence 强等价
structural ambiguity 结构歧义
structure 结构
subcategorization 次范畴化
subcategorization and dependencies 次范畴化和依存关系
subcategorization frame 次范畴化框架
subclass 次类
sub-dialogue 子对话
sublanguage 子语言
subordinating relation 从属关系
subordination conjunction 从属连接词
subphone 次音子
subregularity 亚规则性
substitutability 可替换性
substitution 替换
substructure 子结构
subtask 子任务
subword stage 亚词阶段
suffix 后缀
superarticulate 强势发音
supervised algorithm 有指导的算法
suprasegmental phenomena 超音段现象
surface level 表层
surface realizer 表层实现器
SVO language SVO 语言
syllabification 音节划分
synonymy 同义关系
synset 同义集
syntax 句法
syntax-driven semantic analysis 句法驱动的语义分析
system initiative 系统主动
system network 系统网络
System-Functional linguistics 系统功能语言学
systemic grammar 系统语法
table-driven 表驱动
tag 标记
tagset 标记集
Tamil 泰米尔语
tap 颤音
target cost 目标成本
target word 目标词
task completion cost 任务完成的耗费度
task completion success 任务完成的成功度
taxonomy 分类体系
template 模板
templatic filling 模板填充
template morphology 模板语素
Templatic Morphology 模板式形态学
term 检索词
term 项
term frequency 词频
terminal symbol 终极符号

Text REtrival Conference(TREC) 文本检索会议
text schemata 文本说明图
text segmentation 文本分段
text summarization 文本摘要
text-analysis 文本分析
text-based dialogue system 基于文本的对话系统
Text-To-Speech synthesis 文本 – 语音合成
textual meta-function 语篇元功能
thematic hierarchy 题元层级
thematic role 题元角色
thematic role analysis 题元角色分析
thematic roles 题元作用
theme 主位
thesaurus 类属词典
tied mixture 约束混合法
time-slice 时间片
time-synchronous beam search 时间同步的定向束搜索
tokenization 词例还原
top-down 自顶向下
trace 踪迹
trailing unit 跟踪单位读法
training corpus 训练语料库
training set 训练集
transfer 转换
transfer model 转换模型
Transformation-Based Learning(TBL) 基于转换的学习
transformation-based tagger 基于转换的标注算法
transformation-based tagging 基于转换的标注
transition 转移
transition probability 转移概率
transitive 及物动词
translation memory 翻译记忆库
translator 翻译器
transposition 换位
treebank 树库
tree-structured lexicon 树结构词表
trigram predictability 三元可预测性
triphone context 三音子上下文
truth condition 真值条件
tune 调
tune unit 调单位
Turing test 图灵测试
turn-taking 话轮转换
Turn-taking Rule 话轮转换规则
type 型
type hierarchy 类型层级体系
type underspecification 类型进行次级规范
typographical errors 打字操作错误
ungrammatical 不合语法的
unification 合一
Unification Categorial Grammar 合一范畴语法
unification operator 合一运算
union 结合
universal quqntifier 全称量词
unknown-word 未知词
unsupervised algorithm 无指导的算法
user satisfaction 用户满意度
user-centered design 以用户为中心的设计
utterance 话段
utterance time 发话时间
vagueness 模糊性
vaiables 变元
valence 配价
variable 变量
vector quantization 向量量化
vector space model 向量空间模型
velar 软腭音
verbal behavior 言语行为
verbs of throwing 投掷动词
verifiability 可能性验证
visualization 直观化
viterbi algorithm Viterbi 算法
vocabulary size 词汇容量
vocal tract 声腔
voicing 浊音化
vowel harmony 元音和谐
vowel reduction 元音弱化
vowels 元音
VSO language VSO 语言
waveform 波形

waveform concatenation 波形毗连
waveform synthesis 波形合成
weak vowel merger 弱元音合并
week equivalence 弱等价
weighted automata 加权自动机
weighted links 加权链
weighted transducer 加权转录机
well-formed 良构的
well-formedness 良构
well-formedness constraint 良构的限制
word alignment 词对齐
word association 单词联想
word error rate 词错误率
word form 词形
Word Grammar 词语法
word segmentation 单词切分
word sense 词义
word sense disambiguation 词义排歧
word sense tagging 词义标注
Words and Collocation 单词和搭配
world knowledge 世界知识
World-Wide Web 万维网
zeugma 共轭搭配法
Zipf distribution Zipf 分布