12. 成分语法

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)

语法研究有着古老的血统。帕尼尼(Panini)的梵文语法写于两千多年前，今天在教梵文时仍被引用。我们的单词**句法(syntax)**来自希腊语 sýntaxis，含义是“一起摆放或排列”，是指单词排列在一起的方式。在前面的章节中，我们已经看到了各种句法概念：单词序列的顺序(第 2 章)，这些单词序列的概率(第 3 章)以及使用词类类别作为单词的语法等价类(第 8 章))。在本章和接下来的三章中，我们介绍了远远超出这些简单方法的各种句法现象，以及用于以计算上有用的方式捕获它们的形式模型。

本章的大部分内容专用于上下文无关的语法。上下文无关语法是自然语言(以及计算机语言)句法的许多形式化模型的基础。因此，它们在许多计算应用程序中都发挥着作用，包括语法检查，语义解释，对话理解和机器翻译。它们足够强大，可以表达句子中各个单词之间的复杂关系，但在计算上也很容易处理，以至于存在有效的算法来解析句子(如第 13 章所示)。在第 16 章中，我们将展示它们如何为语义解释提供系统的框架。在这里，我们还介绍了词汇语法的概念，重点放在一个示例上，即**组合范畴语法(combinatory categorial grammar，CCG)**。

在第 14 章中，我们将介绍一个被称为**句法依存(syntactic dependencies)**的正式语法模型，它是这些成分语法的替代方案，我们将给出用于**依存解析(dependency parsing)**的算法。成分形式和依存形式对于语言处理都很重要。

最后，我们从一个叫做 ATIS(空中交通信息系统)的具有相对简单句子的领域(Hemphill 等人，1990)给出了英语语法的简要概述。ATIS 系统是一种早期的口语系统，用户可以通过表达像 I 'd like to fly to Atlanta 这样的句子来预订航班。

12.1. 成分

句法成分是指一组单词可以作为单个单元或成分。发展语法的一部分是建立语言成分的清单。英语中的单词是怎样组合在一起的?考虑名词短语(noun phrase)，它是至少包含一个名词的一系列单词。以下是一些名词短语的例子(thanks to Damon Runyon)：

Harry the Horse	a high-class spot such as Mindy's
the Broadway coppers	the reason he comes into the Hot Box
they	three parties from Brooklyn

我们有什么证据证明这些词可以组合在一起(或“构成成分”)? 有证据表明，它们都可以出现在相似的句法环境中，例如，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，虽然整个名词短语可以出现在动词之前，但组成名词短语的每个单词却不是这样。以下是不符

合语法的英语句子(记得我们用星号(*)来标记不符合语法的英语句子):

*from *arrive*. . .	*as *attracts*. . .
*the *is*. . .	*spot *sat*. . .

因此，要正确地描述这些单词在英语中的顺序，我们必须能够说“Noun Phrases can occur before verbs”这样的话。

其他类型的成分证据来自所谓的**前置(preposed)**或**后置(postposed)**构造。例如，在以下示例中，可以将 on September seventeenth 的介词短语放置在多个不同的位置，包括在开头(前置)或结尾(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，虽然整个短语可以放在不同的位置，但是组成这个短语的每个单词不能是:

*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

12.2. 上下文无关语法(CFG)

使用英语和其他自然语言对成分结构进行建模的最广泛使用的形式系统是**上下文无关语法(Context-Free Grammar，CFG)**。上下文无关的语法也称为**短语结构语法(Phrase-Structure Grammars)**[1]，而它的形式化方法等效于**巴科斯-诺尔范式(Backus-Naur-Form，BNF)**。将语法建立在成分结构上的思想可以追溯到心理学家威廉·温德(Wilhelm Wundt，1900)，但直到乔姆斯基(Chomsky，1956)和巴克(Backus，1959)才正式成立。

上下文无关的语法由一组**规则**或**产生式**，以及单词和符号的**词表(lexicon)**组成。每个规则或产生式表示可以将语言的符号进行组合和排序的方式。例如，下面的结果表示一个 **NP**(或名词短语)可以由一个专有名词或一个限定词(Det)后跟一个**名词性词(Nominal)**组成，一个名词性词可以依次由一个或多个名词组成。

$$\begin{aligned} NP &\rightarrow Det\ Nominal \\ NP &\rightarrow ProperNoun \\ Nominal &\rightarrow Noun \mid Nominal\ Noun \end{aligned}$$

上下文无关的规则可以分层次地嵌入，因此我们可以将之前的规则与其他规则结合起来，如下面所示，这些规则表达了关于词表的事实:

$$\begin{aligned} Det &\rightarrow a \\ Det &\rightarrow the \\ Noun &\rightarrow flight \end{aligned}$$

CFG 中使用的符号分为两类。与语言中单词对应的符号(“The”、“nightclub”)称为**终端(terminal)**符号;词表是一组引入这些终端符号的规则。在这些终端上表达抽象的符号称为**非终端(non-terminal)**。在每个上下文无关的规则中，箭头(→)右边的项是一个由一个或多个终端和非终端组成的有序列表;箭头左边是一个表示某些集群或泛化的非终结符号。与词表中每个单词相关联的非终端是它的词汇范畴，或词类。

---

[1] 译者注：语言结构的两种观点:成分和依存(Constituency and Dependency)结构。

- 成分=短语结构语法=上下文无关语法(CFGs)

  短语结构将单词组织成嵌套的(nested)成分。

- 依存结构显示了哪些单词依赖于哪些其他单词。

摘自《Natural Language Processing with Deep Learning》CS224N/Ling284，Christopher Manning，Lecture 4: Dependency Parsing，Stanford / Winter 2021。

可以用两种方式来理解 CFG:一种是用来生成句子的工具，另一种是用来为给定的句子分配结构的工具。将 CFG 作为生成器来查看，我们可以将“→”箭头理解为“用右边的符号串重写左边的符号”。

So starting from the symbol:	*NP*
we can use our first rule to rewrite *NP* as:	*Det Nominal*
and then rewrite *Nominal* as:	*Det Noun*
and finally rewrite these parts-of-speech as:	*a flight*

我们说 a flight 字符串可以从非终端 NP 派生。因此，CFG 可用于生成一组字符串。规则扩展的这种序列称为单词字符串的**派生(derivation)**。通常用**解析树(parse tree)**来表示派生树(通常以根倒置显示)。图 12.1 显示了该派生树表示。

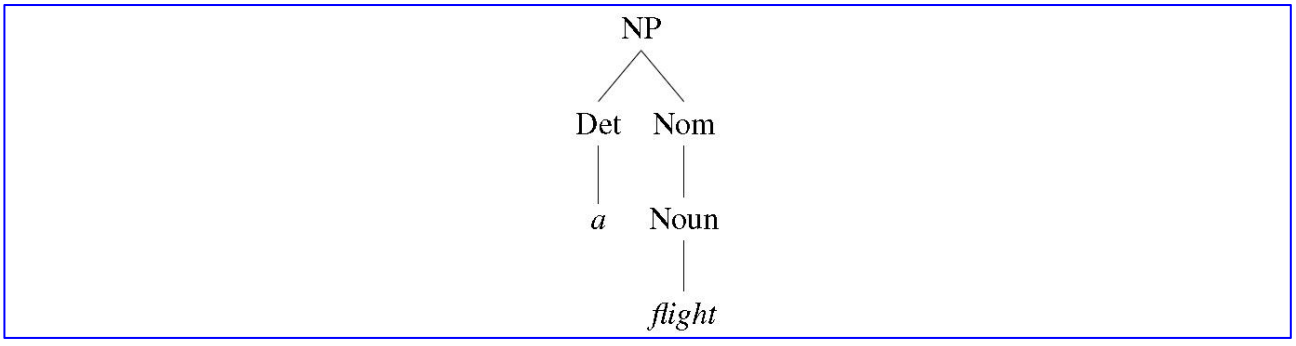


图 12-1：“A flight”的解析树

在图 12.1 所示的解析树中，我们可以说节点 NP **支配(dominate)**着树中的所有节点(Det, Nom, Noun, a, flight)。进一步说，它直接支配节点 Det 和 Nom。

CFG 定义的形式语言是从**指定的起始符号(designated start symbol)**派生的一组字符串。每个语法必须有一个指定的起始符号，通常称为 S。由于上下文无关的语法通常用于定义句子，因此 S 通常被解释为“句子”节点，从 S 派生的字符串集是英语的某些简化版本中的句子集。

让我们在清单上增加一些额外的规则。下面的规则表达了一个句子可以由一个名词短语和一个动词短语组成的事实:

S → NP VP I prefer a morning flight

英语中的动词短语是由一个动词和搭配的其他事物组成的;例如，有一种动词短语由一个动词后跟一个名词短语组成:

VP → Verb NP prefer a morning flight

或者动词后面可以跟名词短语和介词短语:

VP → Verb NP PP leave Boston in the morning

或者动词短语中可以有一个动词后跟一个介词短语:

VP → Verb PP leaving on Thursday

介词短语通常由一个介词跟一个名词短语组成。例如，在 ATIS 语料库中，一种常见的介词短语被用来表示位置或方向:

PP → Preposition NP from Los Angeles

PP 中的 NP 不一定是一个位置;PPs 经常和时间、日期连用，也可以和其他名词连用;它们可以任意复杂。以下是来自 ATIS 语料库的 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 12.2 给出了一个示例词表，图 12.3 总结了我们到目前为止看到的语法规则，我们将其称为 $\mathcal{L}_0$。请注意，我们可以使用表示“或者”的符号“|”来表示非终端符具有替代的、可能的展开方式。

我们可以使用这种语法生成这种“ATIS-language”的句子。我们从 S 开始，将它展开为 NP VP，然后随机选择 NP 的展开(假设是 I)，VP 的随机展开(假设是动词 NP)，以此类推，直到我们生成一个字符串 I prefer a morning flight。图 12.4 显示了一个表示 I prefer a morning flight 的完整派生的解析树。

我们还可以用一种更紧凑的格式表示解析树，称为**方括号表示法(bracketed notation)**；下面是图 12.4 中解析树方括号表示法:

(12.1) [S [NP [Pro I]] [VP [V prefer] [NP [Det a] [Nom [N morning] [Nom [N flight]]]]]]

类似于 $\mathcal{L}_0$ 的 CFG 定义了一种正式语言。我们在第 2 章中看到，正式语言是一组字符串。可以由语法派生的句子(单词的字符串)是由该语法定义的正式语言，称为**符合语法(grammatical)**句子。不能由给定的形式语法派生的句子不是由该语法定义的语言，被称为**不合语法的(ungrammatical)**。

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 12-2：$\mathcal{L}_0$ 的词汇表

Grammar Rules		Examples
S →	NP VP	I + want a morning flight
NP →	Pronoun	I
|	Proper-Noun	Los Angeles
|	Det Nominal	a + flight
Nominal →	Nominal Noun	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 12-3：$\mathcal{L}_0$ 的语法以及每个规则的示例短语

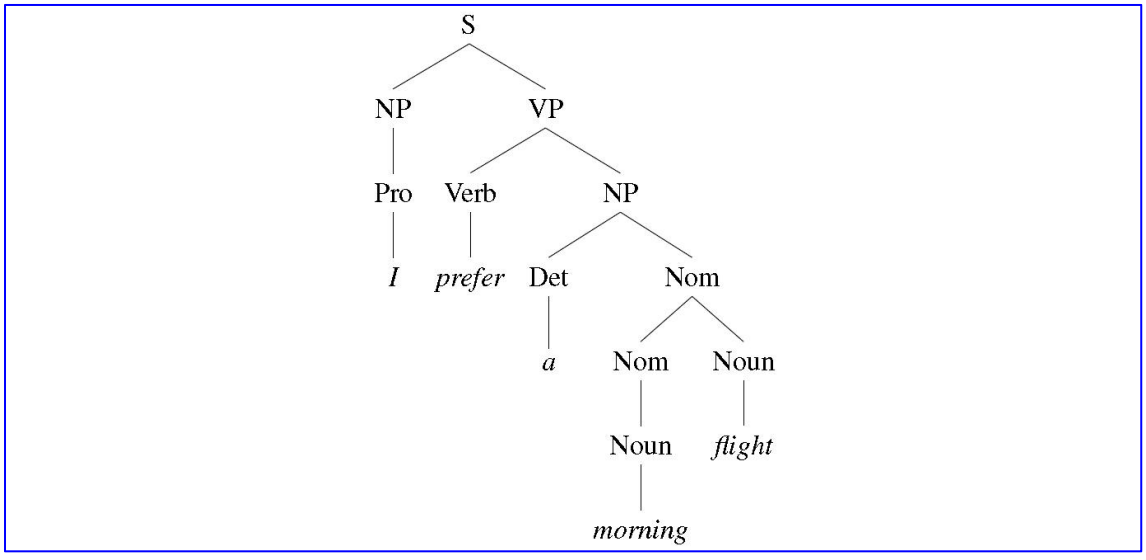


图 12-4：根据语法 $\mathcal{L}_0$ 的“I prefer a morning flight”的解析树

在“输入”和“输出”之间的强硬线（hard line）代表了所有形式语言的特征，但这只是自然语言如

何工作的非常简化的模型。这是因为确定给定句子是否属于给定自然语言(例如英语)的一部分通常取决于上下文。在语言学中，使用形式语言来对自然语言进行建模被称为**生成语法(generative grammar)**，因为该语言是由语法“生成”的一组可能的句子定义的。

12.2.1. 上下文无关语法的正式定义

我们通过对上下文无关的语法及其生成的语言的快速、正式的描述来结束本节。一个上下文无关的语法 G 由四个参数定义：N，Σ，R，S（技术上这是一个“4 元组”）。

> N 非终结符(或变量)的集合
> Σ终端符号(与N不相交)的集合
> R 规则的集合或生成式，每个生成式的形式为A→β，
> 其中：A 是非终端符号，
> β是由字符串(Σ∪N)*中无穷集合的符号构成的字符串
> S 是指定的起始符号，并且是N的成员

在本书的其余部分中，我们在讨论上下文无关语法的形式属性时(与解释关于英语或其他语言的特定事实相反)遵循以下约定：

诸如A、B和S的大写字母	非终端符号
S	起始符号
诸如α、β和γ的小写希腊字母	从(Σ∪N)*推导的符号串
诸如u、v和w的小写罗马字母	终端符号串

语言是通过派生的概念定义的。如果可以通过一系列规则应用程序将一个字符串重写为第二个字符串，则它会派生另一个字符串。根据 Hopcroft 和 Ullman(1979)，可以更正式地说：

> 如果A→β是R的一个生成式，α和γ是(Σ∪N)*集合中的任意字符串，
> 则我们就说，αAγ直接派生出αβγ，或αAγ ⇒ αβγ。

那么派生就是直接派生的概括：

令α_1，α_2，...，α_m为$(\Sigma \cup N)^*$，m≥1 中的字符串，使得

$$\alpha_1 \Rightarrow \alpha_2，\ \alpha_2 \Rightarrow \alpha_3，\ \ldots，\ \alpha_{m-1} \Rightarrow \alpha_m$$

我们说，α_1派生出α_m，或$\alpha_1 \overset{*}{\Rightarrow} \alpha_m$。

然后，我们可以正式地将语法 G 生成的语言 $\mathcal{L}_G$ 定义为一组由可以从指定的开始符号 S 派生的终端符号组成的字符串集合。

$$\mathcal{L}_G = \{w | w \text{ is in } \Sigma^* \text{ and } S \overset{*}{\Rightarrow} w\}$$

将一串单词映射到它的解析树的问题称为**句法解析(syntactic parsing)**，我们在第 13 章中定义了成分解析的算法。

12.3. 一些英语语法规则

在这一节中，我们将介绍英语短语结构的几个方面;为了保持一致性，我们将继续关注来自 ATIS 领域的句子。由于篇幅有限，我们的讨论必然局限于重点内容。强烈建议读者参考一本好的英语语法参考书，如 Huddleston 和 Pullum(2002)。

12.3.1. 句级结构

在小语法 $\mathcal{L}_0$ 中，我们只为陈述性句子提供了一个句子级的构造，就像 I prefer a morning flight 一样。在大量的英语句子构造中，特别常见和重要的句子有**四个：陈述式(declaratives)**、**命令式(imperatives)**、

是非疑问式(yes-no questions)和 **wh-疑问式(wh-questions)**。

陈述式(declaratives)结构的句子有一个主语名词短语，后面跟着一个动词短语，比如“I prefer a morning flight”。这种结构的句子有很多不同的用法，我们将在第 24 章继续讨论。下面是一些来自 ATIS 领域的例子:

I want a flight from Ontario to Chicago
The flight should be eleven a.m. tomorrow
The return flight should leave at around seven p.m.

命令式(imperatives)结构的句子通常以动词短语开头，没有主语。之所以称为命令式命令，是因为它们几乎总是用于命令和建议。在 ATIS 域中，它们是对系统的命令。

Show the lowest fare
Give me Sunday's flights arriving in Las Vegas from New York City
List all flights between five and seven p.m.

我们可以用 S 展开的另一条规则来模拟这个句子结构:

S→VP

是非疑问式(yes-no questions)结构的句子经常(虽然不总是)用来问问题;它们以助动词开头，然后是主语 NP，最后是 VP。下面是一些例子。注意，第三个例子根本不是一个问题，而是一个请求;第 24 章讨论了这些疑问形式的使用，以执行不同的**语用(pragmatic)**功能，如询问、请求、或建议。

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?

这是规则：

$$S \rightarrow Aux\ NP\ VP$$

wh-疑问式(wh-questions)

我们在这里研究的最复杂的句子层次结构是各种各样的 wh 结构。它们之所以这样命名，是因为它们的成分之一是 **wh-短语(wh-phrase)**，也就是说，它包含了 wh-单词(who, whose, when, where, what, which, how, why)。这可以大致分为两类句子层次结构。

第一，**“wh-主语疑问式(wh-subject-question)”**结构与陈述式结构相同，只是第一个名词短语中包含了一些“wh”字。

What airlines fly from Burbank to Denver?
Which flights depart Burbank after noon and arrive in Denver by six p.m?
Whose flights serve breakfast?

这里有一条规则。练习 12.7 讨论组成 Wh-NP 的成分规则。

$$S \rightarrow Wh\text{-}NP\ VP$$

第二，在 **wh-非主语疑问式(wh-non-subject-question)**结构中，wh-短语不是句子的主语，因此句子包含了另一个主语。在这些类型的句子中，助动词出现在主语 NP 之前，就像在是非疑问式结构中一样。下面是一个带有示例规则的示例:

What flights do you have from Burbank to Tacoma Washington?

$$S \rightarrow Wh\text{-}NP\ Aux\ NP\ VP$$

像 wh-非主语疑问式的 Wh-NP 这样的结构包含了所谓的**长距离依存关系(long-distance dependencies)**，因为 Wh-NP“what flight”与 VP 中主要动词“have”在语义上相关的谓词相距甚远。在与上述语法规则兼容的某些解析和理解模型中，长距离依存关系(如 flights 和 have 之间的关系)被视为语义关系。在这样的模型中，弄清楚 flights 是 have 的论元的工作是在语义解释期间完成的。其他解析模型将 flights 和 have 之间的关系表示为语法关系，语法被修改为在动词后面插入一个称为踪迹(trace)或空范畴(empty category)的小标记。我们将在 12.4.1 章节中介绍宾州树库时讨论空范畴模型。

12.3.2. 从句和句子

在我们继续之前，我们应该澄清我们刚才描述的语法中 S 规则的地位。S 规则旨在解释作为话语基本单位的整个句子。然而，S 也可以出现在语法规则的右侧，因此可以嵌入到更大的句子中。很明显，S 不仅仅是作为一个单独的话语单位存在。句子结构(即 S 规则)与其他语法的区别在于它们在某种意义上是**完整的(complete)**。这样，它们就相当于一个从句的概念，而传统语法常常把从句描述为形成一个**完整的思想(complete thought)**。让“完整思想”这个概念更精确的一种方法是说 S 是一个解析树的节点，S 的主要动词在它下面有它所有的**论元(arguments)**。我们稍后再定义论元，但现在让我们看看图 12.4 中“I prefer a morning flight”的图解。动词 prefer 有两个论元：主语 I 和宾语 a morning flight。一个论元出现在 VP 节点下，但是另一个论元(主语 NP)只出现在 S 节点下。

12.3.3. 名词短语

我们的 $\mathcal{L}_0$ 语法介绍了英语中最常见的三种名词短语类型：代词、专有名词、NP→Det Nominal 结构。本节的重点是最后一种类型，因为这是大部分语法复杂性所在的地方。这些名词短语由一个**头部**（名词短语中的**中心词**）以及各种修饰语组成，这些修饰语可以出现在头部名词之前或之后。让我们仔细看看各个部分。

限定词

名词短语可以以简单的词汇限定词开头:

a stop　　the flights　　this flight
those flights　any flights　　some flights

限定词的作用也可以由更复杂的表达式来填充:

United’s flight
United’s pilot’s union
Denver’s mayor’s mother’s canceled flight

在这些例子中，限定词的作用由一个名词短语和一个 ’s 作为所有格标记组成的所有格表达来填充，如下面的规则所示。

$$Det \rightarrow NP's$$

这个规则是递归的(因为 NP 可以从 Det 开始)，这一事实帮助我们为上面的最后两个例子建模，在这两个例子中，一系列所有格表达式充当限定词。

在某些情况下，英语中限定词是可选的。例如，如果限定词修饰的名词是复数，则可以省略:

(12.2)　Show me *flights* from San Francisco to Denver on weekdays

正如我们在第 8 章中看到的，大量名词也不需要确定。记住，物质(mass)名词通常(不总是)涉及到被当作实体的东西(例如，water 和 snow)，不要使用不定冠词“a”，也不要做复数形式。许多抽象名词是集合名词(例如，music 和 homework)。ATIS 领域中的集合名词包括 breakfast, lunch 和 dinner:

(12.3)　Does this flight serve dinner?

名词性词(Nominal)

名词性词紧跟限定词，包含任何前、后中心名词修饰语。正如语法 $\mathcal{L}_0$ 中所示，在最简单的形式中，**名词性词**可以由一个名词组成。

$$Nominal \rightarrow Noun$$

正如我们将看到的，该规则还为各种递归规则的底层提供了基础，用于捕获更复杂的名词性词结构。

中心名词之前

在一个**名词性词**之中，许多不同种类的词类可以出现在中心名词之前，但在限定词(后限定词)之后。

这些词包括**基数词(cardinal numbers)**、**序数词(ordinal numbers)**、**数量修饰语(quantifiers)**和形容词。

基数的例子:

two friends one stop

序数词包括 first、second、third 等，也包括 next、last、past、other 和 another:

the first one the second leg the next day the last flight the other American flight

一些数量修饰语(many， (a) few, several)只出现在复数可数名词中:

many fares

形容词出现在量词之后，名词之前。

a *first-class* fare a *non-stop* flight

the *longest* layover the *earliest* lunch fligh

形容词也可以组合成一个短语，称为**形容词短语(adjective phrase，AP)**。AP 可以在形容词前加一个副词(形容词的定义见第 8 章形容词和副词):

中心名词之后

中心名词后面可以接后修饰语。英语中常见的**名词性词**后修饰语有三种:

介词短语 all flights *from Cleveland*

非限定从句 any flights *arriving after eleven a.m.*

关系从句 a flight *that serves breakfast*

它们在 ATIS 数据库中特别常见，因为它们被用来标记航班的出发地和目的地。

下面是一些介词短语后修饰语的例子，用方括号表示每个短语的边界;注意，两个或更多的 PP 可以串成一个 NP:

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]*

这是一个新的名词类成分规则来解释后名词类成分的 PP：

$$Nominal \rightarrow Nominal\ PP$$

最常见的**非限定(non-finite)**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(gerundive)**后修饰语，是因为它们由动词短语组成，该动词短语以动词的动名词(-ing)形式开头。这里有些例子：

any of those *[leaving on Thursday]*

any flights *[arriving after eleven a.m.]*

flights *[arriving within thirty minutes of each other]*

我们可以使用新的非终端 GerundVP 来定义动名词修饰语的**名词性词**，如下所示：

$$Nominal \rightarrow Nominal\ GerundVP$$

我们可以通过复制所有 VP 产生式来制定 GerundVP 成分的规则，用 GerundV 代替 V。

$$GerundVP \rightarrow GerundV\ NP$$
$$| \ GerundV\ PP \mid GerundV \mid GerundV\ NP\ PP$$

GerundV 可以被定义为:

$$GerundV \rightarrow being \mid arriving \mid leaving \mid ...$$

下面用斜体印出来的短语是另外两种常见的非限定从句、不定式和-ed 形式的例子:

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(更准确地说是限制性关系从句)是以**关系代词(relative pronoun)**(that 和 who 最常用)开头的从句。在下面的例子中，关系代词作内嵌动词的主语:

a flight *that serves breakfast*

flights *that leave in the morning*

the one *that leaves at ten thirty five*

我们可以添加如下规则来处理这些问题:

Nominal → Nominal RelClause

RelClause → (who | that) VP

关系代词也可以作为内嵌动词的宾语，如下面的例子;我们把编写这类更复杂的关系从句的语法规则的练习留给读者去做。

the earliest American Airlines flight that I can get

各种名词后修饰语可以组合在一起:

a flight *[from Phoenix to Detroit] [leaving Monday evening]*

evening flights *[from Nashville to Houston] [that serve dinner]*

a friend *[living in Denver] [that would like to visit me in DC]*

名词短语之前

修饰和出现在 NP 之前的词类称为前限定词，很多都与数字或数量有关；一个常见的前限定词是 all:

all the flights　　all flights　　all non-stop flights

图 12.5 中给出的名词短语的例子说明了当这些规则组合在一起时所产生的一些复杂性。

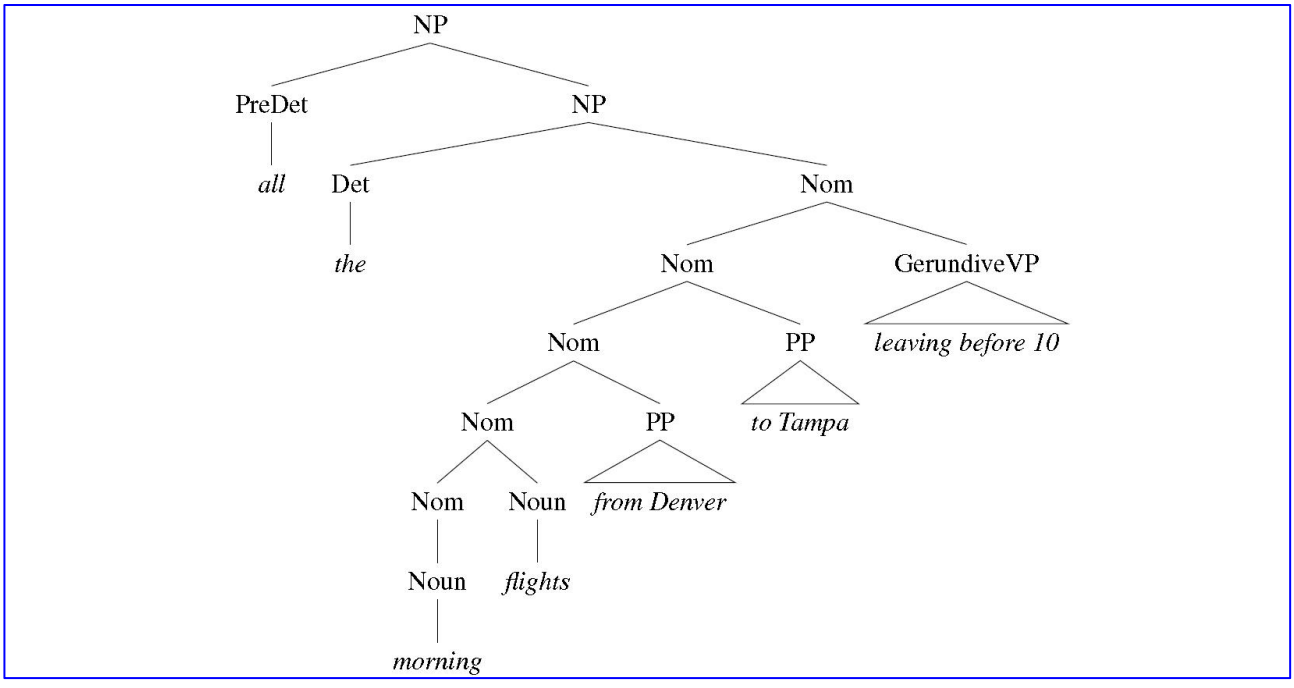


图 12-5：句子解析树的例子

图注：句子“all the morning flights from Denver to Tampa leaving before 10”的解析树。

12.3.4. 动词短语

动词短语由动词和一些其他成分组成。在我们目前建立的简单规则中，其他成分包括 NPs 和 PPs 以及两者的组合:

VP → Verb	disappear
VP → Verb NP	prefer a morning flight
VP → Verb NP PP	leave Boston in the morning
VP → Verb PP	leaving on Thursday

动词短语可以比这个复杂得多。许多其他成分，如整个内嵌的句子，可以跟在动词后面。这些被称为

句子补语(sentential complements):

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 to downtown]]

I [$_{VP}$ [$_{V}$ think [$_{S}$ I would like to take the nine thirty flight]]

这里有一条规则:

$$VP \rightarrow Verb\ S$$

类似地，VP 的另一个潜在成分是另一个 VP。像 want, would like, try, intend, need 这样的动词经常出现这种情况:

I want [$_{VP}$ to fly from Milwaukee to Orlando]

Hi, I want [$_{VP}$ to arrange three flights]

虽然一个动词短语可以有很多可能的成分，但并不是每个动词都与每个动词短语兼容。例如，动词 want 既可以与 NP 补语连用(I want a flight…)，也可以与不定式 VP 补语连用(I want fly to…)。相反，像 find 这样的动词就不能用 VP 补语(* I found to fly to Dallas)。

动词与不同种类的补语相容的观点是很古老的;传统语法区分了**及物(transitive)**动词，比如 find，它带一个直接宾语 NP(I found a flight)，和**不及物(intransitive)**动词，比如 disappear，它不带宾语(*I disappear a flight)。

传统语法将动词**次范畴化(subcategorize，再分类)**为两个范畴(及物和不及物)，而现代语法最多可将动词区分为 100 个次范畴。我们说像 find 这样的动词对 NP 进行了再分类，而像 want 这样的动词则对 NP 或非限定 VP 进行了再分类。我们也称这些成分为动词的补语(因此我们在上面使用了术语补语)。所以我们说 want 可以带一个 VP 补语。这些可能的补语集称为动词的次范畴化**框架(frame)**。讨论动词与其他成分之间关系的另一种方法是，将动词视为逻辑谓词，并将成分视为谓词的逻辑论元。因此，我们可以将谓词-论元关系视为 FIND(I，A FLIGHT)或 WANT(I，TO FLY)。当我们讨论动词语义的谓词演算表示法时，我们将在第 15 章中更多地讨论动词和论元的这种观点。图 12.6 给出了一组示例动词的次范畴化框架。

Frame	Verb	Example
$\emptyset$	eat, sleep	I ate
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]
S	mean	Does this mean [$_{S}$ AA has a hub in Boston]

图 12-6：一组实例动词的次范畴化框架

我们可以通过创建类动词的单独子类型来捕获动词及其补语之间的关联(例如：*Verb-with-NP-complement*, *Verb-with-Inf-VP-complement*, *Verb-with-S-complement*, 等等)

Verb-with-NP-complement → *find* | *leave* | *repeat* | ...

Verb-with-S-complement → *think* | *believe* | *say* | ...

Verb-with-Inf-VP-complement → *want* | *try* | *need* | ...

然后可以修改每个 VP 规则，以要求适当的动词子类型:

VP → *Verb-with-no-complement* disappear

VP → *Verb-with-NP-comp NP* prefer a morning flight

VP → *Verb-with-S-comp S* said there were two flights

这种方法的一个问题是规则数量的显著增加以及相关的通用性损失。

12.3.5. 并列关系

这里讨论的主要词组类型可以与 and、or、but 等**连词(conjunctions)**连用，形成同一类型的更大结构。例如，一个**并列(coordinate)**名词短语可以由两个由连词分隔的其他名词短语组成:

Please repeat [$_{NP}$ [$_{NP}$ the flights] *and* [$_{NP}$ the costs]]

I need to know [$_{NP}$ [$_{NP}$ the aircraft] *and* [$_{NP}$ the flight number]]

以下是允许这些结构的规则:

$$NP \rightarrow NP \text{ and } NP$$

注意，通过连词形成协调短语的能力经常被用作对成分的测试。考虑下面的例子，它们不同于上面的例子，因为它们没有第二个限定词。

Please repeat the [$_{Nom}$ [$_{Nom}$ flights] *and* [$_{Nom}$ costs]]

I need to know the [$_{Nom}$ [$_{Nom}$ aircraft] *and* [$_{Nom}$ flight number]]

这些短语可以连用的事实证明了我们一直在使用的潜在的名词性成分的存在。这里有一条规则:

$$\text{Nominal} \rightarrow \text{Nominal and Nominal}$$

下面的例子说明了 VPs 和 Ss 的连词:

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]]

VP 和 S 的连词规则与上面的 NP 规则是一致的。

$$\text{VP} \rightarrow \text{VP and VP}$$

$$\text{S} \rightarrow \text{S and S}$$

由于所有主要的短语类型都可以以这种方式连接，它也可以更普遍地表示这个连接事实;许多语法形式主义，如 GPSG (Gazdar 等人,1985 年)使用如下的**元规则(metarules)**来实现这一点:

$$X \rightarrow X \text{ and } X$$

这个元规则声明任何非终端符都可以与同一个非终端符连接，以产生相同类型的成分;变量 X 必须被指定为一个代表任何非终端符的变量，而不是非终端符本身。

12.4. 树库

由上下文无关的语法规则组成的足够健壮的语法可以用于将解析树分配给任何句子。这意味着可以构建一个语料库，其中每个句子都与相应的解析树配对。这样的句法注释语料库称为树库(treebank)。正如我们在第十三章中讨论的那样，树库在句法分析以及语言学对句法现象的研究中扮演着重要的角色。

已经创建了各种各样的树库，通常是通过使用解析器(在接下来的几章中描述的那种)自动解析每个句子，然后使用人类(语言学家)手动更正解析。**宾州树库(Penn Treebank)**项目(我们在第 8 章中介绍了它的 POS 标记集)已经从 Brown、Switchboard、ATIS 和华尔街日报的英语语汇中生成了树库，以及阿拉伯语和汉语的树库。许多树库使用我们将在第 14 章中介绍的依存表示，包括许多**通用依存**项目(Nivre 等人，2016b)的一部分。

12.4.1. 例子：宾州树库项目

图 12.7 显示了来自宾州树库 Brown 和 ATIS 部分的句子。注意词类标记的格式差异;这种微小的差异是常见的，在处理树库时必须加以处理。第八章定义了宾州树库词类标记集。对树使用 LISP 风格的圆括号表示法非常常见，类似于我们在(12.1)中看到的方括号表示法。对于那些不熟悉它的人，我们在图 12.8 中展示了一个标准的节点和线条的树表示。

图 12.9 显示了来自《华尔街日报》的树。这棵树显示了宾州树库的另一个特性:使用跟踪(trace)(-NONE-表示节点)来标记长距离依存关系或句法位移(syntactic movement)。例如，引语经常跟在引语动词后面，比如 say。但在这个例子中，“We would have to wait until we have collected on those assets”的引语出现在 he said 之前。空的 S 只包含节点-NONE-标记在 said 之后引语句子经常出现的位置。这个空节点(在

树库 II 和 III 中)被标记为索引 2，就像句子开头的引语 S 一样。这样的相互索引可能使一些解析器更容易恢复这样一个事实，即这个前面的或主题化的引语就是动词 said 的补语。类似的-NONE-节点表示在动词 to wait 之前没有句法主语;相反，主语是更前面的名词短语(NP)We。同样，它们都与索引 1 相互索引。

宾州树库 II 和树库 III 发布了新增的进一步的信息，以便更容易地恢复谓词和论元之间的关系。某些短语用标记(tag)来作标注(mark)，表明短语的语法功能(例如，表层主语、逻辑主题、强调、非 VP 谓语)，它在特定文本类别(标题、题目)中的存在，以及它的语义功能(时间短语、方位)(Marcus 等人 1994, Bies 等人 1995)。图 12.9 显示了-SBJ(表层主语)和-TMP(时间短语)标记的示例。图 12.8 还显示了-PRD 标记，它用于非 VPs 的谓语(图 12.8 中的谓语是一个 ADJP)。在第 14 章讨论依存语法和解析时，我们将回到语法功能的主题。

```
((S
  (NP-SBJ (DT That)
    (JJ cold) (, ,)
    (JJ empty) (NN sky) )
  (VP (VBD was)
    (ADJP-PRD (JJ full)
      (PP (IN of)
        (NP (NN fire)
          (CC and)
          (NN light) ))))
  (. .) ))
```

(a)

```
((S
   (NP-SBJ The/DT flight/NN )
   (VP should/MD
     (VP arrive/VB
       (PP-TMP at/IN
         (NP eleven/CD a.m/RB ))
       (NP-TMP tomorrow/NN )))))
```

(b)

图 12-7：从 LDC 树库 3 版本解析出的句子

图注：图(a)中的句子来自 Brown 语料库，图(b)中的句子来自 ATIS 语料库。

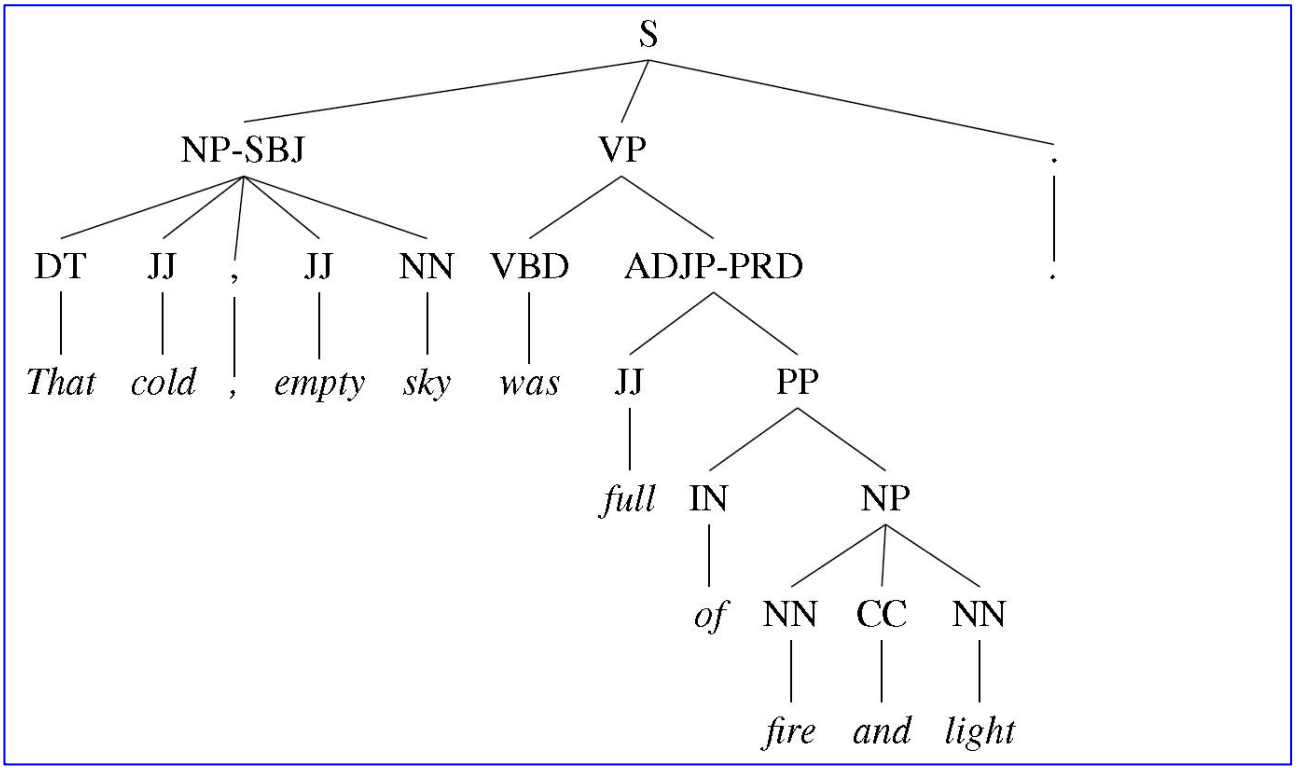


图 12-8：Brown 语料库句子对应的树形图

12.4.2. 树库作为语法

树库中的句子隐含地构成被注释的语料库所代表的语言的语法。例如，从图 12.7 和图 12.9 中三个已解析的句子中，我们可以提取出其中的每一条 CFG 规则。为简单起见，让我们去掉规则后缀(-SBJ 等)，结果生成的语法如图 12.10 所示。

```
( (S (`` ``)
    (S-TPC-2
      (NP-SBJ-1 (PRP We) )
      (VP (MD would)
        (VP (VB have)
          (S
            (NP-SBJ (-NONE- *-1) )
            (VP (TO to)
              (VP (VB wait)
                (SBAR-TMP (IN until)
                  (S
                    (NP-SBJ (PRP we) )
                    (VP (VBP have)
                      (VP (VBN collected)
                        (PP-CLR (IN on)
                          (NP (DT those)(NNS assets))))))))))))
    (, ,) ('' '')
    (NP-SBJ (PRP he) )
    (VP (VBD said)
      (S (-NONE- *T*-2) ))
    (. .) ))
```

图 12-9：来自 LDC 宾州树库的一句话

图注：摘自 LDC 宾州树库《华尔街日报》部分的一句话。请注意空的-NONE-节点的使用方法。

图注：从图 12.7 和图 12.9 中的三个树库句子中提取 CFG 语法规则和词法条目的示例。

用于解析宾州树库的语法相对平坦，从而产生非常多且非常长的规则。例如，在扩展 VPs 的大约 4500 条不同的规则中，有一些单独的规则适用于任意长度的 PP 序列和各种可能的动词论元排列:

VP → VBD PP

VP → VBD PP PP

VP → VBD PP PP PP

VP → VBD PP PP PP PP

VP → VB ADVP PP

VP → VB PP ADVP

VP → ADVP VB PP

以及更长的规则，比如：

VP → VBP PP PP PP PP PP ADVP PP

它来自于用斜体标记的 VP:

This mostly happens because we *go from football in the fall to lifting in the winter to football again in the spring*.

Grammar	Lexicon
S→NP VP .	PRP→we | he
S→NP VP	DT→the | that | those
S→“ S ” , NP VP .	JJ→cold | empty | full
S→-NONE-	NN→sky | fire | light | flight | tomorrow
NP→DT NN	NNS→assets
NP→DT NNS	CC→and
NP→NN CC NN	IN→of | at | until | on
NP→CD RB	CD→eleven
NP→DT JJ , JJ NN	RB→a.m.
NP→PRP	VB→arrive | have | wait
NP→-NONE-	VBD→was | said
VP→MD VP	VBP→have
VP→VBD ADJP	VBN→collected
VP→VBD S	MD→should | would
VP→VBN PP	TO→to
VP→VB S	
VP→VB SBAR	
VP→VBP VP	
VP→VBN PP	
VP→TO VP	
SBAR→IN S	
ADJP→JJ PP	
PP→IN NP	

图 12-10：从三个树库句子中提取 CFG 语法规则的示例

成千上万的 NP 规则包括：

NP → DT JJ NN

NP → DT JJ NNS

NP → DT JJ NN NN

NP → DT JJ JJ NN

NP → DT JJ CD NNS

NP → RB DT JJ NN NN

NP → RB DT JJ JJ NNS

NP → DT JJ JJ NNP NNS

NP → DT NNP NNP NNP NNP JJ NN

NP → DT JJ NNP CC JJ JJ NN NNS

NP → RB DT JJS NN NN SBAR

NP → DT VBG JJ NNP NNP CC NNP

NP → DT JJ NNS , NNS CC NN NNS NN

NP → DT JJ JJ VBG NN NNP NNP FW NNP

NP → NP JJ , JJ “SBAR” NNS

例如，最后两条规则来自以下两个名词短语:

[$_{DT}$ The] [$_{JJ}$ state-owned] [$_{JJ}$ industrial] [$_{VBG}$ holding] [$_{NN}$ company] [$_{NNP}$ Instituto] [$_{NNP}$ Nacional] [$_{FW}$ de] [$_{NNP}$ Industria] [$_{NP}$ Shearson’s] [$_{JJ}$ easy-to-fifilm], [$_{JJ}$ black-and-white] “[$_{SBAR}$ Where We Stand]” [$_{NNS}$ commercials]

从这种角度来看，宾州树库 III 的《华尔街日报》语料库是一个大型语法，它包含大约 100 万个单词，也有大约 100 万个非词汇规则**符记(token)**，由大约 17500 个不同的规则**类型(type)**组成。

关于树库语法的各种事实，比如大量的平面规则，给概率解析算法带来了问题。因此，对从树库中提取的语法进行各种修改是很常见的。我们在附录 C 中进一步讨论这些问题。

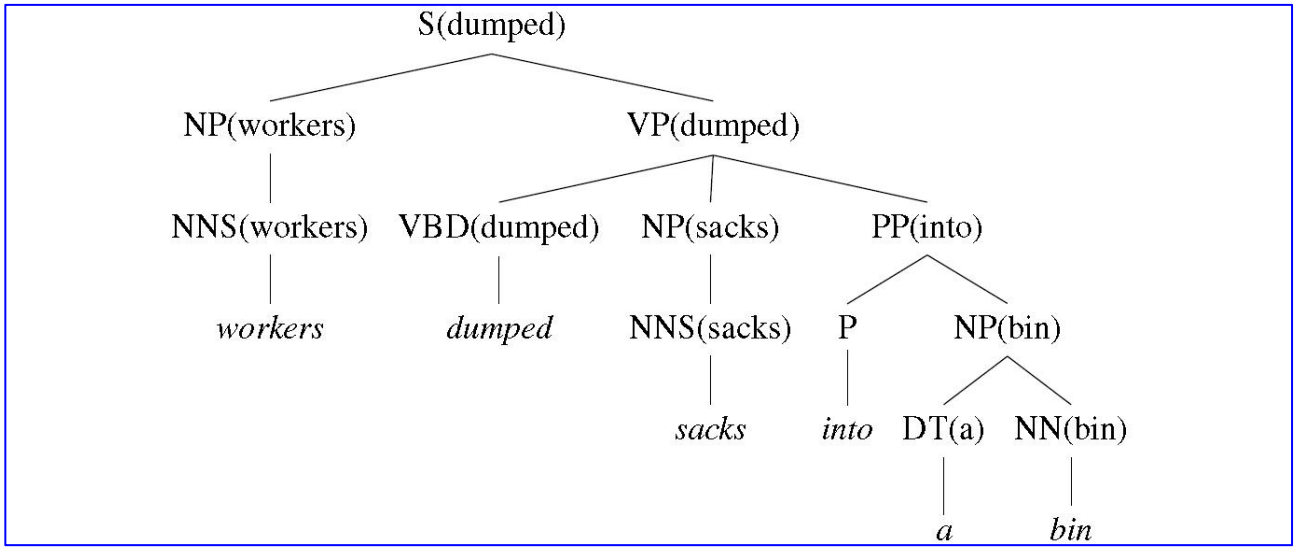


图 12-11：柯林斯(1999)的词汇树

12.4.3. 中心词及其发现

我们先前非正式地提出，语法成分可以与词汇**中心词(head)**相关联;N 是 NP 的中心词，V 是 VP 的中心词。每个成分都有一个中心词的想法可以追溯到 Bloomfield(1914 年)，这是我们将在第 14 章介绍的依存语法和依存解析的核心。中心词在概率分析(附录 C)和基于成分的语法形式中也很重要，如中心词驱动的短语结构语法(Pollard 和 Sag,1994)。

在词汇中心词的一个简单模型中，每个上下文无关的规则都与中心词相关联(Charniak 1997, Collins 1999)。短语的中心词是语法上最重要的单词。中心词被传递到解析树中;因此，解析树中的每个非终端符都用单个单词进行注释，这就是它的词汇中心词。图 12.11 显示了 Collins(1999)的这样一个树的例子，其中每个非终端符都用其中心词进行了注释。

为了生成这样的树，必须对每个 CFG 规则进行扩展，以标识右侧的一个成分作为中心词的孩子(child)。然后，节点的中心词被设置为它的中心词子节点的中心词。对于教科书上的例子来说，选择这些中心词的孩子是很简单的(NN 是 NP 的中心词)，但对于大多数短语来说是很复杂的，而且确实存在争议。(该补语或该动词应是不定式动词短语的中心词吗?)现代语言学的句法理论通常包括一个组件来定义中心词(Pollard 和 Sag, 1994))。

在大多数实际的计算系统中，都有另一种寻找中心词的方法。不是在语法本身中指定中心词规则，而是在树的上下文中为特定的句子动态识别中心词。换句话说，一旦解析了一个句子，就会遍历生成的树，用适当的中心词装饰每个节点。目前的大多数系统都依赖于一套简单的手写规则，例如一个实用的宾州树库语法，它在 Collins(1999)中给出，但最初是由 Magerman(1995)开发的。例如，找到 NP 中心词的规则如下(Collins, 1999, p. 238):

- If the last word is tagged POS, return last-word.
- Else search from right to left for the first child which is an NN, NNP, NNPS, NX, POS, or JJR.
- Else search from left to right for the first child which is an NP.
- Else search from right to left for the first child which is a $, ADJP, or PRN.
- Else search from right to left for the first child which is a CD.
- Else search from right to left for the first child which is a JJ, JJS, RB or QP.
- Else return the last word

从该集合中选取的其他规则如图 12.12 所示。例如 VP 规则为 VP→$Y_1 \cdots Y_n$，算法从 $Y_1 \cdots Y_n$ 左边开始寻找第一个类型为 TO 的 Y_i；如果没有找到 TO，则搜索 VBD 类型的第一个 Y_i；如果没有找到 VBN，它将搜索 VBN，以此类推。更多细节见 Collins(1999)。

Parent	Direction	Priority List
ADJP	Left	NNS QP NN $ ADVP JJ VBN VBG ADJP JJR NP JJS DT FW RBR RBS SBAR RB
ADVP	Right	RB RBR RBS FW ADVP TO CD JJR JJ IN NP JJS NN
PRN	Left	
PRT	Right	RP
QP	Left	$ IN NNS NN JJ RB DT CD NCD QP JJR JJS
S	Left	TO IN VP S SBAR ADJP UCP NP
SBAR	Left	WHNP WHPP WHADVP WHADJP IN DT S SQ SINV SBAR FRAG
VP	Left	TO VBD VBN MD VBZ VB VBG VBP VP ADJP NN NNS NP

图 12-12：Collins(1999)的一些中心词规则

图注：中心词规则也称为中心词渗透表(percolation table)。

12.5. 语法等价与乔姆斯基范式(CNF)

正式语言被定义为(可能是无限的)单词的字符串集合。这表明，我们可以通过询问两种语法是否生成相同的字符串集合来求(ask)它们是否等价。事实上，两个不同的上下文无关的语法可能生成同一种语言。

我们通常要区分两种语法等价:**弱等价(weak equivalence)**和**强等价(strong equivalence)**。如果两种语法生成相同的字符串集，并且为每个句子分配相同的短语结构(只允许重命名非终端符号)，那么这两种语法是强等价的；如果它们生成相同的字符串集合，但不为每个句子分配相同的短语结构，那么两个语法是弱等价的。

对于语法而言，拥有一个**范式(normal form)**通常是很有用的，范式中每个产生式都采用特定形式。例如，如果上下文无关语法是ϵ自由(ϵ-free)的，并且如果另外每个产生式均为 A→B C 或 A→a，则其形式为**乔姆斯基范式(Chomsky normal form，CNF)**(Chomsky,1963)。也就是说，每个规则的右侧要么具有两个非终端符，要么具有一个终端符。乔姆斯基范式语法是**二叉的(binary branching)**，即它们具有二叉树(向下到前词汇节点)。我们将在第 13 章的 CKY 解析算法中使用此二叉属性。

任何上下文无关的语法都可以转换为弱等价的乔姆斯基范式语法。例如,如下形式的规则：

$$A \rightarrow B\ C\ D$$

可以转换成以下两个 CNF 规则(练习 12.8 要求读者写出完整的算法):

$$A \rightarrow B\ X$$

$$X \rightarrow C\ D$$

有时使用二进制分支实际上可以产生更小的语法。例如，句子可能被描述为：

VP → VBD NP PP*

在宾州树库中由以下一系列规则表示:

VP → VBD NP PP

VP → VBD NP PP PP

VP → VBD NP PP PP PP

VP → VBD NP PP PP PP PP

...

但也可以由以下两个规则语法生成:

VP → VBD NP PP

VP → VP PP

一个生成符号 A 具有一个潜在的无限序列的符号 B，其规则是 A→A B，这种生成被称为**乔姆斯基邻接规则（Chomsky-adjunction）**。

12.6. 词汇语法

到目前为止提出的语法方法强调短语结构规则，同时使词汇的作用最小化。但是，正如我们在关于一

致性、次范畴化和长距离依赖项的讨论中所看到的那样，这种方法充其量带来了繁琐的解决方案，产生了多余、难以管理且脆弱的语法。为了克服这些问题，已经开发出了许多替代方法，它们都具有更好地利用词汇的共同主题。在更多与计算相关的方法中，包括词汇功能语法(LFG)(Bresnan，1982)，中心词驱动短语结构语法(HPSG)(Pollard 和 Sag，1994)，树连接语法(TAG)(Joshi，1985)，和组合范畴语法(CCG)。这些方法的不同之处在于它们的词汇化程度—它们依赖词汇而不是短语结构规则来捕捉语言事实的程度。

以下部分提供了 CCG 的介绍，这是一种由句法和语义考虑而激发的高度词汇化的方法，我们将在第 15 章中返回讨论。第 14 章讨论了依存语法，这是一种完全消除短语结构规则的方法。

12.6.1. 组合范畴语法(CCG)

在本节中，我们提供了范畴语法(Ajdukiewicz 1935, Bar-Hillel 1953)的概述：**组合范畴语法(combinational category grammar，CCG)** (Steedman 1996, Steedman 1989, Steedman 2000)。这是一种早期的词汇化语法模式，也是一种重要的现代扩展。

范畴方法由三个主要元素组成:一组**范畴**，一个将单词与范畴联系起来的**词表**，以及一组管理范畴如何在上下文中组合的**规则**。

范畴

范畴是原子元素或单一论元函数，当提供所需范畴作为论元时，返回论元作为值。更正式地说，我们可以定义 C，一组用于语法的范畴如下:

- $A \subseteq C$, where A is a given set of atomic elements
- $(X/Y), (X\backslash Y) \in C$, if $X, Y \in C$

这里显示的斜杠符号用于定义语法中的函数。它指定了期望论元的类型、期望找到它的方向以及结果的类型。因此，(X/Y)是一个函数，它于右侧寻找类型为 Y 的成分，并返回 X 的值;(X\Y)是相同的，除了它于左边寻求它的论元。

原子范畴的集合通常非常小，包括熟悉的元素，如句子和名词短语。功能范畴包括动词词组和复杂名词词组。

词表

词表由单词的范畴分配组成。这些分配可以是原子范畴，也可以是功能范畴，并且由于词汇模糊性，可以将单词分配给多个范畴。考虑以下示例词汇条目。

flight : *N*
Miami : *NP*
cancel : (*S\NP*)/*NP*

名词和专有名词(如 flight 和 Miami)被分配到原子范畴，反映了它们作为函数论元的典型作用。另一方面，像 cancel 这样的及物动词被赋予了范畴(S\NP)/NP:一个函数寻找它右边的 NP，并返回一个具有类型(S\NP)的函数作为它的值。反过来，这个函数可以与左边的 NP 组合，产生一个 S 作为结果。这捕获了第 12.3.4 节中讨论的次范畴化信息，但是这里的信息具有丰富的、计算上有用的内部结构。

像 give 这样的双及物动词，在动词后需要两个论元，就会有((S\NP)/NP)/NP:一个与 NP 结合的函数，产生另一个对应于及物动词(S\NP)/NP 范畴的函数，如上面给出的 cancel。

规则

范畴语法规则指定函数及其论元如何组合。以下两个规则模板构成了所有范畴语法的基础。

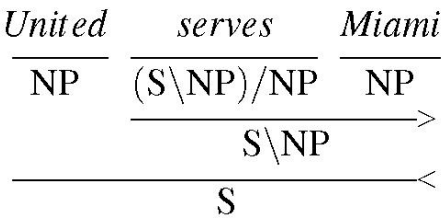
$$X/Y \quad Y \Rightarrow X \tag{12.4}$$

$$Y \quad X\backslash Y \Rightarrow X \tag{12.5}$$

第一个规则将一个函数应用到它右边的论元，而第二个规则从左边查找它的论元。我们将第一个称为前向函数应用程序，第二个称为后向函数应用程序。应用其中任何一个规则的结果都是指定为应用函数值

的类范畴。

给定这些规则和一个简单的词汇表，让我们来分析一下 United serves Miami 的句子。假设 serves 是一个具有范畴(S\NP)/NP 的及物动词，United 和 Miami 都是简单的 NP。同时使用前向和后向函数应用，推导过程如下:

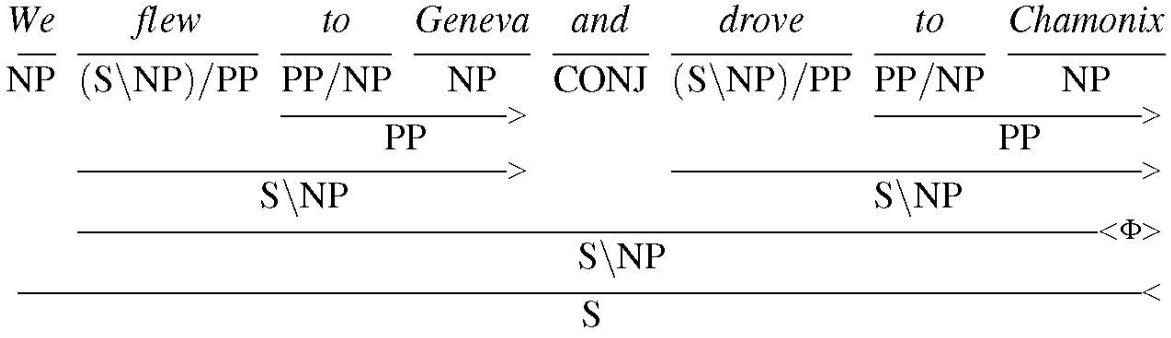


范畴语法的推动从单词开始，规则应用程序用横贯所涉及元素的水平线来说明，操作的类型在该行的右端表示。在此示例中，有两个函数应用程序：一个由>表示的前向函数应用程序，将动词 serves 应用到其右侧的 NP；一个由<表示的后向函数应用程序，将第一个结果应用于 NP United 在其左侧。

通过添加另一个规则，分类方法提供了一种直接的方式来实现前面在第 12.3.5 节描述的并列关系元规则。回想一下，英语允许同一类型的两个成分并列，从而产生同一类型的新成分。下面的规则提供了处理这类示例的机制。

$$X\ CONJ\ X \Rightarrow X \tag{12.6}$$

这条规则表明，当同一个范畴的两个成分被 CONJ 类型的成分分开时，它们可以组合成同一类型的单个更大的成分。下面的推导说明了该规则的使用。



通过连接操作符<Φ>将两个 S\NP 组合成一个较大的同类成分，再通过反向函数应用与主语 NP 组合。

这些例子说明了范畴语法方法的词汇性质。关于一种语言的语法事实很大程度上被编码在词表中，而语法规则被归结为三规则的集合。不幸的是，与传统的 CFG 规则相比，基本的分类方法并没有给我们更多的表达能力;它只是将信息从语法转移到词汇中。为了超越这些限制，CCG 可对函数进行操作。

第一对操作符是组合相邻函数。

$$X/Y\ Y/Z \Rightarrow X/Z \tag{12.7}$$

$$Y\backslash Z\ X\backslash Y \Rightarrow X\backslash Z \tag{12.8}$$

其中：第一个规则（12.7）称为**前向组合**(forward composition)规则，是一个在其右侧寻找类型为 Y 的参数的函数，可以应用于相邻的成分；第二个规则（12.8）称为**向后组合**(backward composition)规则，是一个提供 Y 作为结果的函数，该规则可以将这两个函数组合成一个函数，此函数的第一个成分为类型，第二个成分为参数。尽管这种表示有点笨拙，但第二条规则是一样的，除了我们要从左边而不是从右边查找相关的参数。CCG 图中的 B 表示这两种成分，并用"<"或">"表示方向。

第二对操作符是类型提升(type raising)。类型提升将简单的范畴提升到函数的地位。更具体地说，类型提升接受一个范畴，并将其转换为一个函数，该函数寻求一个以原始范畴作为论元的函数。下面的模式显示了两个版本的类型提升:一个用于右边的论元，一个用于左边的论元。

$$X \Rightarrow T/(T\backslash X) \tag{12.9}$$

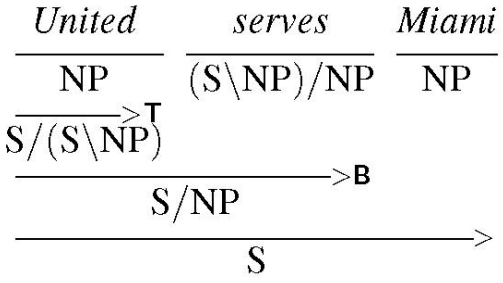
$$X \Rightarrow T\backslash(T/X) \tag{12.10}$$

这些规则中的范畴 T 可以对应于语法中已经存在的任何原子或函数范畴。

一个特别有用的类型提升示例将一个简单的主语位置的 NP 论元转换为一个可以与后面的 VP 组合的函数。为了了解这是如何运作的，让我们回顾一下之前 United serves Miami 的例子。与其将 United 分类为 NP，作为附加于 serve 的功能的一个论元，我们还不如使用类型提升，将其重新定义为一个函数，如下所示。

$$NP \Rightarrow S/(S\backslash NP)$$

将这个类型提升的成分与前向组合规则(12.7)相结合，可以实现以下替代前面推导的方法。



通过类型提升将 United 提升为 S/(S\NP)，我们可以将它与及物动词 serve 组合起来，以产生完成推导所需的(S/NP)函数。

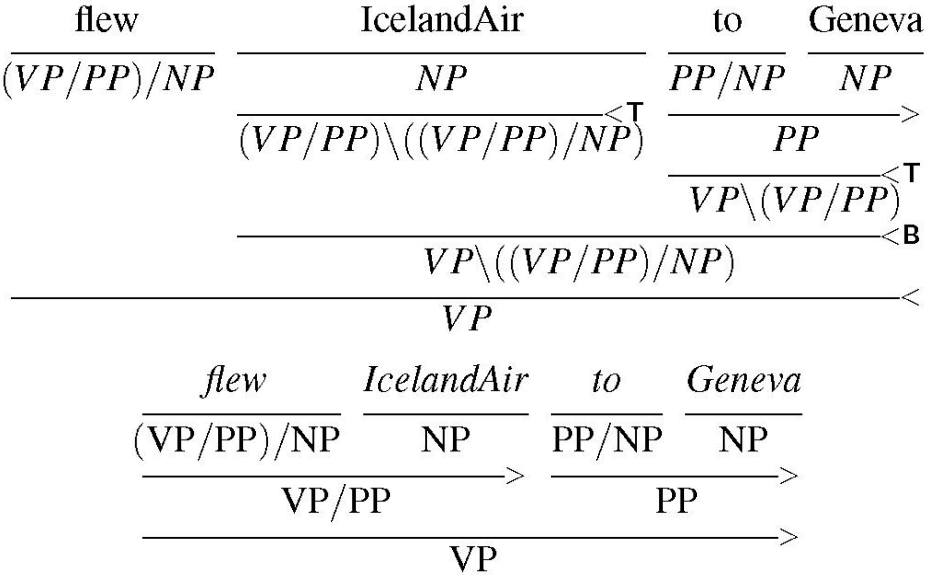
关于这个推导有几个有趣的地方需要注意。首先，它提供了从左到右逐字推导的方法，这更接近于反映人类处理语言的方式。这使得 CCG 成为一个特别适合于心理语言学研究的框架。第二，这一推导涉及到中间分析单元 United services 的使用，这与英语中的传统成分不相对应。这种使用**非成分(non-constituent)**的能力使 CCG 能够处理非适当成分的短语的并列关系，如下面的示例所示。

(12.11) We flew IcelandAir to Geneva and SwissAir to London.

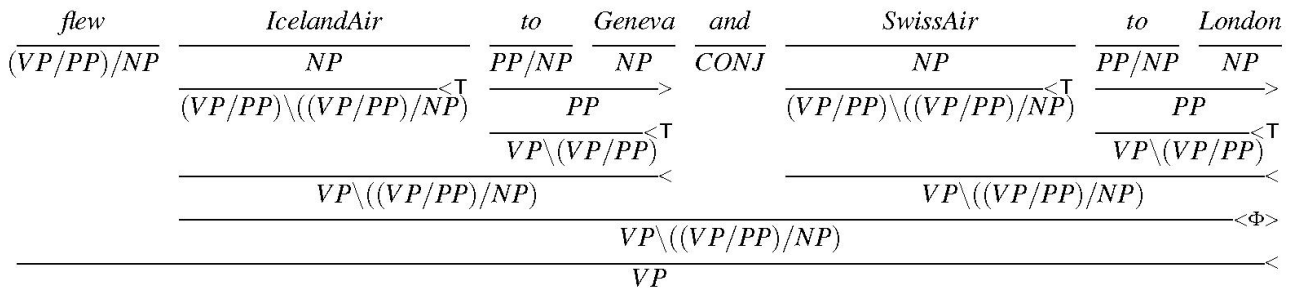
这里，正在并列的部分是 IcelandAir to Geneva 和 SwissAir to London，这些短语通常不被认为是成分，从动词短语 flew IcelandAir to Geneva 的标准推导中可以看出。

在这一推导中，没有单个成分对应于 IcelandAir to Geneva，因此没有机会使用<Φ>操作符。注意，复杂的 CCG 范畴可能会有点麻烦，因此我们将在本文和以下推导中使用 VP 作为(S\NP)的简写。

下面的替代推导通过使用后向类型提升(12.10)和后向函数组合(12.8)来提供所需的元素。



通过对 SwissAir to London 应用相同的分析，满足了<Φ>操作符的需求，产生了原始示例(12.11)的如下派生。

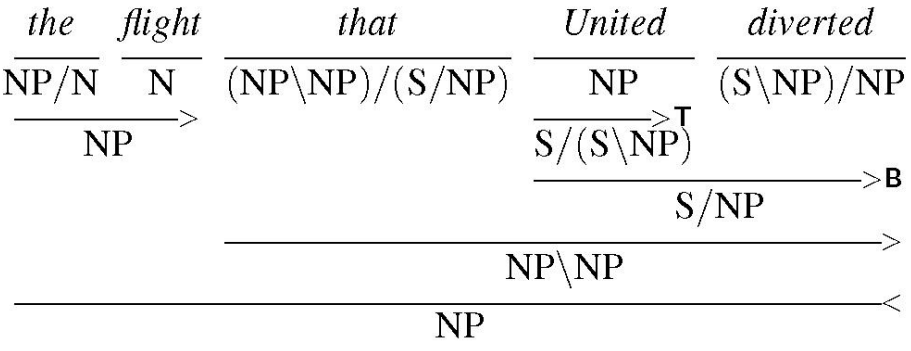


最后，让我们看看如何使用这些高级操作符来处理长距离依存关系(也称为句法位移或提取)。正如第 12.3.1 节所提到的，远距离依存关系产生于许多英语结构，包括 wh-疑问句、关系从句和话题化(topicalization)。这些结构的共同之处是出现在与其通常的或预期的位置不同的地方的成分。以下面的关系从句为例。

the flight that United diverted

其中，divert 是一个及物动词，它有两个 NP 论元，主语 NP 在左，直接宾语 NP 在右；因此它的范畴是(S\NP)/NP。然而，在这个例子中，直接宾语 the flight 已经被“移动”到从句的开头，而主语 United 保持在它的正常位置。我们需要的是一种方法来合并主语论元，同时处理 the flight 不是在其预期位置的事实。

下面的推导实现了这一点，同样是通过结合使用类型提升和函数组合。



正如我们在前面的例子中所看到的，这个推导的第一步是将 United 的类型提升到 S/(S\NP)范畴，允许它把 diverted 与前向组合联合起来。这种组合的结果是 S/NP，它保留了一个事实，即我们仍然在寻找一个 NP 来填补缺失的直接对象。第二个关键部分是词汇范畴被分配给单词 that：(NP\NP)/(S/NP)。此函数查找缺少其右侧论元的动词短语，并将其转换为 NP，以查找左侧缺少元素的 NP，恰好是我们找到 the flight 的位置。

CCG 树库

与短语结构方法一样，树库在基于 CCG 的解析方法中也起着重要作用。CCGBank(Hockenmaier 和 Steedman，2007 年)是最大、使用最广泛的 CCG 树库。它是通过基于规则的方法自动翻译宾州树库中的短语结构树而创建的。该方法成功翻译了宾州树库中超过 99%的树，从而产生 48,934 个句子与 CCG 派生词配对。它还提供了 1200 多个范畴的 44000 个单词的词汇。附录 C 将讨论如何使用这些资源来训练 CCG 解析器。

12.7. 总结

本章通过使用**上下文无关语法**介绍了一些句法上的基本概念。

•在许多语言中，连续单词的集合充当一个组或**成分**，可以用**上下文无关语法**(也称为**短语结构语法**)来建模。

•上下文无关的语法由一组(groups)**规则**或**产生式**组成，用一组**非终端**符和一组**终端**符来表达。形式上，特定的上下文无关语言是可以从特定的上下文无关语法**派生**的字符串集合。

•**生成式语法**是语言学中形式语言的传统名称，用于建模自然语言的语法。

•英语中有许多句子级语法结构;**陈述式、命令式、是非疑问式和 wh-疑问式**是四种常见的类型;这些可以用上下文无关的规则来建模。

•一个英语名词短语的**中心名词**前面可以有**限定词、数字、数量修饰语**和**形容词短语**，中心名词后面可以有一些**后修饰语**;**动名词**和**不定式** VP 是常见的可能性。

•英语中**主语**在人称和数方面与主谓动词**一致**。

•动词可以根据其期望的**补语**类型**次范畴化**(再分类)。简单的次范畴有**及物性**和**不及物性**;大多数语法包括比这些更多的类范畴。

•**树库**的分析句子存在于许多类型的英语和许多语言。可以用树搜索工具搜索树库。

•任何上下文无关的语法都可以转换为**乔姆斯基范式（CNF）**，即每个规则的右边都有两个非终端符或一个终端符。

•词汇语法更加重视词表的结构，减轻了纯短语结构规则的负担。

•**组合范畴语法(CCG)**是一种重要的与计算相关的词汇化方法。

12.8. 文献和历史说明

根据 Percival(1976)的说法，将开创性的句子分解为成分的层次结构的思想出现在开创性的心理学家威廉·温特(Wiltt Wundt)的沃克心理学中(Wundt，1900)：

> *...den sprachlichen Ausdruck für die willkürliche Gliederung einer Gesammtvorstellung in ihre in logische Beziehung zueinander gesetzten Bestandteile*
>
> (将一个整体思想任意划分为相互之间具有逻辑关系的成分的语言表达)

Leonard Bloomfield 在其早期著作《语言研究导论》(Bloomfield，1914)中将 Wundt 的成分概念纳入了语言学。到他后来的著作《语言》(Bloomfield，1933)出版时，当时称为“**直接成分分析（immediate-constituent analysis，ICA）**”的语法已成为美国公认的句法研究方法。相比之下，传统的欧洲语法可以追溯到古典时期，它定义了单词之间的关系，而不是成分之间的关系，欧洲句法学家则将重点放在这种依存语法上，这是第 14 章的主题。

美国结构主义在寻找“发现程序”的过程中看到了直接成分的许多具体定义，这是一种描述语言句法的方法论算法。通常，这些尝试试图抓住这样的直觉，即“直接成分的主要标准是等级(degree)，其中把组合行为（combinations behave）当作简单的单元”(Bazell 1966，第 284 页)。在最具体的定义中，最著名的是哈里斯(Harris)的关于与单个单元的分布相似性的想法，并带有可替代性测试。本质上，该方法通过尝试用简单的结构替换可能的成分来将结构分解为成分，如果一个简单的形式(例如人)的替换可以替换为更复杂的集合的结构(例如紧张的年轻人)，那么紧张的年轻人很可能是一个成分。哈里斯的测试是直觉的开始，即成分是一种等价类。

这种层级成分思想的第一个形式化是 Chomsky(1956)中定义的**短语结构语法**，并在 Chomsky(1957)和 Chomsky(1975)中进一步扩展(并反对)。从那时起，大多数生成式语言学理论至少部分地基于上下文无

关语法或它们的概括(例如，中心词驱动短语结构语法(Pollard 和 Sag，1994)，词汇功能语法(Bresnan，1982) ，极简主义计划(Chomsky，1995 年)和建构语法(Kay 和 Fillmore，1999 年)，等等)； 这些理论中有许多使用了称为 **X-bar 图式(x-bar schemata)**的上下文无关逻辑示意图模板，该模板也依赖于句法中心词的概念。

在乔姆斯基的最初工作后不久，上下文无关语法被 Backus(1959)和 Naur 等人(1960)在他们对 ALGOL 编程语言的描述中独立地重新发明；Backus(1996)指出他受到了 Emil Post 作品的影响，Naur 的作品独立于他(Backus)自己的作品。在这些早期工作之后，大量的自然语言处理的计算模型都是基于上下文无关的语法，因为早期开发了高效的算法来解析这些语法(见第 13 章)。

有各种各样的 CFGs 扩展类，其中许多用于处理句法中的长距离依存关系。(其他语法将远程依赖项视为语义相关，而不是语法相关(Kay 和 Fillmore 1999, Culicover 和 Jackendoff 2005))。

一种扩展的形式主义是树邻接语法(TAG)(Joshi，1985 年)。TAG 的主要数据结构是树，而不是规则。树有两种：初始树和辅助树。例如，初始树可能表示简单的句子结构，而辅助树则将递归添加到树中。树由替换和附加两种操作组合而成。附加操作处理长距离依赖性。有关更多详细信息，请参见 Joshi(1985)。树邻接语法是上下文敏感语言家族的成员。

我们在第 12.4 节提到了另一种处理远程依存关系的方法，这种方法基于使用空类别(empty categories)和共索引(co-indexing)。宾州树库使用了这个模型，它(在各种 Treebank 语库中)借鉴了扩展标准理论和极简主义(Radford, 1997)。

对英语语法感兴趣的读者应该获得三大参考英语语法之一:Huddleston 和 Pullum (2002)， Biber 等人(1999)， Quirk 等人(1985)。

有许多从不同角度介绍句法的好的入门教科书。Sag 等人(2003)从**生成式**的角度对句法进行了介绍，重点研究了中心词驱动型短语结构语法中短语结构规则、统一和类型层次的使用。Van Valin, Jr.和 La Polla(1997)从**功能式**角度介绍了跨语言数据和句法结构的功能动机。

12.9. 练习

12.1 Draw tree structures for the following ATIS phrases:

1. Dallas
2. from Denver
3. after five p.m.
4. arriving in Washington
5. early flights
6. all redeye flights
7. on Thursday
8. a one-way fare
9. any delays in Denver

12.2 Draw tree structures for the following ATIS sentences:

1. Does American Airlines have a flight between five a.m. and six a.m.?
2. I would like to fly on American Airlines.
3. Please repeat that.
4. Does American 487 have a first-class section?
5. I need to fly between Philadelphia and Atlanta.
6. What is the fare from Atlanta to Denver?
7. Is there an American Airlines flight from Philadelphia to Dallas?

12.3 Assume a grammar that has many *VP* rules for different subcategorizations, as expressed in Section 12.3.4, and differently subcategorized verb rules like *Verb-with-NP-complement*. How would the rule for ostnominal relative clauses (12.4) need to be modified if we wanted to deal properly with examples like *the earliest flight that you have*? Recall that in such examples the pronoun *that* is the object of the verb *get*. Your rules should allow this noun phrase but should correctly rule out the ungrammatical *S *I get*.

12.4 Does your solution to the previous problem correctly model the NP *the earliest flight that I can get*? How about *the earliest flight that I think my mother wants me to book for her*? Hint: this phenomenon is called **long-distance dependency**.

12.5 Write rules expressing the verbal subcategory of English auxiliaries; for example, you might have a rule *verb-with-bare-stem-VP-complement → can*.

12.6 *NP*s like *Fortune ' s office* or *my uncle ' s marks* are called **possessive** or **genitive** noun phrases. We can model possessive noun phrases by treating the sub-NP like *Fortune ' s* or *my uncle ' s* as a determiner of the following head noun. Write grammar rules for English possessives. You may treat *' s* as if it were a separate word (i.e., as if there were always a space before *' s*).

12.7 Page 238 discussed the need for a *Wh-NP* constituent. The simplest *Wh-NP* is one of the *Wh-pronouns* (*who, whom, whose, which*). The Wh-words *what* and *which* can be determiners: *which four will you have?*, *what credit do you have with the Duke?* Write rules for the different types of *Wh-NP*s.

12.8 Write an algorithm for converting an arbitrary context-free grammar into Chomsky normal form.