第六章

上下文无关语言

形式语言与自动机

计算模型/自动机：

FA	PDA	LBA	TM
Finite Automata	**Push Down Automata**	**Linear Bounded Automata**	**Turing Machine**
有穷状态自动机	下推自动机	线性有界自动机	图灵机

文法：

• 正则文法	• 上下文无关文法	• 上下文有关文法	• 短语结构文法
• 3型文法	• 2型文法	• 1型文法	• 0型文法

- 正则文法所具有的描述能力是有限的，例如，在计算机高级程序设计语言的翻译中，正则文法只能对语言中组成单词的规则进行描述，通常称其为“词法”。也就是说，正则文法解决的是像标识符如何组成、整数如何组成、实数如何组成等问题。

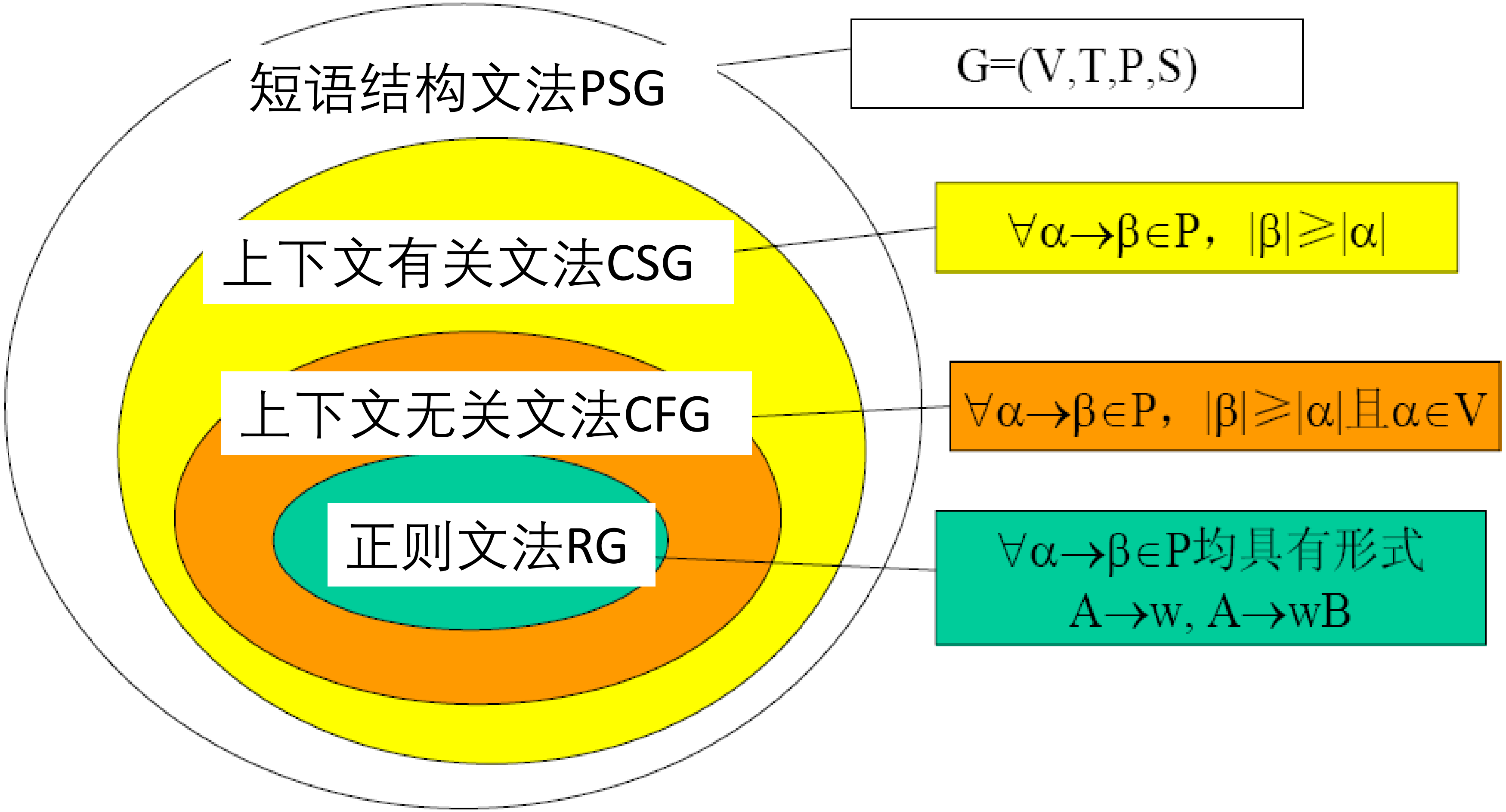
- 对于表达式、语句等更复杂的结构来说，正则文法就失去了描述能力，如含有成对的括号的语言，或$L = \{a^n b^n \mid n \geq 1\}$等语言。

- 经验表明：高级程序设计语言的绝大多数语法结构都可以用上下文无关文法(**CFG**)描述，因此，高级程序设计语言的规范说明及其编译是**CFG**的一个重要应用领域。

- 用来描述高级程序设计语言的**BNF**（巴科斯范式）就是**CFG**的一种特殊形式。**CFG**的这种表达能力，以及计算机系统对于处理**CFG**的适应性，使得**CFG**和相应的上下文无关语言在计算机的各种相关语言的处理、相关理论的研究中占有非常重要的地位。

章节目录

Callback:文法的乔姆斯基体系



四类文法的区别在于：产生式的形式

Callback:上下文无关文法

定义2-5 设文法G=(V, T, P, S)，则

- 如果对于$\forall \alpha \to \beta \in P$，均有$|\beta| \geq |\alpha|$，并且$\alpha \in V$成立，则称G为2型文法(type 2 grammar)，或上下文无关文法(content free grammar, CFG)。

  L(G)叫做2型语言，或上下文无关语言(context free language, CFL)。

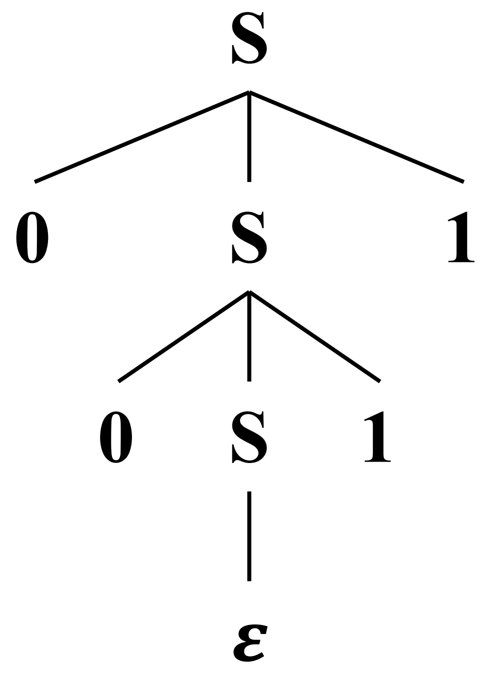
如果$\alpha \to \beta \in P$，则无论α出现在句型的任何位置，我们都可以将α替换成β，而不考虑α的上下文。

6.1 上下文无关文法

6.1.1 上下文无关文法的派生树

例6-1 在文法 $S \rightarrow 0S1 \mid 1S0 \mid SS \mid \varepsilon$ 中推导0011。

$$
\begin{aligned}
S &\Rightarrow 0S1 \\
&\Rightarrow 00S11 \\
&\Rightarrow 0011
\end{aligned}
$$



派生树

<u>**定义6-1**</u> 设有CFG G = (V, T, P, S)，G的派生树是满足如下条件的有序树：

① 树的每个顶点有一个标记X，且$X \in V \cup T \cup \{\varepsilon\}$；

② 树根的标记为S；

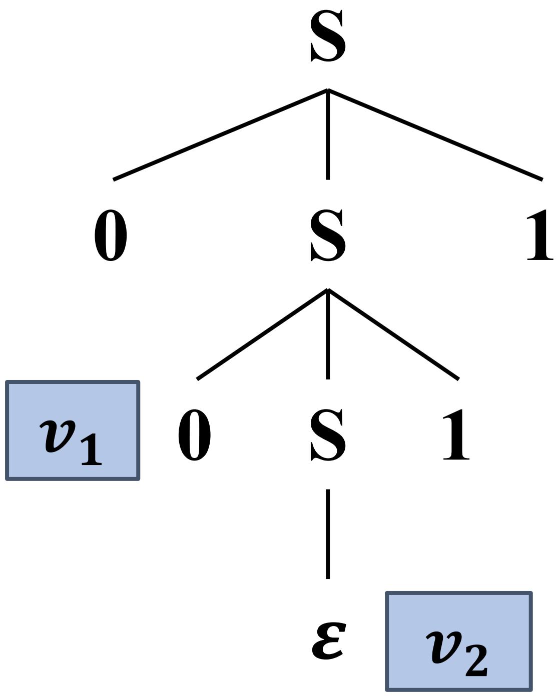
③ 如果一个非叶子节点v标记为A，其子节点从左到右依次为$v_1, v_2, \cdots, v_n$，并且它们分别标记为$X_1, X_2, \cdots, X_n$，则$A \rightarrow X_1X_2\cdots X_n \in P$。

④ 如果X是一个非叶子顶点的标记，则$X \in V$；

⑤ 如果一个顶点v标记为ε，则v是该树的叶子，并且v是其父顶点的唯一儿子。

派生树

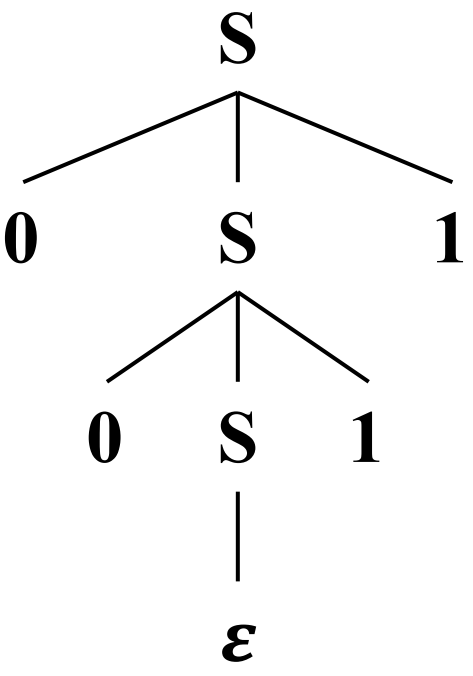
<u>**定义6-2**</u> 设有文法G的一颗派生树T， v_1, v_2是T的两个不同顶点，如果存在顶点v，v至少有两个儿子，使得v_1是v的较左儿子的后代，v_2是v的较右儿子的后代，则称顶点v_1在顶点v_2的左边，顶点v_2在顶点v_1的右边。



6.1.1 上下文无关文法的派生树

派生树

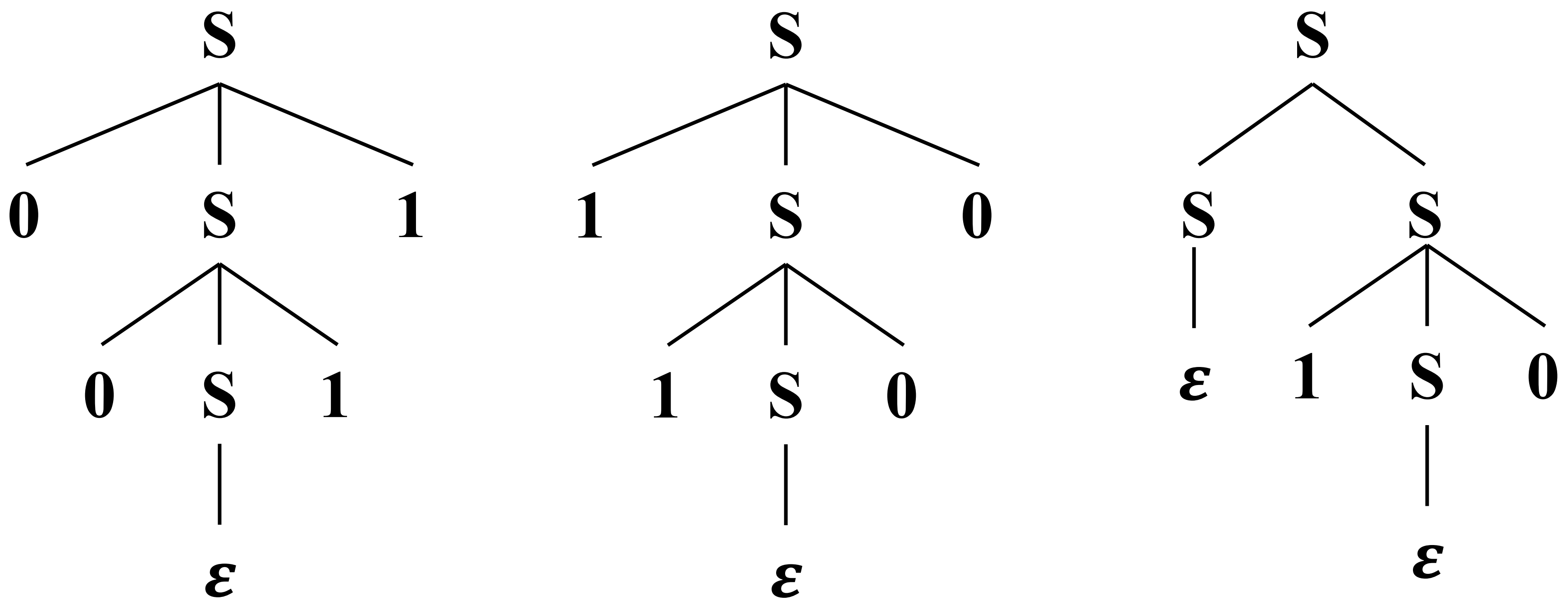
<u>定义6-3</u> 设有文法G的一颗派生树T， T的所有叶子顶点从左到右依次标记为$X_1, X_2, \cdots, X_n$，则称符号串$X_1X_2\cdots X_n$是T的结果。



- 00ε11，即0011是右侧派生树的结果。
- 一个文法可以有多棵派生树T，他们可以有不同的结果。
- 为了明确起见，对于任意一个CFG G，可以称“G的结果为α的派生树”为G的对应于句型α的派生树，简称句型α的派生树。

6.1.1 上下文无关文法的派生树

续例6-1 文法 $S \rightarrow 0S1 \mid 1S0 \mid SS \mid \varepsilon$ 的派生树。



$S \Rightarrow 0S1$
$\Rightarrow 00S11$
$\Rightarrow 0011$

$S \Rightarrow 1S0$
$\Rightarrow 11S00$
$\Rightarrow 1100$

$S \Rightarrow SS$
$\Rightarrow 1S0$
$\Rightarrow 10$

派生子树

定义6-4 满足派生树定义中除了第（2）条以外各条的树称为派生子树(subtree)。如果这个子树的根标记为A，则称之为A子树。

Callback:

唯一差别是派生子树的根结点可以是非开始符号。

定义6-1 设有CFG G = (V, T, P, S)，G的派生树是满足如下条件的有序树：

① 树的每个顶点有一个标记X，且$X \in V \cup T \cup \{\varepsilon\}$；

② 树根的标记为S；

③ 如果一个非叶子节点v标记为A，其子节点从左到右依次为$v_1, v_2, \cdots, v_n$，并且它们分别标记为$X_1, X_2, \cdots, X_n$，则$A \rightarrow X_1X_2 \cdots X_n \in P$。

④ 如果X是一个非叶子顶点的标记，则$X \in V$；

⑤ 如果一个顶点v标记为ε，则v是该树的叶子，并且v是其父顶点的唯一儿子。

定理6-1 设CFG $G = (V, T, P, S)$，$S \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的派生树。

证明：证一个更为一般的结论：对于任意$A \in V$，$A \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的A子树。

① 充分性：设G有一棵结果为α的A子树，非叶子顶点的个数n施归纳，证明$A \Rightarrow^* \alpha$成立。

- 当n=1时，该子树是一个二级子树。假设此树的叶子顶点的标记从左到右依次为$X_1, X_2, \ldots, X_n$，由定义知，必有$A \rightarrow X_1X_2\ldots X_n \in P$。该子树的结果α，则$X_1X_2\ldots X_n = \alpha$，故$A \Rightarrow^* \alpha$成立。

6.1.1 上下文无关文法的派生树

定理6-1 设CFG G = (V, T, P, S)，$S \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的派生树。

证明：证一个更为一般的结论：对于任意A∈V，$A \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的A子树。

- 设当$n \le k$ $(k \ge 1)$时结论成立，往证当$n = k + 1$时也成立。
  - 设A子树有k+1个非叶子顶点，根顶点A的儿子从左到右依次为$v_1, v_2, \cdots, v_m$，并且它们分别标记为X_1，$X_1, X_2, \cdots, X_m$，由定义知，必有$A \to X_1 X_2 \cdots X_m \in P$ 。
  - 设分别以$X_1, X_2, \cdots, X_m$为根的子树的结果依次为$\alpha_1, \alpha_2, \cdots, \alpha_m$。显然，分别以$X_1, X_2, \cdots, X_m$为根的子树的非叶子顶点的个数均不大于k。

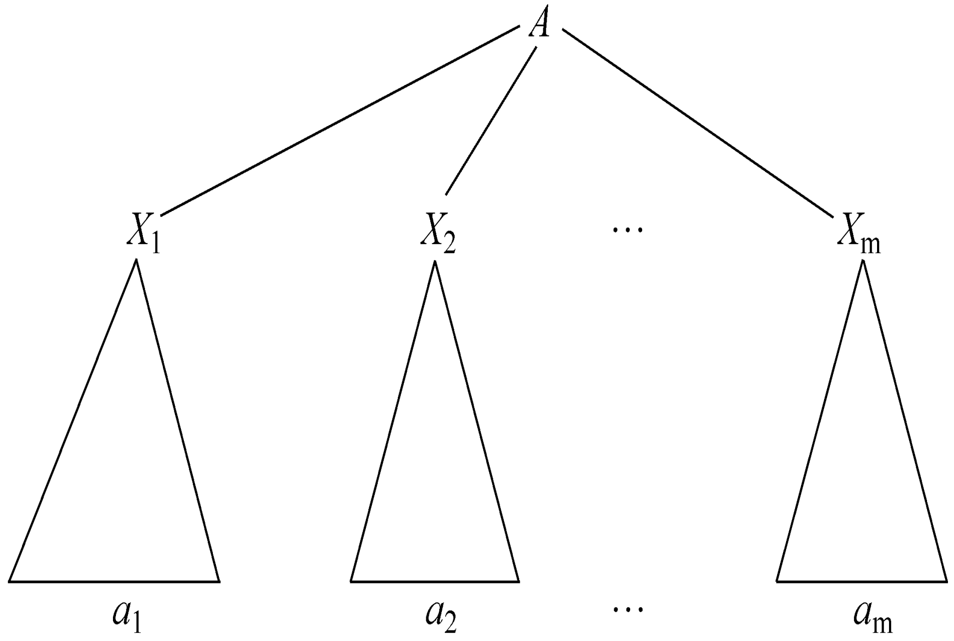
定理6-1 设CFG G = (V, T, P, S)，$S \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的派生树。

证明：证一个更为一般的结论：对于任意A∈V，$A \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的A子树。

$X_1 \Rightarrow^* \alpha_1$,
$X_2 \Rightarrow^* \alpha_2$,
…,
$X_m \Rightarrow^* \alpha_m$

$\alpha = \alpha_1\alpha_2\ldots\alpha_m$

$$
\begin{aligned}
A &\Rightarrow X_1X_2\ldots X_m \\
&\Rightarrow^* \alpha_1X_2\ldots X_m \\
&\Rightarrow^* \alpha_1\alpha_2\ldots X_m \\
&\ldots \\
&\Rightarrow^* \alpha_1\alpha_2\ldots\alpha_m \\
&\Rightarrow^* \alpha
\end{aligned}
$$



即结论当$n = k + 1$时成立，由归纳法原理，结论对任意的n成立。

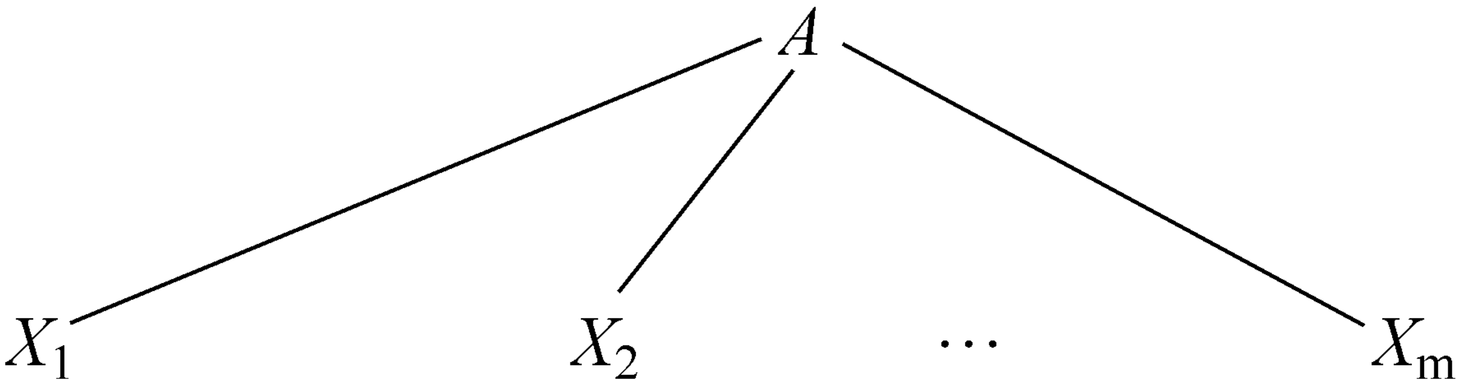
6.1.1 上下文无关文法的派生树

定理6-1 设CFG G = (V, T, P, S)，$S \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的派生树。

证明：证一个更为一般的结论：对于任意A∈V，$A\Rightarrow^*\alpha$的充分必要条件为G有一棵结果为α的A子树。

② 必要性：设$A \Rightarrow^n \alpha$，现施归纳于派生步数n，证明存在结果为α的派生子树。

- 当n=1时，由$A\Rightarrow\alpha$知$A\rightarrow \alpha \in P$。令$\alpha=X_1X_2\ldots X_n$，则有如下所示的A子树，故结论当n=1时成立。



6.1.1 上下文无关文法的派生树

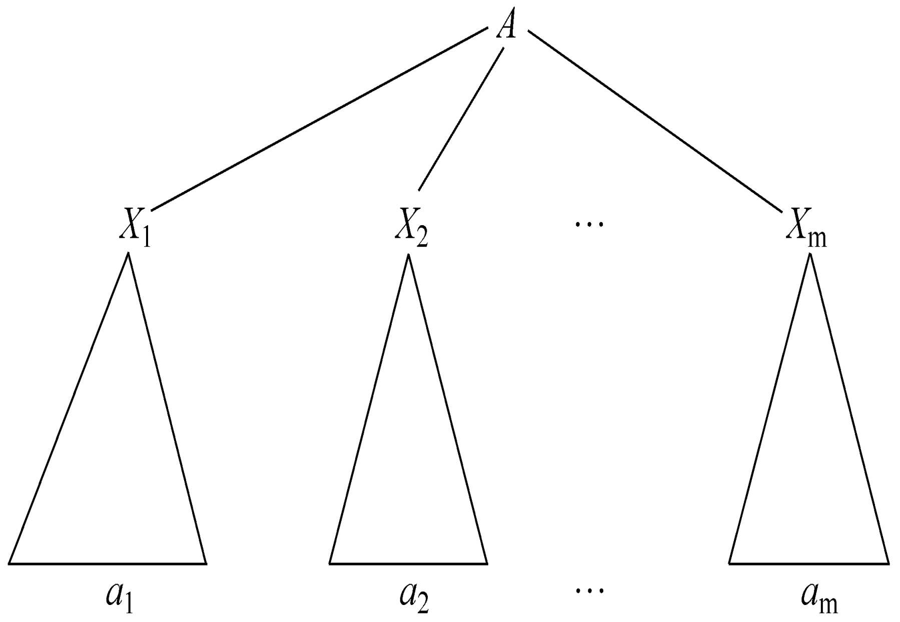
定理6-1 设CFG G = (V, T, P, S)，$S \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的派生树。

证明：证一个更为一般的结论：对于任意A∈V，$A \Rightarrow^* \alpha$的充分必要条件为G有一棵结果为α的A子树。

- 设当 $n \le k$ ($k \ge 1$) 时结论成立，往证当$n = k + 1$时结论也成立。
  令$A \Rightarrow^{k+1} \alpha$，则有

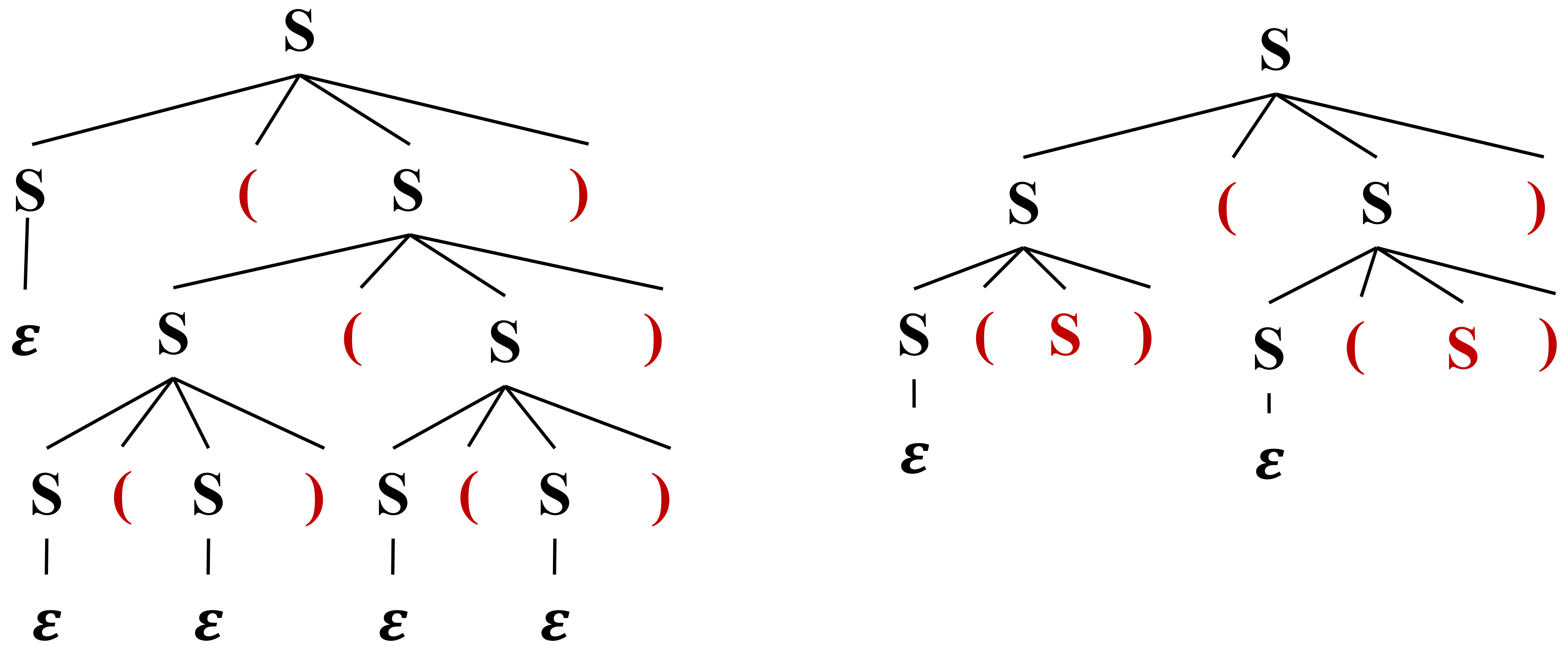
$$
\begin{aligned}
A &\Rightarrow X_1X_2\ldots X_m \\
&\Rightarrow^* \alpha_1X_2\ldots X_m \\
&\Rightarrow^* \alpha_1\alpha_2\ldots X_m \\
&\ldots \\
&\Rightarrow^* \alpha_1\alpha_2\ldots\alpha_m
\end{aligned}
$$

结果为α的A子树如右图，证毕。



6.1.1 上下文无关文法的派生树

例6-2 设G: S → S(S) | ε，求(()(()))和(S)((S))的派生树。



派生树的结果可以是句子，也可以是句型。实际上定义6-1中并没有要求派生树的叶子顶点的标记为文法的终结符号。

最左派生和最右派生

<u>**定义6-5**</u> 设有CFG G = (V, T, P, S)，α是G的一个句型。

- 如果α的派生过程中，每一步都是对当前句型的最左变量进行替换，则该派生称为最左派生；每一步所得到的句型也可叫作左句型；相应的归约叫作最右归约。
- 如果α的派生过程中，每一步都是对当前句型的最右变量进行替换，则该派生称为最右派生；每一步所得到的句型也可叫作右句型；相应的归约叫作最左归约。

6.1.1 上下文无关文法的派生树

■一般地，由于计算机系统处理一个输入串时，通常都是从左至右进行的，这使得对句型的分析按照从左到右的顺序进行是比较自然的。所以，最右派生还称为规范派生，规范派生产生的句型称为规范句型。

定理6-2 如果α是CFG G 的一个句型，则G中存在α的最左派生和最右派生。

定理6-3 如果α是CFG G 的一个句型，α的派生树与最左派生和最右派生是一一对应的，但是，这颗派生树可以对应多个不同的派生。

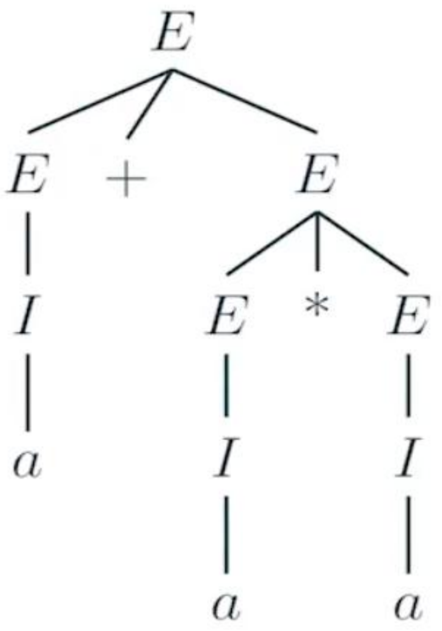
6.1 上下文无关文法

6.1.2 二义性

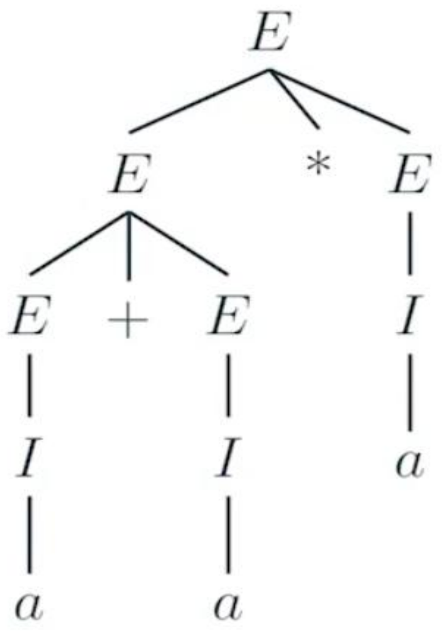
定理6-3指出了派生树与最左派生和最右派生的一一对应关系，

那么句型和派生树又有什么样的关系呢？

例6-3 考虑算数表达式的文法$\mathbf{G_{12}}$，句型$\boldsymbol{a+a*a}$有下面两颗语法派生树。



$$
\begin{aligned}
E &\Rightarrow E + E \\
&\Rightarrow E + E * E \\
&\Rightarrow^{*} a + a * a
\end{aligned}
$$

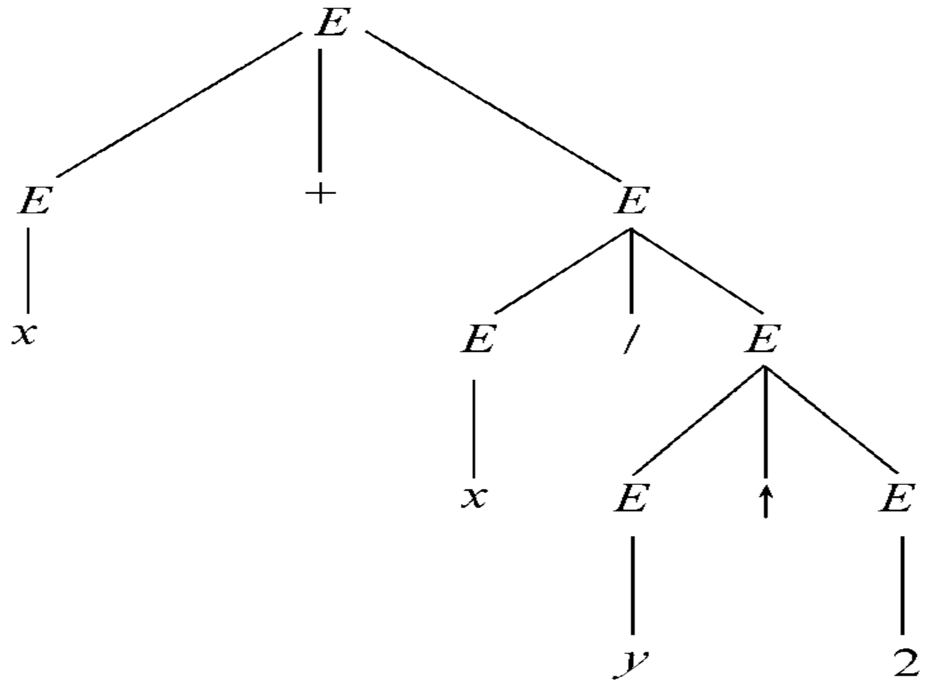


$$
\begin{aligned}
E &\Rightarrow E * E \\
&\Rightarrow E + E * E \\
&\Rightarrow^{*} a + a * a
\end{aligned}
$$

6.1.2 二义性

续例6-3 考虑算数表达式的文法G_{12}，句型$\boldsymbol{x+x/y**2}$在文法G_{12}中有三个不同的最左派生。

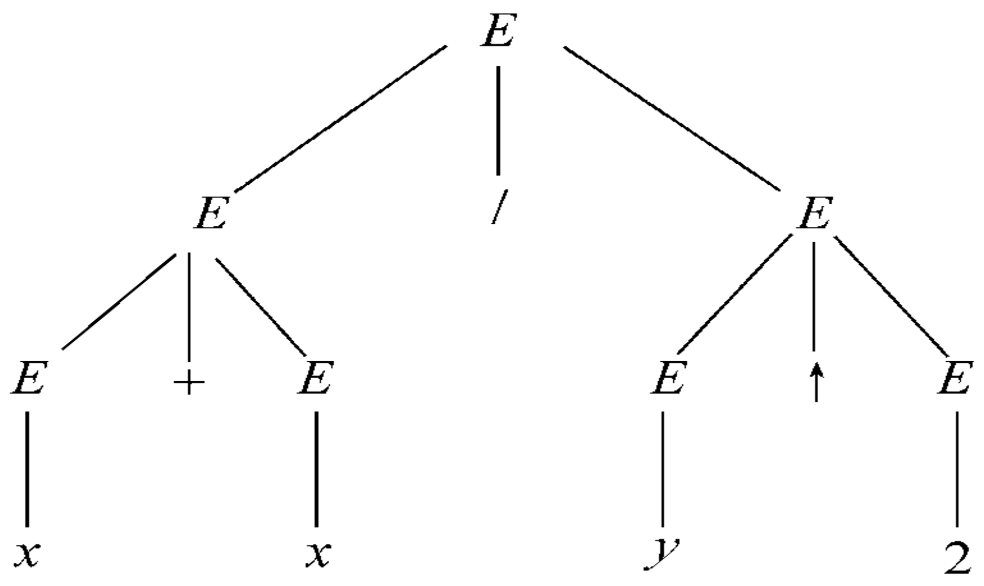
$$
\begin{aligned}
E &\Rightarrow E+E \\
&\Rightarrow x+E \\
&\Rightarrow x+E/E \\
&\Rightarrow x+x/E \\
&\Rightarrow x+x/E**E \\
&\Rightarrow x+x/y**E \\
&\Rightarrow x+x/y**2
\end{aligned}
$$



6.1.2 二义性

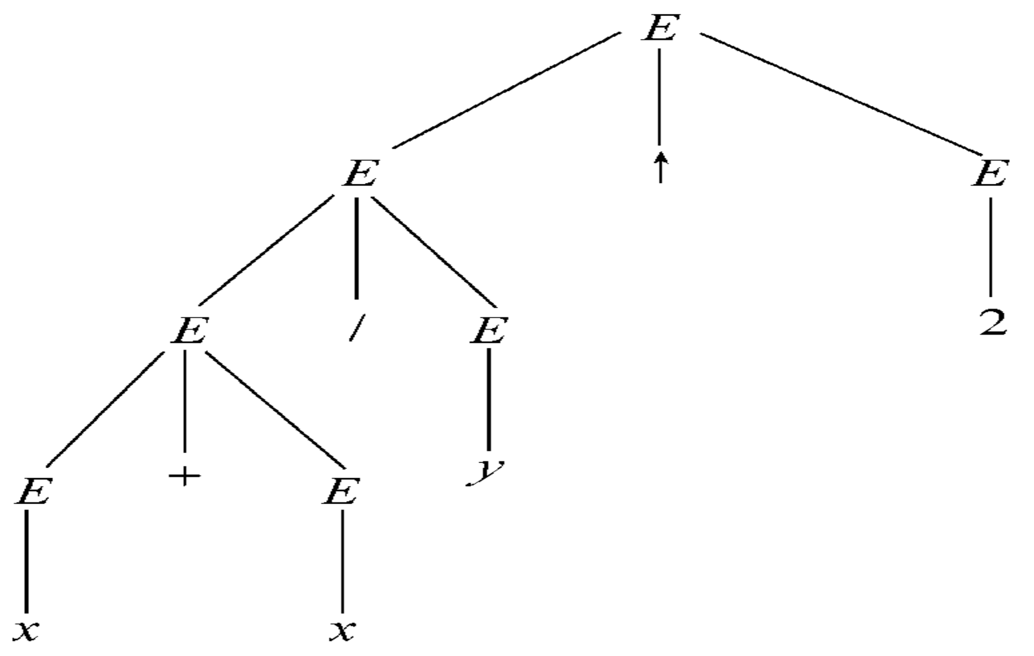
续例6-3 考虑算数表达式的文法$\mathbf{G_{12}}$，句型$\boldsymbol{x+x/y}**\mathbf{2}$在文法$\mathbf{G_{12}}$中有三个不同的最左派生。

$$
\begin{aligned}
E &\Rightarrow E/E \\
&\Rightarrow E+E/E \\
&\Rightarrow x+E/E \\
&\Rightarrow x+x/E \\
&\Rightarrow x+x/E ** E \\
&\Rightarrow x+x/y ** E \\
&\Rightarrow x+x/y ** 2
\end{aligned}
$$



续例6-3 考虑算数表达式的文法G_{12}，句型$x + x/y ** 2$在文法G_{12}中有三个不同的最左派生。

$$
\begin{aligned}
E &\Rightarrow E ** E \\
&\Rightarrow E/E ** E \\
&\Rightarrow E + E/E ** E \\
&\Rightarrow x + E/E ** E \\
&\Rightarrow x + x/E ** E \\
&\Rightarrow x + x/y ** E \\
&\Rightarrow x + x/y ** 2
\end{aligned}
$$



(c)

6.1.2 二义性

<u>定义6-6</u> 二义性(ambiguity)

设有CFG $G = (V, T, P, S)$，如果存在$\omega \in L(G)$，ω至少有两棵不同的派生树，则称G是二义性的。否则，G为非二义性的。

对于一个语言，在产生它的众多文法中，有的可能是二义性的，有的则可能是非二义性的。没有一个一般的方法来证明一个文法是不是二义性的。也就是说，判定给定 CFG G是否为二义性的问题是一个不可解的问题。

6.1.2 二义性

例6-4 设$L=\{0^n1^n2^m3^m|n,m\geq 1\}\cup\{0^n1^m2^m3^n|n,m\geq 1\}$可以用如下文法产生语言$L$。

G： $S \to AB|0C3$

$A \to 01|0A1$

$B \to 23|2B3$

$C \to 0C3|12|1D2$

$D \to 12|1D2$

不难找到句子00112233的两个不同的最左派生

$$S \Rightarrow AB \Rightarrow 0A1B \Rightarrow 0011B \Rightarrow 00112B3 \Rightarrow 00112233$$

$$S \Rightarrow 0C3 \Rightarrow 00C33 \Rightarrow 001D233 \Rightarrow 00112233$$

很容易可以画出这两个最左派生对应的不同派生树，因此G是二义性的。实际上对于L中形如 $0^n1^n2^n3^n$ 的句子，都有不同的派生树存在。语言L不存在非二义性的文法。

6.1.2 二义性

<u>定义6-7</u> 固有二义性(ambiguity)

如果语言L不存在非二义性文法，则称L是固有二义性的，又称L是先天二义性的。

定义6-6和定义6-7表明：

- 文法可以是二义性的。
- 语言可以是固有二义性的。