华中科技大学

编译原理

第2章　文法和语言

2019年11月4日星期一

编译原理课程组

内容摘要

本章介绍形式语言理论的基本内容，重点讨论文法和语言的概念、上下无关文法及其句型分析的基本问题。

形式语言与自动机理论是计算机科学的基础理论之一，是构造词法分析程序和语法程序的理论依据，也是学习编译原理的核心和基础内容。

关于自动机理论基本内容将在第三章讨论。

内容摘要

本章介绍形式语言理论的基本内容，重点讨论文法和语言的概念、上下无关文法及其句型分析的基本问题。

形式语言与自动机理论是计算机科学的基础理论之一，是构造词法分析程序和语法程序的理论依据，也是学习编译原理的核心和基础内容。

关于自动机理论基本内容将在第三章讨论。

重点讲解

2.1　文法的直观概念

2.2　符号和符号串

2.3　文法和语言的形式定义

2.4　文法类型

2.5　上下文无关文法及其语法树

2.6　句型分析

2.7　文法在实用中的一些说明

2.1 文法的直观概念

从语言结构的角度看，组成语言的基本形式是句子，句子是由单词序列构成的，单词是由语言基本符号（字母或单字）组成的。

语言既包含单词和句子这样的语言成分，又包含将这些成分组织起来的语言规则，如词法规则、句法规则等。

下面以自然语言为例，说明如何对语言规则进行形式化描述的基本思路。

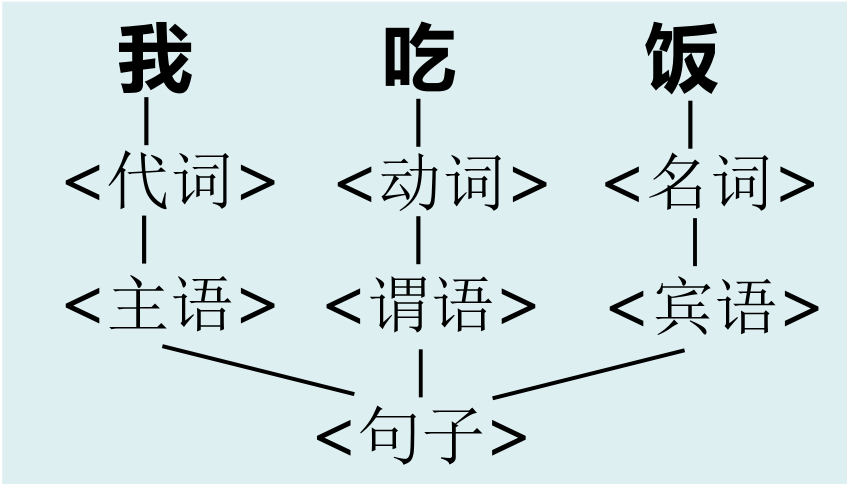
语言

- 语法：是一组规则，定义符号如何排列，排列与符号含义无关。
- 语句：简单句：主谓宾定状；复合句：条件、转折、虚拟。。。
- 语义 ：研究语法的含义
  - 静态语义
  - 动态语义

文法是阐述语法的一个工具，语句是语法的实例

2.1　文法的直观概念

考虑句子：“我吃饭”。它的语法成分构成如下，显然是符合中文语法规则—简单句有主谓宾三部分组成。



中文语法规则可以采用如下方式陈述：

<句子>是由<主语> <谓语> <宾语>组成

<主语>是由<名词>组成

… … …

2.1 文法的直观概念

约定(1)：符号∷=的意义如下，

"… ∷= …"表示 "…是由…组成的"。

语法规则可以形式化描述如下：

<句子> ∷=<主语> <谓语> <宾语>
<主语> ∷=<名词>
<主语> ∷=<代词>
<谓语> ∷=<动词>
<宾语> ∷=<名词>
<宾语> ∷=<代词>
<代词> ∷= 我
<代词> ∷= 你
<动词> ∷= 吃
<动词> ∷= 做
<名词> ∷= 饭
<名词> ∷= 菜

注解：

语法概念加上< … >表示是可替换部分，目的是与构成语句的"单词"加以区别。

2.1 文法的直观概念

约定(2)：符号 | 的表示“或者”的意义。

语法规则可以简化描述如下：

<句子> ::= <主语> <谓语> <宾语>
<主语> ::= <名词> | <代词>
<谓语> ::= <动词>
<宾语> ::= <名词> | <代词>
<代词> ::= 我 | 你
<动词> ::= 吃 | 做
<名词> ::= 饭 | 菜

注解： <主语> ::= <名词> | <代词>是两个规则的简写，仍然表示两个规则。

2.1 文法的直观概念

约定(3)：符号⇒表示“推导”。

依据某一个规则，将被替换的内容中出现的某一个与该规则左部相同的部分，用该规则右部替换形成另一个新内容，这个过程叫做一步“推导”。

使用推导符号⇒，一步推导可以描述成：**原内容⇒新内容**

采用“多步推导过程”，可以表示语法分析过程。

<句子> ⇒ <主语> <谓语> <宾语>

⇒ <代词> <谓语> <宾语>

⇒ 我 <谓语> <宾语>

⇒ 我<动词> <宾语>

⇒ 我吃<宾语>

⇒ 我吃<名词>

⇒ 我吃饭

注解：

①推导过程不唯一

②推导起点的不同，导致语法意义上差异的推导结果

2.1　文法的直观概念

假设将“语法概念”用词<句子>、<主语>、<谓语>、<宾语>、<代词>、<动词>和<名词>，分别用A、B、C、D、E、F和G表示，“单词”本身：我、你、吃、做、饭和菜，分别用a、b、c、d、e和f表示，则句子、规则和推导过程，可以进一步抽象而形式化。

规则：	推导过程：
$A ::= BCD$	$A \Rightarrow BCD$
$B ::= G \mid E$	$\Rightarrow ECD$
$C ::= F$	$\Rightarrow aCD$
$D ::= G \mid E$	$\Rightarrow aFD$
$E ::= a\ b$	$\Rightarrow acD$
$F ::= c\ d$	$\Rightarrow acG$
$G ::= e \mid f$	$\Rightarrow ace$

2.1 文法的直观概念

语法形式化方法要点：

- 语法规则的形式化
- 语法规则含有语法单位符号
- 语法规则含有构成语句的单词符号
- 特殊的语法单位符号——开始符号[注]

综上所述，语法形式化的最终目的在于将语法分析的问题将转换成形式化的推导过程。

[注]：从不同的符号开始推导，将得到完全不同语法意义上的结果。

2.2　符号和符号串

2.2.1　基本概念

- **字母表**　字母表Σ是非空有穷集合，其元素称为符号。
- **符号串**　由字母表Σ中的符号组成的有穷序列称为（字母表Σ上的）符号串。特别地，不含任何符号的有穷序列称为空串，记为ε。单词和源程序都是符号串！

例：

设字母表Σ＝{0，1}，则

101是Σ上的符号串，

201不是Σ上的符号串。

2.2.1　基本概念

- **符号串长度** 符号串α的长度是指符号串α中含有符号的个数，记为|α|。特别约定，空串ε为零，即|α|=0。

- **符号串集合** 如果集合A的元素都是字母表∑上的符号串，则称集合A为∑上的符号串集合，简称串集。

例:

设字母表∑＝{a，b，c}，A＝{ε，a，ba，cab}，B＝{a1，ba，cab}，则

A是∑上的符号串集合，

B不是∑上的符号串集合。

2.2.2 基本运算

· **符号串连接运算** 设x和y是字母表Σ上的符号串，在符号串x的最后一个符号之后顺序接上符号串y的符号得到的新符号串z，则称符号串z是由符号串x和符号串y经过连接运算的结果，记为z＝x·y，其中，·是连接运算符。

例：设字母表Σ＝{a，b，c，0，1}，

x＝abc，y＝01cba，

则 z＝x·y＝abc01cba

2.2.2 基本运算

- **符号串方幂运算** 设x是字母表∑上的符号串，z是由n(≥0)个x自身连接得到的符号串，则称符号串z是由符号串x的n次方幂运算的结果，记为$z = x^n$。特别约定，$x^0 = \varepsilon$, $x^1 = x$。

讨论：$x \cdot x = x^2$，$x \cdot x \cdot x = x^3$，…，$\underbrace{x \cdot x \cdot \cdots x}_{n个x} = x^n$

- **符号串集连接运算** 设A，B是字母表∑上的符号串集，·是符号串集连接运算，则$C = A \cdot B = \{x \cdot y \mid x \in A，y \in B\}$。笛卡尔积

- **符号串集方幂运算** 设A是字母表∑上的符号串集，则C是由n(≥0)个A自身连接得到的符号串集，则称符号串集C是由符号串A的n次方幂运算的结果，记为$C = A^n$。特别约定，$A^0 = \{\varepsilon\}$,$A^1 = A$。

2.2.2 基本运算

- **符号串集正闭包运算** 设A是字母表Σ上的符号串集， A^+ 是A的正闭包，则： $A^+ = A^1 \cup A^2 \cup A^3 \cup \cdots \cup A^n \cdots$ 。
- **符号串集闭包运算** 设A是字母表Σ上的符号串集， A^* 是A的闭包，则 ： $A^* = A^0 \cup A^+$ ，

  即：$A^* = A^0 \cup A^1 \cup A^2 \cup A^3 \cup \cdots \cup A^n \cdots$ 。

语句是单词集合的连接

例：设字母表∑＝{a，b}，A＝{aa，bb}，B＝{ab，ba}，C＝{a}，则

$A \cdot B$ = {aaab，aaba，bbab，bbba}，

A^2 = {aaaa，aabb，bbaa，bbbb}，

C^+ = {a，aa，aaa，…} = $\{a^n \mid n \ge 1\}$，

C^* = {ε,a，aa，aaa，…} = $\{a^n \mid n \ge 0\}$ 。

源程序就是语句的闭包

2.3 文法和语言的形式定义

规则是字母表V上形如 $\alpha::=\beta$的式子，可以简写成$\alpha\rightarrow\beta$。其中，符号串$\alpha\in V^{+}$称为规则的左部，符号串$\beta\in V^{*}$称为规则的**右部**。**规则**也称为**重写规则**、**产生式**或**生成式**。

特别地，$\alpha::=\varepsilon$（ε空串）称为α的**空规则**。

对于相同左部的多个规则，可以使用符号 | 简写。如，规则$\alpha::=\beta$和$\alpha::=\delta$，简写成$\alpha::=\beta\mid\delta$。简写为$\alpha\rightarrow\beta\mid\delta$

定义2.1 文法

文法G定义为一个四元组（V_N，V_T，P，S），记为G＝（V_N，V_T，P，S）。其中，

① V_N是非空有穷集合，称为非终结符集，其元素称为非终结符；

② V_T是有穷集合，称为终结符集，其元素称为终结符；

③ P是非空有穷集合，称为规则集，其元素是字母表$V_N \cup V_T$上的规则，$V_N \cup V_T$称为文法的字母表V，且$V_N \cap V_T = \Phi$；

④ $S \in V_N$，称为开始符。

2.3 文法和语言的形式定义

例2.1 定义文法G1如下：

$G1=(V_N, V_T, P, S)$，

其中，$V_N = \{S\}$，

$V_T = \{a, b\}$，

$P = \{S \rightarrow aSb, S \rightarrow ab\}$

通常,文法还可以采用其它形式给出定义。如文法G1可以写成其它形式如下。

$G1=(\{S\}, \{a, b\}, P, S)$，

其中，$P= \{S \rightarrow aSb, S \rightarrow ab\}$

或者写成：

$G1=(\{S\}, \{a, b\}, \{S \rightarrow aSb, S \rightarrow ab\}, S)$

$G1[S]: S \rightarrow aSb, S \rightarrow ab$

定义2.2 直接推导、直接归约

设文法G＝（V_N，V_T，P，S），如果$\alpha \rightarrow \beta \in P$，则称$\gamma\alpha\delta$推导出$\gamma\beta\delta$，记为$\gamma\alpha\delta \Rightarrow \gamma\beta\delta$，其中，$\gamma$，$\delta \in V^*$。

$\gamma\alpha\delta \Rightarrow \gamma\beta\delta$也称为直接推导或一步推导。

如果$\gamma\alpha\delta \Rightarrow \gamma\beta\delta$，则也称为$\gamma\beta\delta$归约到$\gamma\alpha\delta$，也称为直接归约或一步归约。

例如，例3.1 定义的文法G1＝({S}，{a，b}，{S→aSb，S→ab}，S)，推导例子有：

（1）S⇒ aSb （α＝S，β＝aSb，γ＝ε，δ＝ε）

（2）aSb ⇒ aaSbb （α＝S，β＝aSb，γ＝a，δ＝b)

（3）aSb ⇒ aabb （α＝S，β＝ab，γ＝a，δ＝b)

（4）aSbSb ⇒ aaSbbSb （α＝S，β＝aSb，γ＝a，δ＝bSb ）

定义2.3 多步推导、多步归约

设文法G=（V_N，V_T，P，S），α，β∈（V_N∪V_T）*，　如果α，β之间存在推导序列：

$$\alpha = W_0 \Rightarrow W_1 \Rightarrow W_2 \cdot\cdot\cdot\cdot \Rightarrow W_n = \beta \ (n \geqslant 1),$$

则称α经过n步推导出β，记为$\alpha \overset{+}{\Rightarrow} \beta$。其中，$W_i$∈($V_N$∪$V_T$)*(1≤i≤n)。$\alpha \overset{+}{\Rightarrow} \beta$也称n步推导或多步推导。

如果$\alpha \overset{+}{\Rightarrow} \beta$，也称为β归约到α，也称为n步归约或多步归约。

例如，例3.1 定义的文法G1=({S}，{a，b}，{S→aSb，S→ab}，S)，多步推导($\overset{+}{\Rightarrow}$)例子有：

(1) $S \overset{+}{\Rightarrow} ab$ （∵$S \Rightarrow ab$)

(2) $S \overset{+}{\Rightarrow} aabb$ （∵ $S \Rightarrow aSb \Rightarrow aabb$)

(3) $S \overset{+}{\Rightarrow} aaaSbbb$ （∵ $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb$)

(4) $aSb \overset{+}{\Rightarrow} aaabbb$ （∵ $aSb \Rightarrow aaSbb \Rightarrow aaabbb$)

定义2.4　0步或0步以上推导与归约

设文法G＝（V_N，V_T，P，S），α，β ∈（$V_N \cup V_T$）*，如果有α → β 或 $\alpha \overset{+}{\Rightarrow} \beta$，则称α经过0步或0步以上推导出β，记为 $\alpha \overset{*}{\Rightarrow} \beta$。亦称β经过0步或0步以上归约到α。

例如，例3.1 定义的文法G1＝({S}，{a，b}，{S→aSb，S→ab}，S)，0步或0步以上推导($\overset{*}{\Rightarrow}$)例子有：

（1）$S \overset{*}{\Rightarrow}$ ab，因为有$S \overset{+}{\Rightarrow}$ ab

（2）$S \overset{*}{\Rightarrow}$ aabb，　因为有$S \overset{+}{\Rightarrow}$aabb

（3）$S \overset{*}{\Rightarrow}$ aaabbb，因为有$S \overset{+}{\Rightarrow}$aaabbb

（4）aSb $\overset{*}{\Rightarrow}$ aaabbb，因为有aSb$\overset{+}{\Rightarrow}$ aaabbb

（5）aSbSb $\overset{*}{\Rightarrow}$aSbSb，因为有aSbSb $\overset{+}{\Rightarrow}$ aSbSb

定义2.5 句型、句子

设文法G＝（V_N，V_T，P，S），如果有$S\overset{*}{\Rightarrow}\beta$，则称β是文法**G的句型**。如果有$S\overset{*}{\Rightarrow}\beta$，且$\beta\in V_T^*$，则称β是文法**G的句子**。

例如，例3.1 定义的文法G1＝({S}，{a，b}，{S→aSb，S→ab}，S)，句型和句子例子有：

（1）ab是G的句子，因为有$S\overset{*}{\Rightarrow}ab$，$ab\in V_T^*$

（2）aabb是G的句子，因为有$S\overset{*}{\Rightarrow}aabb$，$aabb\in V_T^*$

（3）aaaSbbb是G的句型，因为有$S\overset{*}{\Rightarrow}aaaSbbb$($aaaSbbb\notin V_T^*$)

定义2.6 语 言

文法$G=(V_N, V_T, P, S)$的产生语言定义为文法G的句子集合，记为L(G)。即：

$$L(G)=\{\beta \mid S \overset{*}{\Rightarrow} \beta,\ \beta \in V_T^*\}。$$

例如，例3.1 定义的文法G1，其语言L(G)是由相同个数（≥1）的a和b符号、且b符号均在a符号之后出现的符号串构成的集合。即：

$$L(G)=\{a^nb^n \mid n\geq 1\}。$$

a={
b=}

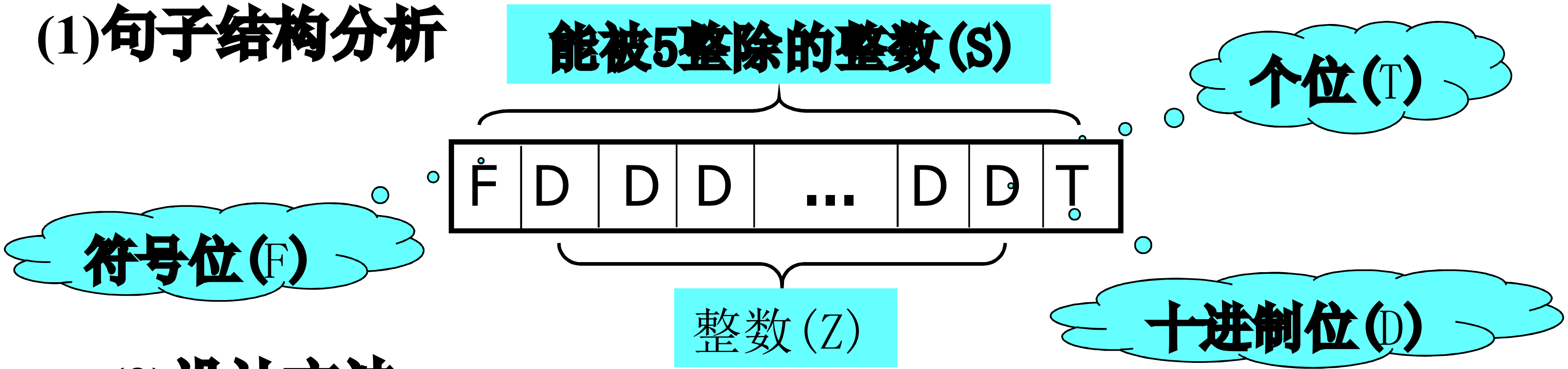
例3.1 $G1=(\{S\},\ \{a, b\},\ P,\ S)$，
其中，$P=\{S\rightarrow aSb,\ S\rightarrow ab\}$

2.3　文法设计举例

试设计一文法G，使得L(G)为能被5整除的整数集。

解法Ⅰ：

(1)句子结构分析



(2)设计文法

G[S]：$S \rightarrow F\,Z\,T$

$F \rightarrow + \mid - \mid \varepsilon$

$Z \rightarrow ZD \mid \varepsilon$

$D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

$T \rightarrow 0 \mid 5$

2.3 文法设计举例

解法Ⅱ：

(1)句子结构分析

句型1 | F | T |

句型2 | T |

句型3 | F | D | D | D | … | D | D | T |

句型4 | D | D | D | … | D | D | T |

(2)设计文法

$$
\begin{aligned}
G[S]:\ & S \rightarrow FT \mid T \mid FZT \mid ZT \\
& F \rightarrow + \mid - \\
& Z \rightarrow ZD \mid D \\
& D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
& T \rightarrow 0 \mid 5
\end{aligned}
$$

思考题：试设计一文法G，使得L(G)为能被3整除的整数集。

定义2.7 文法等价

设G1 和G2是两个文法，如果L(G1)＝L(G2)，则称文法G1和G2是等价的。

例如，下列文法G2和G3是等价的。因为它们产生的语言都是以字母a开头、字母a和b构成的符号串的集合。即L(G2)＝L(G2）＝{a}{a，b}*。

G2＝({S，C}，{a，b}，P，S），
其中，P＝{S→aC，C→aC ，C→bC, C→ε}。

G3＝({S}，{a，b}，P，S），
其中，P＝{S→Sa，S→Sb ，S→a}。

2.4 文法类型

对规则构成加以限制，可以将文法的分类为四种类型：0型文法、1型文法、2型文法和3型文法。

(1) 0型文法　　设文法G＝（V_N，V_T，P，S），如果任意 $\alpha \to \beta \in P$，α 中至少含有一个非终结符，则称文法G属于0型文法。0型文法，也称为短语文法。

例3.4　文法G4定义如下。显然G4是0型文法。L(G4)＝{}。

G4 ＝（V_N，V_T，P，S），

其中，V_N＝{A，B，S}，

V_T＝{0，1}，

P ＝{S→0AB，1B→0，B→SA | 01，

A1→SB1，A0→S0B}

2.4 文法类型

(2) 1型文法　　设文法G＝（V_N，V_T，P，S），如果任意 $\alpha \rightarrow \beta \in P$，$\alpha$ 中至少含有一个非终结符，且除空规则之外，α 的长度不大于 β 的长度，即 $|\alpha| \leqslant |\beta|$，则称文法G属于1型文法。　1型文法，也称为上下文有关文法。

例3.5 文法G5定义如下，显然G5是1型文法。

$L(G5) = \{a^n b^n c^n \mid n \geq 1\}$。

$G5 = (V_N, V_T, P, S)$，

其中，$V_N = \{S, B, C\}$，

$V_T = \{a, b, c\}$，

$P = \{S \rightarrow aSBC \mid aBC, CB \rightarrow BC,$

$aB \rightarrow ab, bB \rightarrow bb,$

$bC \rightarrow bc, cC \rightarrow cc\}$

2.4 文法类型

（3）2型文法　　设文法G＝（V_N，V_T，P，S），如果任意 $\alpha \rightarrow \beta \in P$，$\alpha \in V_N$，则称文法G属于2型文法。2型文法，也称为上下文无关文法。

例3.6　文法G6定义如下，显然G6是2型文法。

$L(G6) = \{w\$w^R \mid n \geq 0,\ w^R$ 为w之逆，$w \in \{0, 1\}^*\}$。

$G6 = (V_N, V_T, P, S)$，

其中，$V_N = \{S\}$，

$V_T = \{\$, 0, 1\}$，

$P = \{S \rightarrow 0S0 \mid 1S1 \mid \$\}$

2.4 文法类型

(4) 3型文法　　设文法G＝（V_N，V_T，P，S），如果任意 $\alpha \rightarrow \beta \in P$，$\alpha \in V_N$，且 β 只能是aB或a（除空规则之外），则称文法G属于右线性3型文法。

设文法G＝（V_N，V_T，P，S），如果任意 $\alpha \rightarrow \beta \in P$，$\alpha \in V_N$，且 β 只能是Ba或a（除空规则之外），则称文法G属于左线性3型文法。

左线性3型文法和右线性3型文法统称3型文法，也称为正规文法。

文法G7定义如下。显然G7是3型文法。

G7 ＝（V_N，V_T，P，S），

其中，V_N＝{S，A，B}，

V_T＝{0，1}，

$P = \{S \rightarrow A0 \mid B1,\ A \rightarrow 0 \mid 1,\ B \rightarrow 0 \mid 1\}$

2.4 文法类型

文法分类是对规则形式逐步加以限制而得。换言之，从0型文法到1型文法、2型文法和3型文法，其规则形式逐步简单。自然，其表达力也随之逐步减弱。

如果L0、L1、L2和L3分别是0型文法、1型文法、2型文法和3型文法能产生的语言之集，则有如下关系：

$$L0 \supsetneq L1 \supsetneq L2 \supsetneq L3。$$

在《编译原理》里，仅仅涉及到2型文法和3型文法的使用。后面，也仅仅继续讨论2型文法（或3型文法）的有关内容。

2.5　上下无关文法及其语法树

上下无关文法一个显著特征是规则左部一定有且仅有一个非终结符。利用这个特征，可以不列出V_N和V_T，给出一个上下无关文法的简洁描述方法：①文法名G改写成G[S]，其中，S表示开始符；②规则集P，仅书写其具体规则。

如，例2.7定义的文法G6，简化描述如下。

G6[S]：

$S \rightarrow A0 \mid BS0$

$A \rightarrow 0 \mid 1$

$B \rightarrow 0 \mid 1$

⇒

$V_N = \{S, A, B\}$

$V_T = \{0, 1\}$

定义2.8 最左推导、最右推导

如果在推导的每一步总是选择当前句型的最左（最右）边非终结符进行推导，则称这种推导过程为**最左（最右）推导**。最右推导，也叫**规范推导**。由规范推导所得的句型，叫做**规范句型**。规范推导的逆过程，叫做**规范归约**。

例2.8　已知文法如下，试给出句子aabbaa的推导过程。

G[S]：S→aAS | a
　　　A→SbA | SS | ba

最左推导：S ⇒ aAS ⇒ aSbAS ⇒ aabAS ⇒ aabbaS ⇒ aabbaa

最右推导：S ⇒ aAS ⇒ aAa ⇒ aSbAa ⇒ aSbbaa ⇒ aabbaa

一般推导：S ⇒ aAS ⇒ aSbAS ⇒ aSbAa ⇒ aabAa ⇒ aabbaa

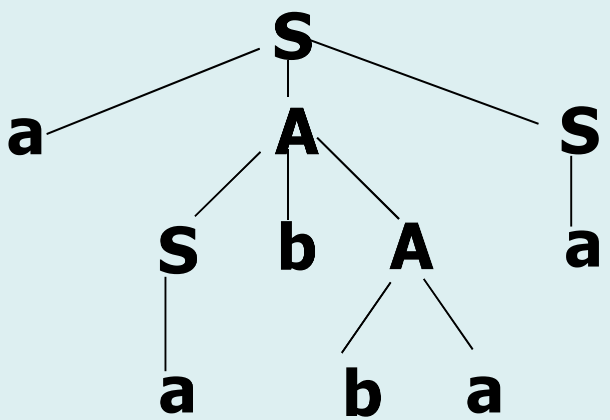
定义2.9 语法树

假设文法G=（V_N，V_T，P，S），则文法G的语法树是一个满足下列条件的多叉树：

（1）以文法开始符S做为树根；

（2）以终结符号或非终结符号做为树的其他结点，且子树根和其孩子结点分别是某规则的左部和右部。

例如，例3.8定义的文法G[S]，句子aabbaa对应的语法树如下。



例3.8 定义文法G[S]:

G[S]：S→aAS | a

A→SbA | SS | ba

推论： ①非叶子结点一定是非终结符

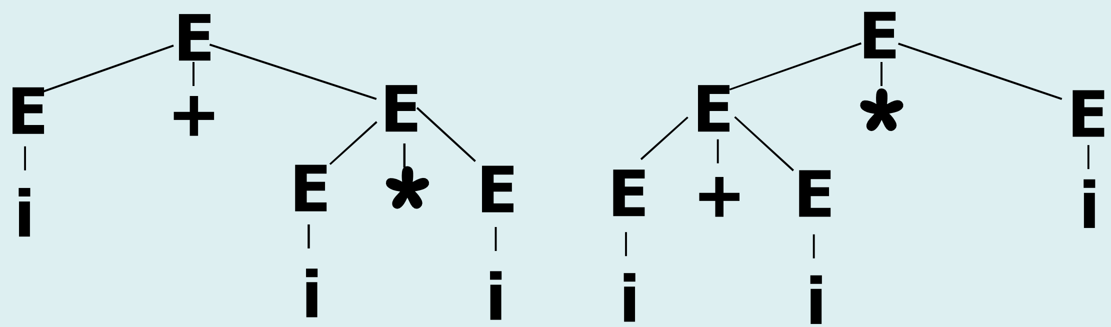
②全部叶子结点组成的符号串是文法的句子

定义2.10 语法二义性

如果一个文法G，某个句子存在对应的至少两棵不同的语法树，则称文法G是二义性的。

例3.9 已知文法G[E]：E→E+E | E*E | i ，证明G是二义性的。

证明： ∵句子i+i*i存在下列两棵不同的语法树



∴ 文法G[E]是二义性的文法

推论 ① 如果文法是无二义性的，一个句子的语法树反映了该句子的全部推导过程；②如果文法是无二义性的，一个句子的最左（最右）推导是唯一的。

语言的先天二义性

文法的二义性，并不等同于语言的二义性，尽管两者之间可能存在非必然的联系。

因为二义性文法G，可能存在与之等价的无二义性的文法G′，即L(G)＝L(G′）。

如果一个语言不存在无二义性的文法，则称该语言是先天二义性的。

例如，语言$L=\{a^i b^j c^k \mid (i=j$ 或 $i=k)，(i，j，k \geq 1)\}$不存在无二义性的文法，是先天二义性的语言。

已经证明：文法的二义性判定问题是递归不可解的。即不存在这个判定问题的算法。

2.6　句型分析

假设文法G[S]是语言L之文法，即L(G)＝L，则“符号串α是否符合语言L的语法问题”被等价地转化成“推导或归约问题”，即：

$$S \overset{*}{\Rightarrow} \alpha \wedge \alpha \in V_T^{\ *}$$

这样，自然地形成了推导法和归约法两大类分析方法。推导法和归约法，也分别称为自上而下的分析方法和自下而上的分析方法。

2.6.1 自上而下的分析方法

自上而下分析法：从文法开始符号出发，反复使用规则，寻找匹配符号串（推导）的句型，直到推导出句子或规则用遍。**进行每步推导时，存在两个选择问题：**

(1) 选择句型中哪一个非终结符进行推导

(2) 选择非终结符的哪一个规则进行推导

问题(1)可以采用最左推导解决。问题(2)通常需要穷举每一个规则的可能推导。

成功：在推到过程中一旦出现个符号串α，便结束穷举过程，断定符号串α是句子。

失败：当穷举全部可能的推导，而不存在一个符号串α之推导过程的时候，才可以断定符号串α不是句子。

2.6.1 自上而下的分析方法

（1）输入串cabd的推导过程

（2）输入串cabc的推导过程

G[S]：

1. S→cAd
2. A→a
3. A→ab

序号	推导过程	输入串	说明
1	S	cabd	从S开始推导
2	S=>cAd	cabd	选1号规则，c匹配成功
3	S=>cad	cabd	选2号规则，a匹配成功d不成功，回溯
4	S=>cAd	cabd	选3号规则推导
5	S=>cabd	cabd	abd匹配成功，所以cabd是一个句子

这样带回溯的推导法效率很低。后面重点研究避免回溯的推导法！

2.6.2 自下而上的分析方法

自下而上分析法：从输入符号串α开始，逐步进行"归约"，直至归约出文法的开始符号 S，则输入串α是文法G定义的语言的句子。否则不是。

这种分析方法在进行每步归约时，存在两个如何选择句型α的子串β进行归约的问题（α =β δ）。

如果文法规则没有相同的右部，则在语法分析的过程中，一旦出现子串β与某条规则的右部相同，就可以使用这条规则进行归约,简单优先分析法就是采用此方法进行归约。

但这种限制，实际上也限制了文法的表达能力，所以通常是通过在句型中寻找所谓的"**句柄**"的途径解决的。

定义 2.11 短语、直接短语、句柄

设G[S]是一文法，αβδ是文法G的句型，如果有$S \overset{*}{\Rightarrow} \alpha A \delta$且$A \overset{+}{\Rightarrow} \beta$，则称β是句型αβδ的、相对于非终结符A的短语。

特别地，当$A \overset{+}{\Rightarrow} \beta$实际是$A \Rightarrow \beta$即一步推导时，则又称β是句型αβδ的、相对于非终结符A的直接短语（或简单短语）。

句型的最左直接短语，称为该句型的句柄。

短语的理解：

"αβδ是文法G的句型"，即$S \overset{*}{\Rightarrow} \alpha\beta\delta$

"$S \overset{*}{\Rightarrow} \alpha A \delta$且$A \overset{+}{\Rightarrow} \beta$"，即$\underbrace{S \Rightarrow \cdots \Rightarrow}_{\geqslant 0} \alpha A \delta \underbrace{\Rightarrow \cdots \Rightarrow}_{\geqslant 1} \alpha\beta\delta$

这表明，如果β是句型αβδ的、相对于A的短语，则至少存在一个推导，使得$\alpha A \delta \overset{+}{\Rightarrow} \alpha\beta\delta$，或者$\alpha\beta\delta \overset{+}{\Leftarrow} \alpha A \delta$。

特别地，如果β是直接短语，则$\alpha A \delta \Rightarrow \alpha\beta\delta$，或者$\alpha\beta\delta \Leftarrow \alpha A \delta$。

短语、直接短语、句柄举例

例2.12　设文法G[E]：E→E+T | T，T→T*F | F，F→(E) | i，分析句型i+i*i的短语、直接短语和句柄。

从句型i+i*i的如下的推导过程之一如下：

$$E \Rightarrow E+T \Rightarrow E+T*F \Rightarrow E+T*i_3 \Rightarrow E+F*i_3 \Rightarrow E+i_2*i_3 \Rightarrow T+i_2*i_3 \Rightarrow F+i_2*i_3 \Rightarrow i_1+i_2*i_3$$

可以看出部分短语如下：

(1) i_3是句型$E+T*i_3$的、相对于非终结符F的直接短语；

(2) i_2是句型$E+i_2*i_3$的、相对于非终结符T的短语，F的直接短语；

(3) i_1 是句型i+i*i的、相对于非终结符F的短语、直接短语和句柄。

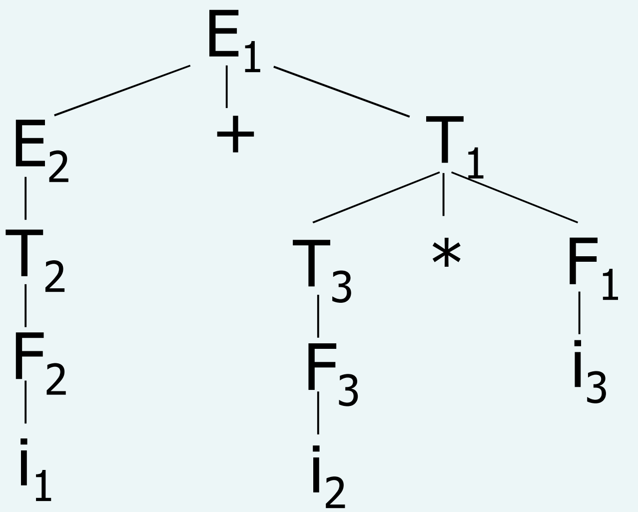
为了讨论方便起见，句型i+i*i改写成$i_1+i_2*i_3$。其中，下标1、2、3仅仅是用来标记不同i出现在句型中的位置。

语法树与短语、直接短语、句柄举例

从句型的语法树上，可以直观地找出句型的短语。在语法树中相对于每个非空子树的根是一个非终结符，其非空子树叶子结点组成的符号串即为短语。如果子树根与子树叶子结点之间均为父子关系，则该短语还是直接短语。

G[E]：E→E+T | T
T→T*F | F
F→(E) | i

例2.12 设文法G[E]定义如右，分析句型i+i*i的短语、直接短语和句柄。



句型i+i*i语法树

短语： $i_1+i_2*i_3$（相对于E_1）
i_1（相对于E_2）
i_2*i_3（相对于T_1）
i_1（相对于T_2）
i_2（相对于T_3）

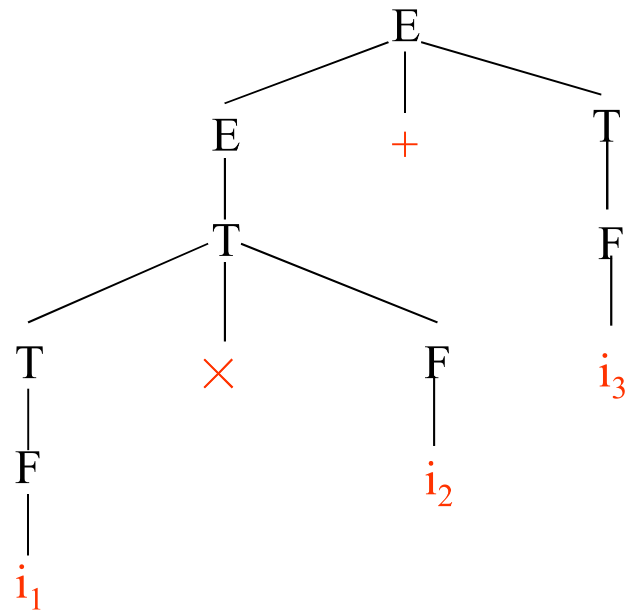
直接短语： i_3（相对于F_1）
i_2（相对于F_3）

句柄： i_1（相对于F_2）

根据句柄进行归约举例

仍然以例2.12定义的文法G[E]为例，给出通过寻找句柄，对符号串 i+i*i进行归约法分析过程，说明归约法的基本思想。

G[E]：E→E+T | T, T→T*F | F, F→(E) | i



分析过程借用了语法树确定句型的句柄，显然是不符合逻辑的。因此，如何依据文法寻找句柄是归约法的关键问题。

2.7 文法在实用中的一些说明

在实际应用中，对于文法规则提出了一些限制条件，但这些并没有限制文法的语言描述能力。限制下列 3种规则的使用：

（1）有害规则　形如U→U的规则，称为有害规则。

（2）不可达规则　不在任何规则右部出现的非终结符对应的规则，称为不可达规则。

（3）不可终止规则　如果从某非终结符开始，不可能推导出任意终结串来，则该非终结符对应的规则称为不可终止规则。

不含有多余规则的文法，称为压缩过的文法。在后面讨论的文法时，都假设是压缩过的的文法。

例2.13　文法G定义如下，

G[S]：　S→Be

B→Ce | Af，

A→Ae | e

C→Cf

D→f

显然，C→Cf是不可终止规则，D→f是不可达规则。除去这些规则及其相关规则之后，得到的等价文法G′如下。

G′[S]：　S→Be

B→Af

A→Ae | e

ε规则问题

在文法设计中，使用ε规则有时会带来方便，但会导致文法讨论和证明的复杂。

一个上下文无关文法G是否必须使用ε规则，完全取决于文法G产生的语言L(G[S])中是否含有ε语句。

可以证明，如果ε∉L(G[S]),则存在一个等价的文法G’[S’]，且G’ 不含ε规则。

如果ε∈ L(G[S]),则存在一个等价的文法G’[S’]，且G’ 仅含S’ →ε的一个空规则。

提示：使用“代入法”，即可得到等价的文法G’(S’)

“代入法”举例

已知文法G[S]如下。容易看出L(G[S])={0,1,+0,+1,−0,−1}，且ε∉L(G[S])。

G[S]:

S→FD

F→+ | − | ε

D→0 | 1

对于右部出现F的规则，使用F规则的右部替代其右部出现的F，之后删除F的规则，得到等价于G[S]的文法G’[S]如下。

G[S]:

S→ +D | −D | D

D→0 | 1

本章小结

本章重点介绍了形式化语言的基本理论、语言形式化描述方法、语言的文法分类与特性、文法与句型分析。

提出的基本概念有语言、文法、规则、推导、归约、句型、短语、句柄、语法树和语法二义性等。其中，推导是最核心的、关键的概念。重点掌握的内容是

① 设计一个已知语言的文法；

② 确定已知文法定义的语言；

③ 求句型的短语、直接短语和句柄。

④ 文法二义性判定。