- 上下文无关文法与上下文无关语言

上下文无关文法与上下文无关语言

- 上下文无关文法的基本概念
- 归约与推导
- 上下文无关语言
- 文法与语言的 *Chomsky* 分类
- 语法分析树
- 归约、推导与分析树之间关系
- 文法和语言的二义性

上下文无关文法的基本概念

✧ 回顾：在第一讲中介绍过如下内容

设 $\Sigma = \{ 0, 1 \}$， $L = \{ 0^n 1^n \mid n \geq 1 \}$，如 0011, 000111, $01 \in L$, 而 10, 1001 , ε, $010 \notin L$.

如下是一个可接受该语言的上下文无关文法:

$S \rightarrow 01$

$S \rightarrow 0S1$

上下文无关文法的基本概念

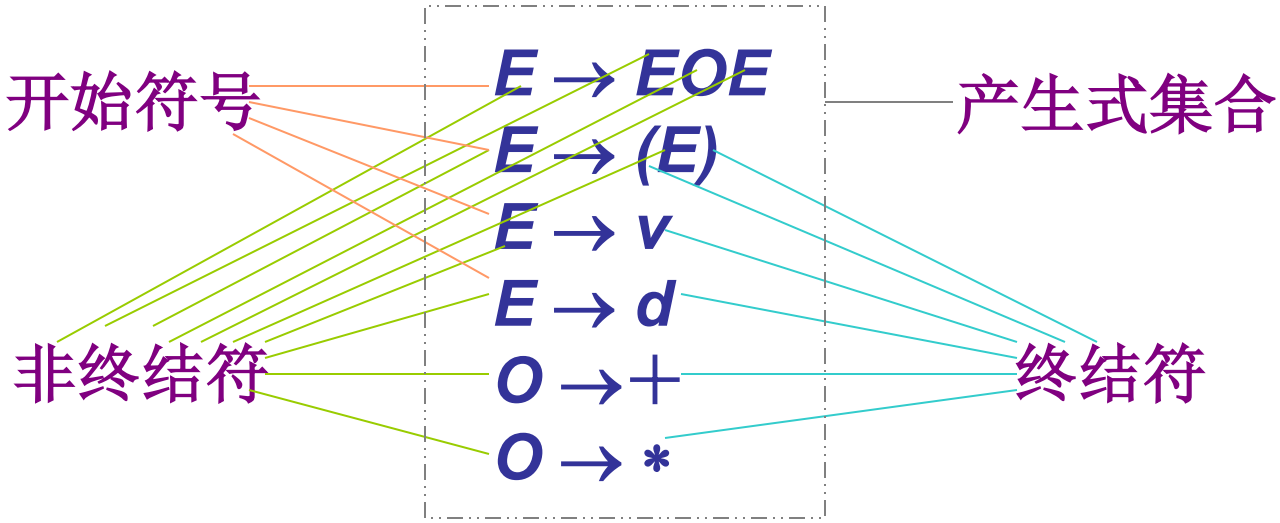
✧ 另一个例子

$E \rightarrow EOE$
$E \rightarrow (E)$
$E \rightarrow v$
$E \rightarrow d$
$O \rightarrow +$
$O \rightarrow *$

上下文无关文法的基本概念

✧ 上下文无关文法*(context-free grammars)*的四个基本要素

1. 终结符*(terminals)*的集合 有限符号集，相当于字母表
2. 非终结符*(nonterminals)*的集合 有限变量符号的集合
3. 开始符号*(start symbol)* 一个特殊的非终结符
4. 产生式*(productions)*的集合 形如：*<head> → <body>*



上下文无关文法的基本概念

✧ 上下文无关文法的形式定义

一个上下文无关文法 ***CFG (context-free grammars)***
是一个四元组 $G = (V, T, P, S)$.

非终结符的集合 — V

终结符的集合 — T

产生式的集合 — P

开始符号 — S

满足

$V \cap T = \Phi$

$S \in V$

产生式形如 $A \to \alpha$, 其中 $A \in V$, $\alpha \in (V \cup T)^*$

上下文无关文法的基本概念

✧ 上下文无关文法举例

(1) CFG $G_{01} = (\{S\}, \{0,1\}, P, S)$. 其中产生式集合 P 为

$$S \to 01$$
$$S \to 0S1$$

(2) CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$.
其中产式集合 P 为

$$E \to EOE$$
$$E \to (E)$$
$$E \to v$$
$$E \to d$$
$$O \to +$$
$$O \to *$$

上下文无关文法的基本概念

✧ 产生式集合的缩写记法

形如 $A \to \alpha_1, A \to \alpha_2, \ldots, A \to \alpha_n$ 的产生式集合可简缩记为 $A \to \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_n$，如

$S \to 01$
$S \to 0S1$

$S \to 01 \mid 0S1$

$E \to EOE$
$E \to (E)$
$E \to v$
$E \to d$
$O \to +$
$O \to *$

$E \to EOE \mid (E) \mid v \mid d$
$O \to + \mid *$

归约与推导

- 用于推理字符串是否属于文法所定义的语言

  一种是自下而上的方法，称为递归推理（*recursive inference*），递归推理的过程习称为归约；另一种是自上而下的方法，称为推导（*derivation*）

- 归约过程

  将产生式右部（*body*）形式的符号串替换为产生式左部（*head*）的符号

- 推导过程

  将产生式左部的符号替换为产生式右部的符号串

✧ 归约过程举例

对于CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$，$P$ 为

（1）$E \to EOE$

（2）$E \to (E)$

（3）$E \to v$

（4）$E \to d$

（5）$O \to +$

（6）$O \to *$

递归推理出字符串 $v*(v+d)$ 的一个归约过程为

$$v*(v+d) \xrightarrow{(4)} v*(v+E) \xrightarrow{(6)} vO(v+E) \xrightarrow{(3)} vO(E+E) \xrightarrow{(5)} vO(EOE) \xrightarrow{(1)} vO(E) \xrightarrow{(2)} vOE \xrightarrow{(3)} EOE \xrightarrow{(1)} E$$

小测试

对于CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$，$P$ 为

（1）$E \rightarrow EOE$
（2）$E \rightarrow (E)$
（3）$E \rightarrow v$
（4）$E \rightarrow d$
（5）$O \rightarrow +$
（6）$O \rightarrow *$

以下哪些字符串可以归约为E？

(A) $(v + d) * (d + d)$

(B) $(v + *$

(C) $v + d * d$

(D) $(v * d) + d * v)$

小测试

对于CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$，$P$ 为

（1）$E \rightarrow EOE$
（2）$E \rightarrow (E)$
（3）$E \rightarrow v$
（4）$E \rightarrow d$
（5）$O \rightarrow +$
（6）$O \rightarrow *$

以下哪些字符串可以归约为E？

$$(v+d)*(d+d) \xrightarrow{(3)} (E+d)*(d+d) \xrightarrow{(5)}$$
$$(EOd)*(d+d) \xrightarrow{(4)} (EOE)*(d+d) \xrightarrow{(1)}$$
$$(E)*(d+d) \xrightarrow{(2)} E*(d+d) \xrightarrow{(6)} EO(d+d)$$
$$\ldots \xrightarrow{(2)} EOE \xrightarrow{(1)} E$$

小测试

对于CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$，$P$ 为

（1）$E \to EOE$
（2）$E \to (E)$
（3）$E \to v$
（4）$E \to d$
（5）$O \to +$
（6）$O \to *$

以下哪些字符串可以归约为E？

$$v + d * d \xrightarrow{(3)} E + d * d \xrightarrow{(5)} E\,O\,d * d \xrightarrow{(4)} E\,O\,E * d$$
$$\xrightarrow{(1)} E * d \xrightarrow{(6)} E\,O\,d \xrightarrow{(4)} E\,O\,E \xrightarrow{(1)} E$$

归约与推导

✧ 推导过程举例

对于CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$，$P$ 为

（1）$E \to EOE$
（2）$E \to (E)$
（3）$E \to v$
（4）$E \to d$
（5）$O \to +$
（6）$O \to *$

从开始符号到字符串 $v*(v+d)$ 的一个推导过程为

$$E \xrightarrow{(1)} EOE \xrightarrow{(6)} E*E \xrightarrow{(2)} E*(E) \xrightarrow{(3)} v*(E)$$

$$\xrightarrow{(1)} v*(EOE) \xrightarrow{(5)} v*(E+E) \xrightarrow{(3)} v*(v+E) \xrightarrow{(4)} v*(v+d)$$

小测试

对于CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$，$P$为

（1）$E \rightarrow EOE$
（2）$E \rightarrow (E)$
（3）$E \rightarrow v$
（4）$E \rightarrow d$
（5）$O \rightarrow +$
（6）$O \rightarrow *$

以下哪些字符串可以从E推导出来？

(A) $(v + d) * (d + d)$

(B) $(v + *$

(C) $v + d * d$

(D) $(v * d) + d * v)$

归约与推导

✧ 推导关系

对于 CFG $G=(V,T,P,S)$，上述推导过程可用关系$\Rightarrow_G$描述. 设 $\alpha,\beta\in(V\cup T)^*$，$A\to\gamma$ 是一个产生式，则定义

$$\alpha A\beta \Rightarrow_G \alpha\gamma\beta .$$

若 G 在上下文中是明确的，则简记为$\alpha A\beta \Rightarrow \alpha\gamma\beta$.

✧ 扩展推导关系到自反传递闭包

定义上述关系的传递闭包，记为 $\overset{*}{\Rightarrow}_G$，可归纳定义如下：

基础　对任何 $\alpha\in(V\cup T)^*$，满足 $\alpha \overset{*}{\Rightarrow}_G \alpha$.

归纳　设 $\alpha,\beta,\gamma\in(V\cup T)^*$，若 $\alpha \overset{*}{\Rightarrow}_G \beta$，$\beta \Rightarrow_G \gamma$ 成立，则

$$\alpha \overset{*}{\Rightarrow}_G \gamma .$$

归约与推导

✧ 最左推导（*leftmost derivations*）

若推导过程的每一步总是替换出现在最左边的非终结符，则这样的推导称为**最左推导**. 为方便，最左推导关系用$\underset{lm}{\Rightarrow}$表示，其传递闭包用$\overset{*}{\underset{lm}{\Rightarrow}}$表示.

如对于文法G_{exp}，下面是关于 $v*(v+d)$ 的一个最左推导：

$$E \overset{*}{\underset{lm}{\Rightarrow}} EOE \overset{*}{\underset{lm}{\Rightarrow}} vOE \overset{*}{\underset{lm}{\Rightarrow}} v*E$$
$$\overset{*}{\underset{lm}{\Rightarrow}} v*(E) \overset{*}{\underset{lm}{\Rightarrow}} v*(EOE) \overset{*}{\underset{lm}{\Rightarrow}} v*(vOE)$$
$$\overset{*}{\underset{lm}{\Rightarrow}} v*(v+E) \overset{*}{\underset{lm}{\Rightarrow}} v*(v+d)$$

$E \to EOE$
$E \to (E)$
$E \to v$
$E \to d$
$O \to +$
$O \to *$

归约与推导

✧ 最右推导（*rightmost derivations*）

若推导过程的每一步总是替换出现在最右边的非终结符，则这样的推导称为**最右推导**. 为方便，最右推导关系用$\underset{rm}{\Rightarrow}$表示，其传递闭包用$\overset{*}{\underset{rm}{\Rightarrow}}$表示.

如对于文法G_{exp}，下面是关于 $v*(v+d)$ 的一个最右推导：

$$E \overset{*}{\underset{rm}{\Rightarrow}} EOE \overset{*}{\underset{rm}{\Rightarrow}} EO(E) \overset{*}{\underset{rm}{\Rightarrow}} EO(EOE)$$

$$\overset{*}{\underset{rm}{\Rightarrow}} EO(EOd) \overset{*}{\underset{rm}{\Rightarrow}} EO(E+d) \overset{*}{\underset{rm}{\Rightarrow}} EO(v+d)$$

$$\overset{*}{\underset{rm}{\Rightarrow}} E*(v+d) \overset{*}{\underset{rm}{\Rightarrow}} v*(v+d)$$

$$E \to EOE$$
$$E \to (E)$$
$$E \to v$$
$$E \to d$$
$$O \to +$$
$$O \to *$$

最左推导与最右推导

例1：给定上下文无关文法

$$S \to A1B$$
$$A \to 0A \mid \epsilon$$
$$B \to 0B \mid 1B \mid \epsilon$$

给出串 00101 的最左推导和最右推导。

解：最左推导如下。

步骤	产生式	字符串
0		S
1	$S \to A1B$	$A1B$
2	$A \to 0A$	$0A1B$
3	$A \to 0A$	$00A1B$
4	$A \to \epsilon$	$001B$

最左推导与最右推导

例1：给定上下文无关文法

$$S \to A1B$$
$$A \to 0A \mid \epsilon$$
$$B \to 0B \mid 1B \mid \epsilon$$

给出串 00101 的最左推导和最右推导。

解：最左推导如下。

步骤	产生式	字符串
5	$B \to 0B$	$0010B$
6	$B \to 1B$	$00101B$
7	$B \to \epsilon$	00101

最左推导与最右推导

例1：给定上下文无关文法

$$S \to A1B$$
$$A \to 0A \mid \epsilon$$
$$B \to 0B \mid 1B \mid \epsilon$$

给出串 00101 的最左推导和最右推导。

解：最右推导如下。

步骤	产生式	字符串
0		S
1	$S \to A1B$	$A1B$
2	$B \to 0B$	$A10B$
3	$B \to 1B$	$A101B$
4	$B \to \epsilon$	$A101$

最左推导与最右推导

例1：给定上下文无关文法

$$S \to A1B$$
$$A \to 0A \mid \epsilon$$
$$B \to 0B \mid 1B \mid \epsilon$$

给出串 00101 的最左推导和最右推导。

解：最右推导如下。

步骤	产生式	字符串
5	$A \to 0A$	$0A101$
6	$A \to 0A$	$00A101$
7	$A \to \epsilon$	00101

✧ 句型 *(sentential forms)*

设 ***CFG G = (V, T, P , S)***，称 $\alpha \in (V \cup T)^*$ 为 ***G*** 的一个句型，当且仅当 $S \overset{*}{\Rightarrow} \alpha$.

若 $S \overset{*}{\underset{lm}{\Rightarrow}} \alpha$，则 α 是一个左句型（***left-sentential form***）；

若 $S \overset{*}{\underset{rm}{\Rightarrow}} \alpha$，则 α 是一个右句型（***right-sentential form***）.

若句型 $\alpha \in T^*$，则称 α 为一个句子（***sentence***）.

举例：我想预订一个**A**月**B**日从**X**到**Y**的航班

上下文无关语言

✧ 上下文无关文法的语言

设 *CFG* $G = (V, T, P, S)$，定义 G 的语言为

$$L(G) = \{ w \mid w \in T^* \wedge S \overset{*}{\underset{G}{\Rightarrow}} w \}$$

CFG $G_{exp} = (\{E,O\}, \{ (,), +, *, v, d \}, P, E)$ 的语言 $L(G_{exp}) = ?$

$E \to EOE$
$E \to (E)$
$E \to v$
$E \to d$
$O \to +$
$O \to *$

归纳定义：

1 基础　$v, d \in L(G_{exp})$

2 归纳

if $e \in L(G_{exp})$, then $(e) \in L(G_{exp})$

if $e_1, e_2 \in L(G_{exp})$, then $e_1 + e_2 \in L(G_{exp})$

if $e_1, e_2 \in L(G_{exp})$, then $e_1 * e_2 \in L(G_{exp})$

上下文无关语言

✧ 上下文无关语言（*context-free languages*）

如果一个语言 ***L*** 是某个 ***CFG G*** 的语言，即

L(G) = L，

则是上下文无关语言.

例2：请给出下列语言的一个上下文无关文法

$$L = \{0^n 1^n | n \geq 1\}$$

解：该语言具有递归的特点，所以可以先写出递归式

$$S \to 0S1$$

由于上式是一个无限递归，必须要增加基础的情况

$$S \to 01$$

这样也满足了 $n \geq 1$ 的条件。因此最终的文法如下

$$S \to 0S1 \mid 01$$

例3：请给出下列语言的一个上下文无关文法

$$L = \{0^n 1^n | n \geq 0\}$$

解：这道题和例1非常像，唯一的区别在于把 $n \geq 1$ 改成了 $n \geq 0$ 。因此，递归部分不变，只需修改基础部分

$$S \rightarrow 0S1 \mid \epsilon$$

例4：请给出下列语言的一个上下文无关文法

$$L = \{a^m b^m c^n d^n | m \geq 1, n \geq 1\}$$

解：该语言的字符串可以分成相对对立的两部分，左部由 a 和 b 组成，可以表示为

$$A \rightarrow aAb \mid ab$$

右部由 c 和 d 组成，可以表示为

$$B \rightarrow cBd \mid cd$$

最后还需要第三个式子把左右部分拼接起来

$$S \rightarrow AB$$

例5：请给出下列语言的一个上下文无关文法

$$L = \{a^n b^m c^m d^n | m \geq 1, n \geq 1\}$$

解：该语言的字符串可以分为内外两个相对对立的部分。

内层的字符串由 b 和 c 组成

$$D \rightarrow bDc \mid bc$$

外面再包上一层成对出现的 a 和 d

$$C \rightarrow aCd$$

需要注意 C 的基础情况不能写成 $C \rightarrow ad$，因为 C 内部的字符串由 b 和 c 组成。怎么办？

例5：请给出下列语言的一个上下文无关文法

$$L = \{a^n b^m c^m d^n | m \geq 1, n \geq 1\}$$

解：内层和外层的关系可以这样表示

$$C \rightarrow aCd \mid D$$

为了至少生成一对 a 和 d，需要在句子级别做修改

$$S \rightarrow aCd$$

这样保证了先生成至少一对 a 和 d，然后有两种可能扩展 C：进入递归不断生成外层的字符串，或者变成 D 开始生成内层的字符串。

例5：请给出下列语言的一个上下文无关文法

$$L = \{a^n b^m c^m d^n | m \geq 1, n \geq 1\}$$

解：综上所述，最终的文法为

$$S \rightarrow aCd$$

$$C \rightarrow aCd \mid D$$

$$D \rightarrow bDc \mid bc$$

例6：请给出下列语言的一个上下文无关文法

$$L = \{a^m b^m c^n d^n | m \geq 1, n \geq 1\} \cup \{a^n b^m c^m d^n | m \geq 1, n \geq 1\}$$

解：这道题是例4和例5的结合。把两个文法合并起来即可

$$\begin{aligned} S &\rightarrow AB \mid aCd \\ A &\rightarrow aAb \mid ab \\ B &\rightarrow cBd \mid cd \\ C &\rightarrow aCd \mid D \\ D &\rightarrow bDc \mid bc \end{aligned}$$

例7：请给出下列语言的一个上下文无关文法

$$\{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge m \neq n\}$$

解：这道题的难点在于要求 $m \neq n$，受例6的启发，可以把该语言拆成两个部分的并集

$$\{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge m > n\} \cup$$
$$\{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge m < n\}$$

首先来看第一部分，如何来实现 $m > n$，也就是 a 的数量至少要比 b 多1个？需要注意在这种情况下，实际上是要求：$m \geq 1 \wedge n \geq 0 \wedge m > n$

例7：请给出下列语言的一个上下文无关文法

$$\{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge m \neq n\}$$

解：首先要给 a “特权”，在生成句子时至少先包含1个a

$$S \rightarrow aA$$

$$A \rightarrow aA$$

接着是正常的递归部分：

$$A \rightarrow aAb$$

由于 b 的数量可以为0，因此基础部分应该写成

$$A \rightarrow \epsilon$$

例7：请给出下列语言的一个上下文无关文法

$$\{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge m \neq n\}$$

解：再来考虑第二部分

$$\{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge m < n\}$$

可以类似地进行处理

$$S \to Bb$$

$$B \to Bb$$

$$B \to aBb$$

$$B \to \epsilon$$

例7：请给出下列语言的一个上下文无关文法

$$\{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge m \neq n\}$$

解：最后需要将两部分合并起来

$$S \rightarrow aA \mid Bb$$

$$A \rightarrow aA \mid aAb \mid \epsilon$$

$$B \rightarrow Bb \mid aBb \mid \epsilon$$

例8：请给出下列语言的一个上下文无关文法

$$L = \{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge 3m \geq n \geq 2m\}$$

解：这道题一眼看上去感觉无从下手，如何实现区间关系？

不管那么多，不妨先想办法实现 $n = 2m$

$$S \rightarrow aSbb$$

显然，这种递归形式可以保证2倍的关系。那么3倍的关系也很容易实现了

$$S \rightarrow aSbbb$$

那么怎么实现区间关系呢？

例8：请给出下列语言的一个上下文无关文法

$$L = \{a^m b^n | m \geq 0 \wedge n \geq 0 \wedge 3m \geq n \geq 2m\}$$

解：幸运的是，当文法同时包含这两条规则时，恰好实现了区间关系

$$S \rightarrow aSbb$$

$$S \rightarrow aSbbb$$

这是因为两条规则的混用恰好能实现区间关系。加入基础部分，就得到了最终的文法

$$S \rightarrow aSbb \mid aSbbb \mid \epsilon$$

例9：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, \text{count}(w, a) = \text{count}(w, b)\}$$

解：在前面的例子里，a 和 b 的顺序是指定的，而这道题就难很多，因为对顺序没有要求，只要求数量相等。可以考虑三种情况。第1种情况是空串 $S \to \epsilon$ 。

第2种情况是字符串以a 开头，则在剩下的字符串中 b 的数量必然比 a 多1个。对剩下的字符串重新计数，当第1次遇到 b 的数量比 a 多1个时停下来，可将剩下的字符串分成三部分：左部和右部中a的数量和 b 相等，中间是一个单独的分割点b 。

例9：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a,b\}^*, \text{count}(w,a) = \text{count}(w,b)\}$$

解：不妨看几个具体的例子

$aabbab$　　$ababab$　　$aaaabbbb$

左部　中部　右部

注意：中间部分作为分割点的 b 是唯一的。

因此，很容易用递归的形式来定义

$$S \to aSbS$$

例9：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, \text{count}(w, a) = \text{count}(w, b)\}$$

解：第3种情况是字符串以b 开头，可以类似地递归定义

$$S \to bSaS$$

综上所述，该文法定义为

$$S \to aSbS \mid bSaS \mid \epsilon$$

例10：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a,b\}^*, |\text{count}(w,a) - \text{count}(w,b)| = 2\}$$

解：由于使用了绝对值，因此要区分两种情况

$$L = \{w | w \in \{a,b\}^*, \text{count}(w,a) - \text{count}(w,b) = 2\} \cup$$
$$\{w | w \in \{a,b\}^*, \text{count}(w,b) - \text{count}(w,a) = 2\}$$

首先考虑第一种情况，即 a 的数量比 b 多2个。依然可以采用之前的分割技术。从左往右扫描，计算 a 的数量和 b 的数量，当 a 的数量第1次比 b 的数量多1时，将该 a 确定为第一个分割点。

例10：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a,b\}^*, |\text{count}(w,a) - \text{count}(w,b)| = 2\}$$

解：然后将计数清零，类似地去找第2个分割点 a 。这两个分割点可以将字符串切成5个部分

$$bbaa|a|bbaa|a|ba$$

蓝色部分是切割点，红色部分两个元素的个数相等。

因此可以写成：

$$A \to CaCaC$$
$$C \to aCbC \mid bCaC \mid \epsilon$$

例10：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, |\text{count}(w, a) - \text{count}(w, b)| = 2\}$$

解：第二种情况是对称的，可以类似地处理。

因此，最终的文法是

$$S \to A \mid B$$
$$A \to CaCaC$$
$$B \to CbCbC$$
$$C \to aCbC \mid bCaC \mid \epsilon$$

例11：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a,b\}^*, \text{count}(w,a) \neq \text{count}(w,b)\}$$

解：可以用前面的策略，先把文法拆成两部分

$$L = \{w | w \in \{a,b\}^*, \text{count}(w,a) > \text{count}(w,b)\} \cup$$
$$\{w | w \in \{a,b\}^*, \text{count}(w,a) < \text{count}(w,b)\}$$

首先考虑第一部分，即 a 的数量比 b 多。该怎么下手？

随机列几个符合条件的字符串，仔细观察

$$aab \quad aabaaba \quad baaab \quad abaaaaba$$

例11：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, \text{count}(w, a) \neq \text{count}(w, b)\}$$

解：假设对于每个字符串从左往右扫描，边扫描边数 a 和 b 的数量，遇到第一次 a 的数量比 b 多1就停下来。这样可以把字符串分成三部分，左边的部分 a 的数量和 b 相等，中间部分只有1个 a，右边的部分 a 的数量不低于 b

aab　　$aabaaba$　　$baaab$　　$abaaaaba$

注意：中间部分作为分割点的 a 是唯一的。

例11：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, \text{count}(w, a) \neq \text{count}(w, b)\}$$

解：如何实现 a 的数量不低于 b？可以在数量相等的基础上增加 a 的数量。

$$D \to aDbD \mid bDaD \mid aD \mid Da \mid \epsilon$$

因此，第一种情况可以表示为

$$A \to CaD$$
$$C \to aCbC \mid bCaC \mid \epsilon$$
$$D \to aDbD \mid bDaD \mid aD \mid Da \mid \epsilon$$

例11：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, \text{count}(w, a) \neq \text{count}(w, b)\}$$

解：第二种情况是对称的，可以类似地处理。

因此，最终的文法是

$$S \to A \mid B$$
$$A \to CaD$$
$$B \to CbE$$
$$C \to aCbC \mid bCaC \mid \epsilon$$
$$D \to aDbD \mid bDaD \mid aD \mid Da \mid \epsilon$$
$$E \to aEbE \mid bEaE \mid bE \mid Eb \mid \epsilon$$

例11：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a,b\}^*, \text{count}(w,a) \neq \text{count}(w,b)\}$$

解：但是上面的答案比较冗余，尤其是后三个式子有些重复。能不能写得更简单一些？以第一种情况为例，只要分割点两侧都满足a的数量不低于b即可。

$$abaaaaba$$

这样可以简化一些

$$A \to CaC$$

$$C \to aCbC \mid bCaC \mid aC \mid Ca \mid \epsilon$$

例11：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, \text{count}(w, a) \neq \text{count}(w, b)\}$$

解：按照这种方式得到的文法比之前少一个式子

$$S \to A \mid B$$
$$A \to CaC$$
$$B \to DbD$$
$$C \to aCbC \mid bCaC \mid aC \mid Ca \mid \epsilon$$
$$D \to aDbD \mid bDaD \mid bD \mid Db \mid \epsilon$$

例11：请给出下列语言的一个上下文无关文法

$$L = \{w | w \in \{a, b\}^*, \text{count}(w, a) \neq \text{count}(w, b)\}$$

解：不过这种方法的分割点不是唯一的。以下分割都合法。

$abaaaaba$

$abaaaaba$

$abaaaaba$

$abaaaaba$

$abaaaaba$

$abaaaaba$

✧ 证明给定语言 *L* 是某个文法 *G* 的语言

– 一般步骤

- *if* $w \in L$ *then* $w \in L(G)$
- *if* $w \in L(G)$ *then* $w \in L$.

对于前者，多数情况下可以归纳于 *w* 的长度 | *w* |；
对于后者，一般情况下可以归纳于推导*w*的步数.

上下文无关语言证明

例12：给定一个上下文无关文法 G

$$P \to 0P0 \mid 1P1 \mid 0 \mid 1 \mid \epsilon$$

证明 $L(G)$ 是字母表为 $\{0,1\}$ 上回文的集合。

解：首先要解题，所谓回文就是两边对称的字符串。

$$0110 \qquad 100001 \qquad 010010$$

题目可以重新表述为

$$\forall w \in \{0,1\}^*, w \in L(G) \Leftrightarrow w = w^R$$

其中 w^R 是 w 的逆序字符串。例如：

$$w = 101010 \qquad w^R = 010101$$

需要从两个方向进行证明。

例12：给定一个上下文无关文法 G

$$P \to 0P0 \mid 1P1 \mid 0 \mid 1 \mid \epsilon$$

证明 $L(G)$ 是字母表为 $\{0,1\}$ 上回文的集合。

解：首先证明正向命题，归纳于推导的步数。

$$\forall w \in \{0,1\}^*, w \in L(G) \Rightarrow w = w^R$$

（基础）考虑以下三个产生式

$$P \to 0 \qquad P \to 1 \qquad P \to \epsilon$$

显然 0 、1 和 ϵ 都属于 $L(G)$ 而且都是回文。命题成立。

（归纳）假设 $P \overset{*}{\Rightarrow} x$ 可以使用 n 步推导完成，x 是一个字符串且是回文。下面来考虑 $n+1$ 步推导。

上下文无关语言证明

例12：给定一个上下文无关文法 G

$$P \to 0P0 \mid 1P1 \mid 0 \mid 1 \mid \epsilon$$

证明 $L(G)$ 是字母表为 $\{0,1\}$ 上回文的集合。

解：$n+1$ 步推导分为两种情况。

第一种情况是在第一步使用了规则 $P \to 0P0$

$$P \Rightarrow 0P0 \overset{*}{\Rightarrow} 0x0 = w$$

因此，$P \overset{*}{\Rightarrow} w$ 可以在 $n+1$ 步推导内完成且 w 是回文。

第二种情况是在第一步使用了规则 $P \to 1P1$

$$P \Rightarrow 1P1 \overset{*}{\Rightarrow} 1x1 = w$$

因此，$P \overset{*}{\Rightarrow} w$ 可以在 $n+1$ 步推导内完成且 w 是回文。

上下文无关语言证明

例12：给定一个上下文无关文法 G

$$P \to 0P0 \mid 1P1 \mid 0 \mid 1 \mid \epsilon$$

证明 $L(G)$ 是字母表为 $\{0,1\}$ 上回文的集合。

解：然后证明反向命题，归纳于字符串的长度。

$$\forall w \in \{0,1\}^*, w \in L(G) \Leftarrow w = w^R$$

要证明 $w \in L(G)$，其实是要证明 $P \overset{*}{\Rightarrow} w$。

（基础）字符串长度可以为0或者1，那么此时 w 一定是 0、1 或 ϵ。根据以下三个产生式

$$P \to 0 \qquad P \to 1 \qquad P \to \epsilon$$

可知 $P \overset{*}{\Rightarrow} w$。

例12：给定一个上下文无关文法 G

$$P \to 0P0 \mid 1P1 \mid 0 \mid 1 \mid \epsilon$$

证明 $L(G)$ 是字母表为 $\{0,1\}$ 上回文的集合。

解：（归纳）假设命题在 $|w| = n$ 时是成立的，下面考虑 $|w| = n+1$ 时的情况。由于长度0和1已经讨论过，假设 w 的长度大于1并且是回文，则首字符和尾字符必然相同。需要考察两种情况。

第一种情况是首字符和尾字符都是0，则该字符串可以写成 $w = 0x0$ 。由于 $|x| = n-1$，根据归纳假设，可以得到：$P \overset{*}{\Rightarrow} x$。

上下文无关语言证明

例12：给定一个上下文无关文法 G

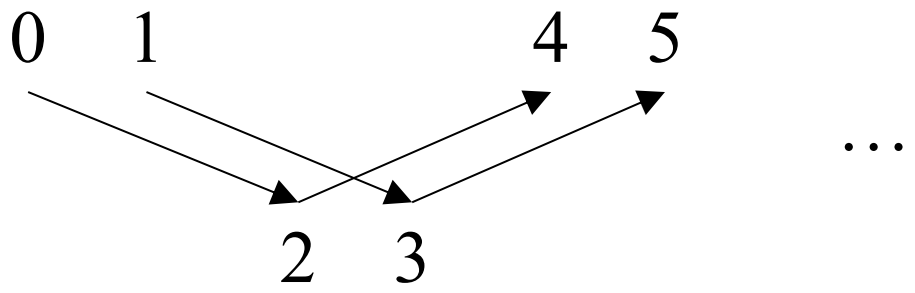
$$P \to 0P0 \mid 1P1 \mid 0 \mid 1 \mid \epsilon$$

证明 $L(G)$ 是字母表为 $\{0,1\}$ 上回文的集合。

解：因此，必然存在一个推导

$$P \Rightarrow 0P0 \overset{*}{\Rightarrow} 0x0 = w$$

第二种情况是首尾字符是1，也是类似地处理。



例13：给定一个上下文无关文法 G

$$S \to aS \mid aSb \mid \epsilon$$

证明 $L(G) = \{a^m b^n | m \geq n \geq 0\}$。

解：首先证明当 w 具备 $a^m b^n$ 的形式且满足 $m \geq n \geq 0$ 时，必然有 $S \stackrel{*}{\Rightarrow} w$。

（基础）$|w| = 0$，即 $w = \epsilon$。基于产生式

$$S \to \epsilon$$

可知 $S \stackrel{*}{\Rightarrow} w$。

（归纳）假设命题在 $|w| = k$ 时成立。下面考虑在 $|w| = k + 1$ 时的情况。

例13：给定一个上下文无关文法 G

$$S \to aS \mid aSb \mid \epsilon$$

证明 $L(G) = \{a^m b^n | m \geq n \geq 0\}$ 。

解：需要分两种情况来讨论。

第一种情况是 $m > n$ 。可以将 w 写成 $w = ax$ 的形式，其中 x 中 a 的数量不小于 b 。由于 $|x| = k$，根据归纳假设，可以得到 $S \stackrel{*}{\Rightarrow} x$ 。因此存在推导生成 w ： $S \Rightarrow aS \stackrel{*}{\Rightarrow} ax = w$ 。

第二种情况是 $m = n$。可以将 w 写成 $w = axb$ 的形式，其中 x 中 a 的数量等于 b。由于 $|x| = k - 1$，根据归纳假设，可以得到 $S \stackrel{*}{\Rightarrow} x$ 。

例13：给定一个上下文无关文法 G

$$S \to aS \mid aSb \mid \epsilon$$

证明 $L(G) = \{a^m b^n | m \geq n \geq 0\}$ 。

解：因此存在推导 $S \Rightarrow aSb \overset{*}{\Rightarrow} axb = w$ 。

然后证明当 $w \in L(G)$ 时，w 必然具备 $a^m b^n$ 的形式并且满足 $m \geq n \geq 0$ 。归纳于推导的步数。

（基础）考虑推导步数为1的情况，即使用 $S \to \epsilon$ 。则 $w = a^0 b^0$ ，满足 $m \geq n \geq 0$ 。

（归纳）假设推导步数为 k 时成立，下面考虑推导步数为 $k+1$ 时的情况。

例13：给定一个上下文无关文法 G

$$S \to aS \mid aSb \mid \epsilon$$

证明 $L(G) = \{a^m b^n | m \geq n \geq 0\}$ 。

解：下面分两种情况来讨论。

第一种情况是第一步使用了规则

$$S \Rightarrow aS \overset{*}{\Rightarrow} ax = w$$

由于生成 x 使用了 k 步推导，根据归纳假设，x 可以写成 $x = a^m b^n$ 的形式且满足 $m \geq n \geq 0$。那么 $w = ax = a^{m+1} b^n$ 显然也满足条件。

例13：给定一个上下文无关文法 G

$$S \to aS \mid aSb \mid \epsilon$$

证明 $L(G) = \{a^m b^n | m \geq n \geq 0\}$ 。

解：第二种情况是第一步使用了规则

$$S \Rightarrow aSb \overset{*}{\Rightarrow} axb = w$$

由于生成 x 使用了 k 步推导，根据归纳假设，x 可以写成 $x = a^m b^n$ 的形式且满足 $m \geq n \geq 0$。那么 $w = axb = a^{m+1} b^{n+1}$ 显然也满足条件。

上下文无关语言证明

例14：给定一个上下文无关文法 G

$$S \to 1 \mid 1T1$$
$$T \to 0 \mid 0S0$$

证明若 $S \overset{*}{\Rightarrow} w$，**则有** $\text{count}(w,1) = \text{count}(w,0) + 1$。

解：仔细观察，可以发现两个产生式相互“纠缠”。因此可以使用互归纳法。引入两个命题：

$$P_1(n): S \overset{*}{\Rightarrow} w \Rightarrow \text{count}(w,1) = \text{count}(w,0) + 1$$
$$P_2(n): T \overset{*}{\Rightarrow} w \Rightarrow \text{count}(w,0) = \text{count}(w,1) + 1$$

可以使用这两个命题相互支撑，归纳于推导步数。

上下文无关语言证明

例14：给定一个上下文无关文法 G

$$S \to 1 \mid 1T1$$
$$T \to 0 \mid 0S0$$

证明若 $S \overset{*}{\Rightarrow} w$，则有 $\text{count}(w,1) = \text{count}(w,0) + 1$。

解：（基础）考察 $P_1(1)$，即当推导步数为1时，只能使用产生式 $S \to 1$，则 $w = 1$，1的数量比0多一个。成立。

再考察 $P_2(1)$，即当推导步数为1时，只能使用产生式 $T \to 0$，则 $w = 0$，0的数量比1多一个。成立。

（归纳）假设 $P_1(n)$ 和 $P_2(n)$ 成立，即在推导步数为 n 时时成立。下面考察推导步数为 $n+1$ 时的情况。

上下文无关语言证明

例14：给定一个上下文无关文法 G

$$S \to 1 \mid 1T1$$
$$T \to 0 \mid 0S0$$

证明若 $S \overset{*}{\Rightarrow} w$，则有 $\text{count}(w,1) = \text{count}(w,0) + 1$。

解：考察 $P_1(n+1)$，假设在推导的第1步使用了产生式 $S \to 1T1$ 时，则有 $S \Rightarrow 1T1 \overset{*}{\Rightarrow} w$。因此，$w$ 可以写成 $w = 1x1$ 的形式。由于推导 $T \overset{*}{\Rightarrow} x$ 使用 n 步推导得到，根据归纳假设 $P_2(n)$ 可知 x 中0的数量比1多一个，因此 w 中1的数量比0多一个。命题成立。

$P_2(n+1)$ 可以类似地处理。

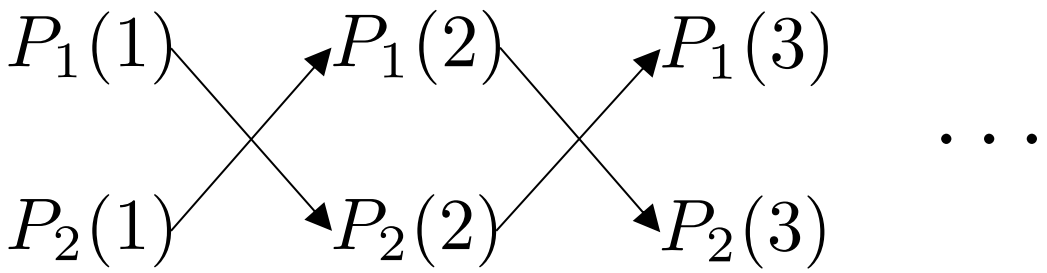
上下文无关语言证明

例14：给定一个上下文无关文法 G

$$S \to 1 \mid 1T1$$
$$T \to 0 \mid 0S0$$

证明若 $S \overset{*}{\Rightarrow} w$，**则有** $\text{count}(w,1) = \text{count}(w,0) + 1$。

解：两个命题的相互支撑关系如下



✧ 证明给定语言*L*是某个文法*G* 的语言

– 举例

!!Exercise 5.1.8 考虑定义了下面的产生式的 CFG *G*：

S → aSbS | bSaS | ε

证明 L(G) 是所有有相同个数的a和b的串的集合。

– 证明思路

用归纳法证明：

- 若串w中包含相同个数的a和b，则w ∈ L(G) .
  （通过对|w|进行归纳来证明w在 L(G) 中，即S ⇒*w）
- 若 w ∈ L(G)，即S ⇒*w，则w中包含相同个数的a和b.
  （对从S到w的推导过程的步数进行归纳 ）

（留作作业）

✧ 证明给定语言*L*是某个文法*G* 的语言

– **举例** 设 *G* 为上下文无关文法，其终结符集合为 {*a*, *b*}，开始符号为*S*，产生式集合如下：

$$S \to \varepsilon \mid aB \mid bA$$
$$A \to a \mid aS \mid bAA$$
$$B \to b \mid bS \mid aBB$$

试证明 $L(G) = \{ w \mid w \in \{a, b\}^*, occur(w, a) = occur(w, b) \}$.
其中，对于符号 *a* 和串 *w*，*occur*(*a*, *w*) 表示 *a* 在 *w* 中出现的次数.

– **证明思路**

用互归纳法证明：对所有的 $w \in \{a, b\}^*$，如下三个等价式成立：

1）$S \Rightarrow^* w$ *iff* $occur(a, w) = occur(b, w)$；
2）$A \Rightarrow^* w$ *iff* $|w|>0 \wedge occur(a, w) = occur(b, w) + 1$；
3）$B \Rightarrow^* w$ *iff* $|w|>0 \wedge occur(b, w) = occur(a, w) + 1$

（留作思考题）

文法与语言的Chomsky分类方法

✧ 文法（grammar）文法是一个四元组

$$G = (V, T, P, S),$$

V、*T*、*P* 及 *S* 的含义如前. ***Chomsky*** 通过对产生式施加不同的限制，把文法分成四种类型，即*0* 型、 *1* 型、 *2* 型和 *3* 型.

文法与语言的Chomsky分类方法

✧ **0** 型文法

0 型文法 $G = (V, T, P, S)$ 的产生式形如 $\alpha \to \beta$，其中 $\alpha, \beta \in (V \cup T)^*$，但 α 中至少包含一个非终结符.

能够用 **0** 型文法定义的语言称为**0** 型语言.

✧ 结论

0 型文法的能力相当于图灵机（*Turing machines*）.

文法与语言的Chomsky分类方法

✧ 1 型文法

1 型文法 $G = (V, T, P, S)$ 的产生式形如 $\alpha \to \beta$，满足 $|\alpha| \le |\beta|$，仅 $S \to \varepsilon$ 例外，且要求 S 不得出现在任何产生式的右部. 1 型文法也称谓上下文有关文法（context-sensitive grammars）.

能够用 1 型文法定义的语言称为 1 型语言 或 上下文有关语言.

与1 型文法的能力相当的一种状态机模型为线性有界自动机.

文法与语言的Chomsky分类方法

✧ 2 型文法

2 型文法 $G = (V, T, P, S)$ 的产生式形如 $A \to \beta$，其中 $A \in V$，$\beta \in (V \cup T)^*$. 2 型文法即上下文无关文法.

能够用 2 型文法定义的语言称为2 型语言，即上下文无关语言.

✧ 结论

与2 型文法的能力相当的一种状态机模型为下推自动机（*Pushdown Automata*）.

文法与语言的Chomsky分类方法

✧ 3 型文法

3 型文法 $G = (V, T, P, S)$ 的产生式形如 $A \to aB$ 或 $A \to a$, 其中 $A, B \in V$， $a \in T \cup \{\varepsilon\}$

3 型文法也称为正规文法.

能够用 3 型文法定义的语言称为3 型语言，即正规语言.

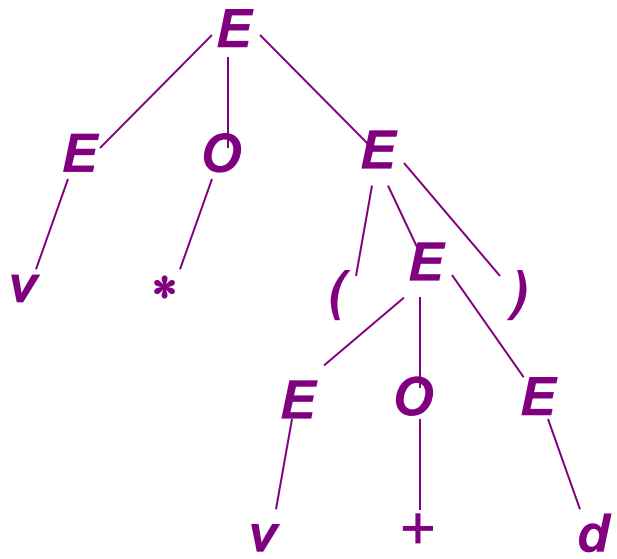
✧ 结论

3 型文法的能力等价于有限状态自动机.

语法分析树

- 归约过程自下而上构造了一棵树

如对于文法G_{exp}，关于 v*(v+d) 的一个归约过程可以认为是构造了如下一棵树：



(1) E → EOE
(2) E → (E)
(3) E → v
(4) E → d
(5) O → +
(6) O → *

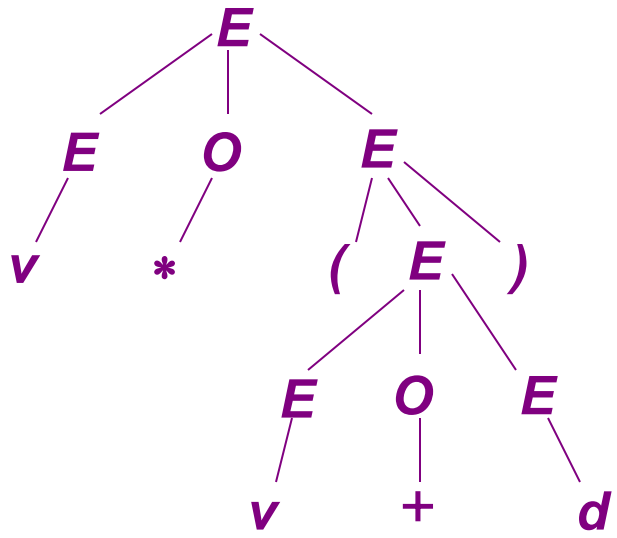
v*(v+d) —(4)→ v*(v+E) —(6)→ vO(v+E) —(3)→ vO(E+E)

—(5)→ vO(EOE) —(1)→ vO(E) —(2)→ vOE —(3)→ EOE —(1)→ E

语法分析树

✧ 推导过程自上而下构造了一棵树

如对于文法G_{exp}，关于 v *(v+d) 的一个推导过程可以认为是构造了如下一棵树：



(1) E → EOE
(2) E → (E)
(3) E → v
(4) E → d
(5) O → +
(6) O → *

$E \xrightarrow{(1)} EOE \xrightarrow{(6)} E*E \xrightarrow{(2)} E*(E) \xrightarrow{(3)} v*(E)$

$\xrightarrow{(1)} v*(EOE) \xrightarrow{(5)} v*(E+E) \xrightarrow{(3)} v*(v+E) \xrightarrow{(4)} v*(v+d)$

语法分析树

✧ 语法分析树（*parse trees*）

对于 ***CFG G = (V, T, P , S)***，语法分析树是满足下列条件的树：

(1) 每个内部结点由一个非终结符标记.

(2) 每个叶结点或由一个非终结符，或由一个终结符，或由 ε 来标记. 但标记为 ε 时，它必是其父结点唯一的孩子.

(3) 如果一个内部结点标记为 ***A***，而其孩子从左至右分别标记为$X_1, X_2, \ldots, X_k$，则 $A \to X_1X_2\ldots X_k$ 是 ***P*** 中的一个产生式. 注意：只有 $k=1$ 时上述 X_i 才有可能为ε，此时结点 ***A*** 只有唯一的孩子，且 $A \to \varepsilon$ 是***P*** 中的一个产生式.

语法分析树

✧ 语法分析树的果实（ *yield* ）

设 ***CFG G = (V, T, P , S)***. 将语法分析树的每个叶结点按照从左至右的次序连接起来，得到一个 $(V \cup T)^*$ 中的字符串，称为该语法树的果实.

G 的每个句型都是某个根结点为 ***S*** 的分析树的果实; 这些分析树中，有些树的果实为句子，它们构成了 ***G*** 的语言.

归约、推导与分析树之间关系

✧ 三者之间的关系

设 *CFG* $G = (V, T, P, S)$. 以下命题是相互等价的：

(1) 字符串 $w \in T^*$ 可以归约（递归推理）到非终结符 A；

(2) $A \overset{*}{\Rightarrow} w$；

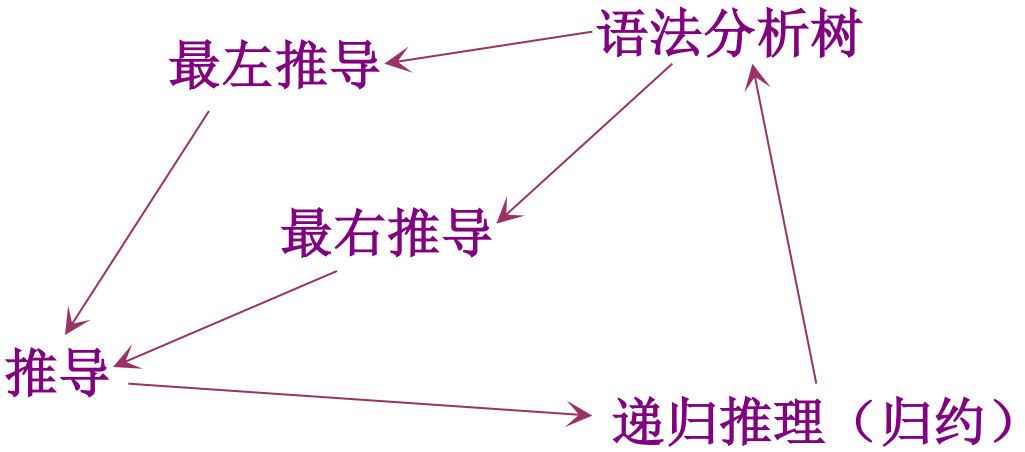
(3) $A \overset{*}{\underset{lm}{\Rightarrow}} w$；

(4) $A \overset{*}{\underset{rm}{\Rightarrow}} w$；

(5) 存在一棵根结点为 A 的分析树，其果实为 w.

归约、推导与分析树之间关系

✧ 证明策略



归约、推导与分析树之间关系

✧ 从归约到分析树

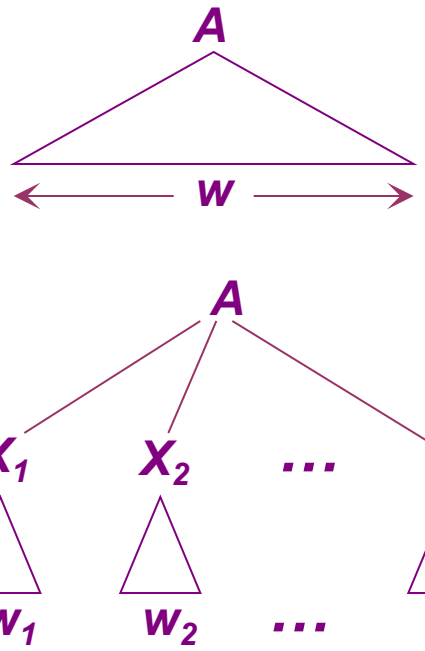
设 *CFG* $G = (V, T, P, S)$. 如果字符串 $w \in T^*$ 可以归约到非终结符A，则存在一棵根结点为 A 的分析树，其果实为 w.

✧ 证明思路

归纳于从 w 归约到 A 的步数.

基础 步数为 *1*. 一定有产生式 $A \to w$. 存在右上图所示的分析树.

归纳 设步数大于 *1*，且最后一步归约使用了产生式 $A \to X_1X_2\ldots X_k$. 存在右下图所示的分析树.



归约、推导与分析树之间关系

✧ 从分析树到推导

设 *CFG* $G = (V, T, P, S)$. 如果存在一棵根结点为 A 的分析树，其果实为字符串 $w \in T^*$，则 $A \Rightarrow^* w$， $A \underset{lm}{\Rightarrow}^* w$，$A \underset{rm}{\Rightarrow}^* w$.

✧ 证明思路

只证明 $A \underset{lm}{\Rightarrow}^* w$，$A \underset{rm}{\Rightarrow}^* w$ 可类似证明；

同时也证明了 $A \Rightarrow^* w$.

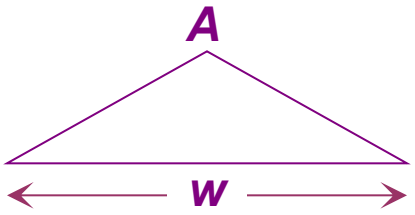
归纳于分析树的高度来证明 $A \underset{lm}{\Rightarrow}^* w$.

归约、推导与分析树之间关系

✧ 从分析树到最左推导

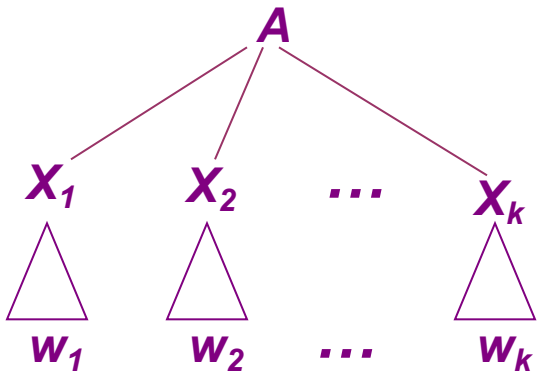
– 证明思路　归纳于分析树的高度来证明 $A \overset{*}{\underset{lm}{\Rightarrow}} w$.

基础　高度为 *1*. 分析树一定如右图所示，必定有产生式 $A \to w$. 因此，$A \overset{*}{\underset{lm}{\Rightarrow}} w$.



归纳 高度大于 *1* 的分析树一定如右下图所示，必定有产生式 $A \to X_1X_2\ldots X_k$.　存在$w_1, w_2, \ldots, w_k$，w_i 是X_i子树的果实或 $w_i = X_i$（$1 \le i \le k$），且 $w = w_1w_2\ldots w_k$，由归纳假设，$X_i \overset{*}{\underset{lm}{\Rightarrow}} w_i$（$1 \le i \le k$）.

在此基础上易证得　$A \overset{*}{\underset{lm}{\Rightarrow}} w$.



归约、推导与分析树之间关系

✧ 从推导到归约

设 *CFG* $G = (V, T, P, S)$. 如果对于非终结符 A 和字符串 $w \in T^*$， $A \overset{*}{\Rightarrow} w$，则 w 可以归约到 A.

✧ 证明思路 归纳于推导 $A \overset{*}{\Rightarrow} w$ 的步数.

基础 步数为 *1*. 一定有产生式 $A \to w$. w 可以归约到 A.

归纳 设步数大于 *1*，第一步使用了产生式$A \to X_1X_2\ldots X_k$.
该推导形如 $A \Rightarrow X_1X_2\ldots X_k \overset{*}{\Rightarrow} w$. 可以将 w 分成 $w = w_1w_2\ldots w_k$，其中

(a) 若 X_i 为终结符，则 $w_i = X_i$.

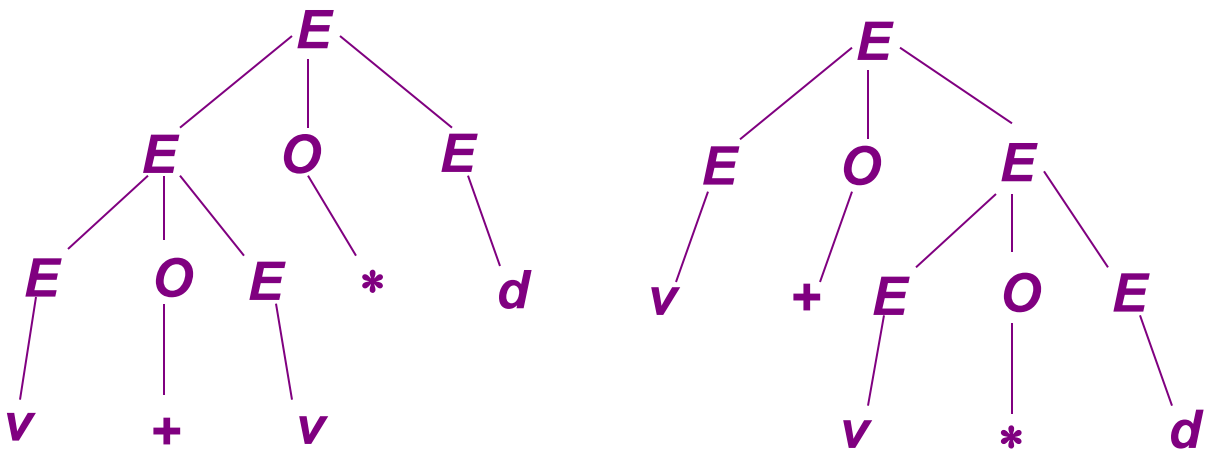
(b) 若 X_i 为非终结符，则 $X_i \overset{*}{\Rightarrow} w_i$. 由归纳假设， w_i 可以归约到 X_i .

这样，w_i 或者为X_i，或者可以归约到 X_i，使用产生式$A \to X_1X_2\ldots X_k$，得出w 可以归约到 A.

文法和语言中的二义性

✧ 文法的二义性

– 二义文法（*ambiguous grammars*）举例　考虑右下文法，对于终结符串 *v* + *v* * *d*，存在两棵不同的分析树，它们的根结点都为开始符号 *E*，果实都为*v* + *v* * *d* .



(1) $E \to EOE$
(2) $E \to (E)$
(3) $E \to v$
(4) $E \to d$
(5) $O \to +$
(6) $O \to *$

文法和语言中的二义性

✧ 文法的二义性

– 二义文法概念　*CFG* $G = (V, T, P, S)$ 为二义的，如果对某个 $w \in T^*$，存在两棵不同的分析树，它们的根结点都为开始符号 S，果实都为 w. 如果对每一 $w \in T^*$，至多存在一棵这样的分析树，则 G 为无二义的.

– 二义性的判定　一个 *CFG* 是否为二义的问题是不可判定的，即不存在解决该问题的算法. *(theorem9.20)*

– 消除二义性　将会看到，没有通用的办法可以消除文法的二义性. 在实践中，对于特定的文法，通常可以找到消除二义性的办法.

文法和语言中的二义性

✧ 文法二义性的另一种定义

– 定义 *CFG* $G = (V, T, P, S)$ 为二义的，如果存在某个 $w \in T^*$， 存在两个不同的从开始符号 S 到 w 的最左推导.

该定义源于如下结论：

– 结论 对 *CFG* $G = (V, T, P, S)$ 和 $w \in T^*$，w 具有两棵不同的分析树，当且仅当存在两个不同的从开始符号 S 到 w 的最左推导.

证明思路 从不同的分析树可构造不同的最左推导；反之，从不同的最左推导可构造不同的分析树.

– 有时方便证明文法的无二义性 例如，练习5.4.7.

文法和语言中的二义性

✧ 语言中的二义性

- 如果上下文无关语言 *L* 的所有文法都是二义的，则称 *L* 是固有二义的（***inherently ambiguous***）

- 举例　上下文无关语言
  $L = \{a^n b^n c^m d^m \mid n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$
  是固有二义的. 以下是 *L* 的一个 *CFG*

$$S \to AB \mid C$$
$$A \to aAb \mid ab$$
$$B \to cBd \mid cd$$
$$C \to aCd \mid aDd$$
$$D \to bDc \mid bc$$

- 推论　没有通用的办法可以消除文法的二义性.

文法和语言中的二义性

✧ 无二义文法的设计

请给出下列语言 *L* 的一个无二义文法：

$$L = \{ a^n b^m \mid m \geq n \geq 0 \}$$

$$S \to aSb \mid B$$
$$B \to bB \mid \epsilon$$

$$S \to AB$$
$$A \to aAb \mid \epsilon$$
$$B \to bB \mid \epsilon$$

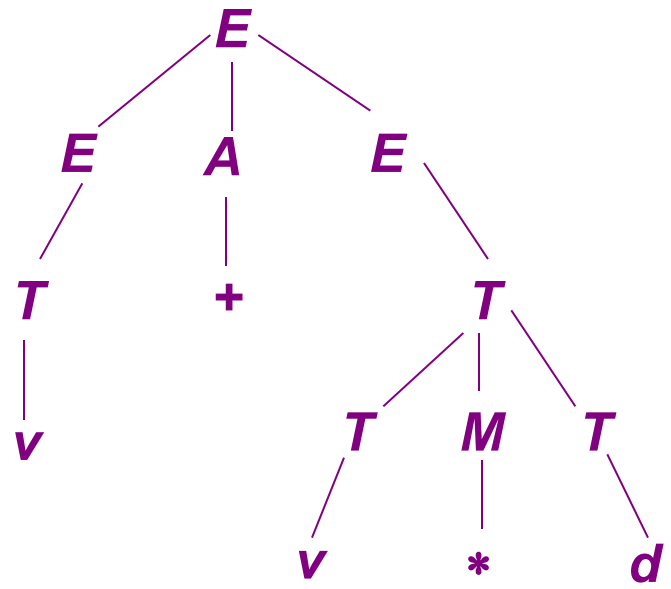
文法和语言中的二义性

✧ 消除二义性的几种文法变换方法

– 对于右上图的文法，采用算符优先级联方法将其变换为左下图的文法，对于该文法，串 $v+v*d$ 存在唯一的分析树.

$$E \to EOE \mid (E) \mid v \mid d$$
$$O \to + \mid *$$

$$E \to EAE \mid T$$
$$T \to TMT \mid (E) \mid v \mid d$$
$$A \to +$$
$$M \to *$$

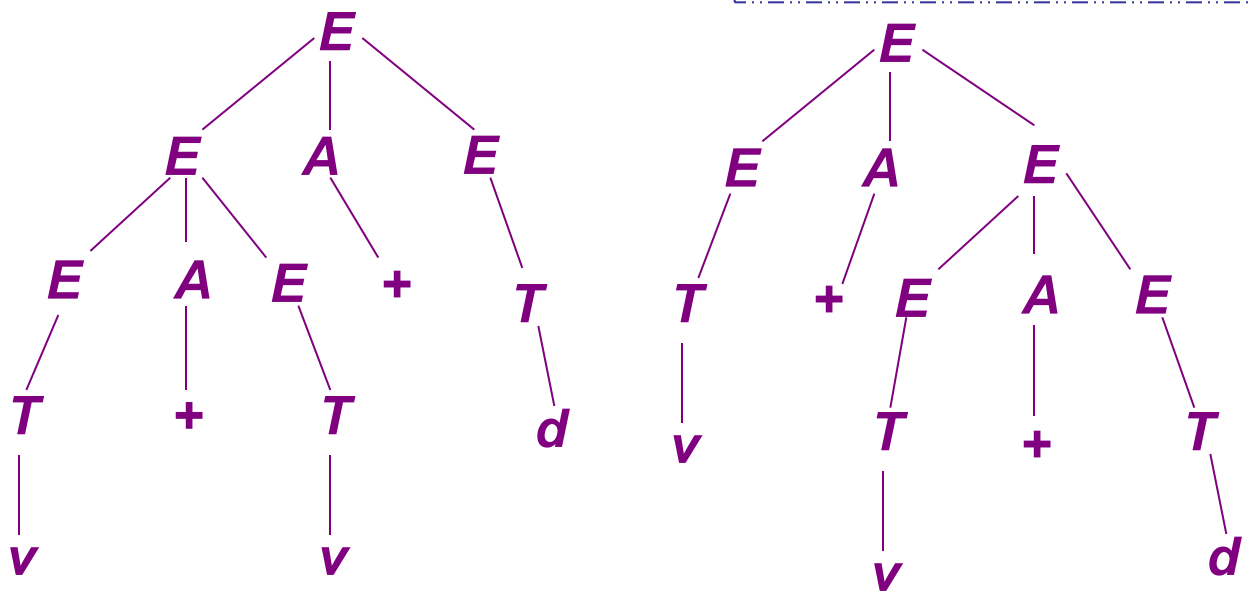


文法和语言中的二义性

✧ 消除二义性的几种文法变换方法

– 右上图的文法仍然是二义文法，串 $v+v+d$ 存在不同的分析树（下图）.

$$E \to EAE \mid T$$
$$T \to TMT \mid (E) \mid v \mid d$$
$$A \to +$$
$$M \to *$$

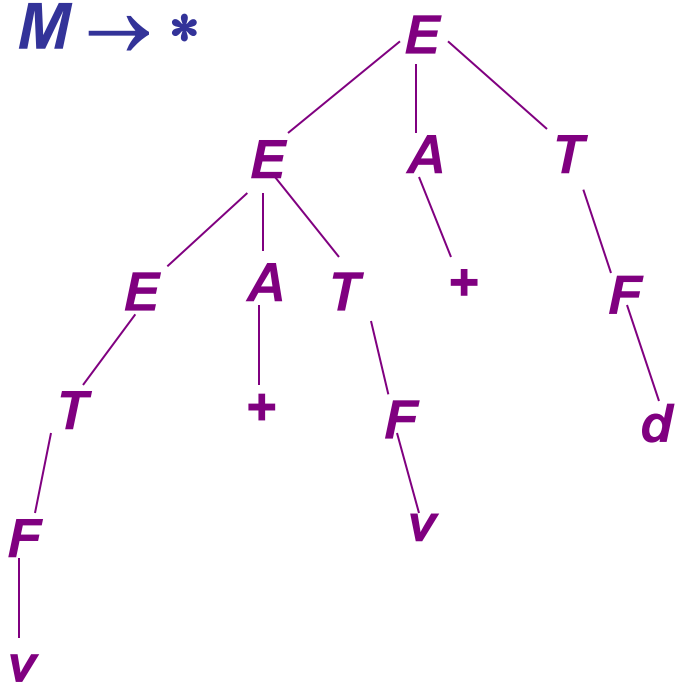


文法和语言中的二义性

✧ 消除二义性的几种文法变换方法

– 采用左结合方法将右上图的文法变换为左下图，串 *v* + *v* + *d* 存在唯一的分析树（右下图）.

$E \to E\,A\,E \mid T$
$T \to T\,M\,T \mid (\,E\,) \mid v \mid d$
$A \to +$
$M \to *$



$E \to E\,A\,T \mid T$
$T \to T\,M\,F \mid F$
$F \to (\,E\,) \mid v \mid d$
$A \to +$
$M \to *$

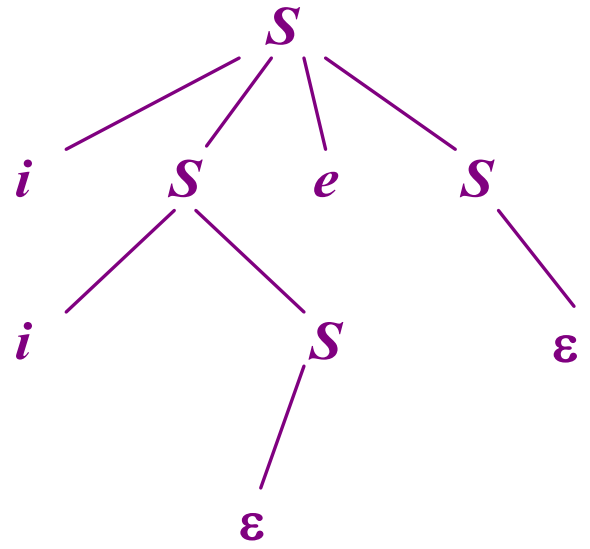
文法和语言中的二义性

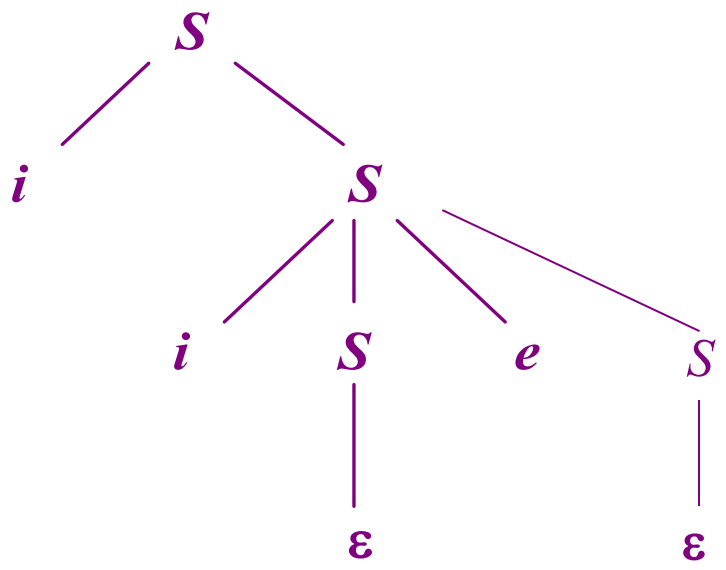
✧ 消除二义性的几种文法变换方法

– 悬挂else二义性

$S \rightarrow \varepsilon \mid i\,S \mid i\,S\,e\,S$

$i\,i\,e$





文法和语言中的二义性

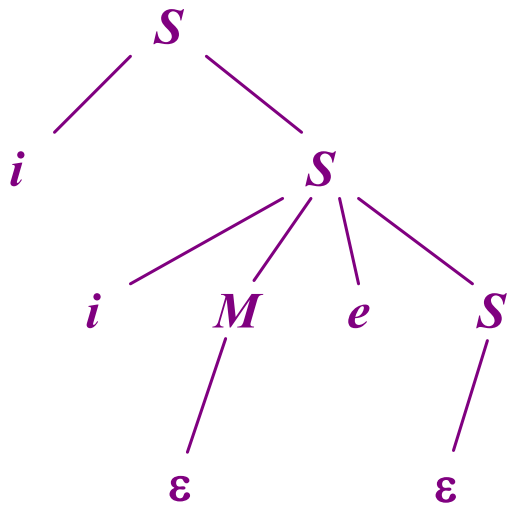
✧ 消除二义性的几种文法变换方法

– 采用最近嵌套匹配方法
消除悬挂 **else** 二义性

$S \to \varepsilon \mid iS \mid iSeS$

将右上部的文法变换为
下面的文法

$S \to \varepsilon \mid iS \mid iMeS$
$M \to \varepsilon \mid iMeM$



串 *iie* 存在唯一的
分析树（右图）

课后练习

✧ 必做题:

*! *Ex.5.1.1(b)*, *Ex.5.1.2(c)*（最左推导和最右推导各一个）
Ex.5.1.6 (b), !! *Ex.5.1.8*, !*Ex.5.2.2*, *Ex.5.4.7 (a)*

附加1 构造如下语言的上下文无关文法：

（1） $\{a^n b^{2n} c^m \mid n, m \geq 0\}$

（2） $\{a^n b^{n+m} c^m \mid n, m \geq 0\}$

（3） $\{a^m b^n c^p d^q \mid n,m,p,q \geq 0$ 及 $m+n = p+q\}$

（4） $\{a^n b^i c^j d^m \mid n, m, i, j \geq 0 \wedge n + m = i + j\}$

（5） $\{uawb \mid u, w \in \{a, b\}^* \wedge |u| = |w|\}$

（6） $\{a^i b^j c^k \mid i, j, k \geq 0$, 若 $j=1$ 则 $i=k\}$

附加2 给出语言 $\{a^m b^n \mid m \geq 2n \geq 0\}$ 的二义文法和非二义文法各一个

附加3 适当变换文法，找到下列文法所定义语言的一个无二义的文法：$S \to SaS \mid SbS \mid ScS \mid d$

附加4 设 G[S]为上下文无关文法，其终结符集为{0, 1}，产生式集合如下：

$S \to \varepsilon \mid 0T1$

$T \to \varepsilon \mid 1S0$

证明：若 $S \Rightarrow^* w$，则有 $w \in L = \{(01)^k \mid k \geq 0\}$。

课后练习

✧ 思考题:

- !Ex.5.1.1 (c)
- !Ex.5.1.7 (a)
- Ex.5.4.7 ! (b)
- 附加：

（1）设G为上下文无关文法，其终结符集合为 {a，b，c}，开始符号为S，产生式集合如下：

$$S \to A \mid aSc$$
$$A \to B \mid bAc$$
$$B \to \varepsilon \mid Bc$$

试证明 $L(G) = \{ a^i b^j c^k \mid i + j \le k$，其中 i、j、k 均为自然数}。

（2）完成第21页的证明。

（3）构造产生如下语言的上下文无关文法:

{ w1cw2c … cwkccwjR ｜ k≥1 ∧ 1≤j≤ k 以及对任何1≤ i ≤ k ，有 wi∈{a, b}+ }

课后练习

✧ 自测题:

– 试给出下列语言的一个上下文无关文法：

（1） $\{a^n b^n c^m d^m \mid n\geq 1, m\geq 1\} \cup \{a^n b^m c^m d^n \mid n\geq 1, m\geq 1\}$

（2） $\{a^n b^m \mid n, m \geq 0 \wedge n\neq m\}$

（3） $\{a^n b^m \mid n\geq 0, m\geq 0,$ 以及 $3n \geq m \geq 2n\}$

（4） $\{w \mid w \in \{a, b\}^*,$ w 中 a 和 b 的数目不同$\}$

（5） $\{w \mid w\in\{a, b\}^*,$ 且 w 中 a 与 b 的数目相差为 $2\}$

– 考虑由下列产生式定义的上下文无关文法 G：

$$S \to aS \mid aSb \mid \varepsilon$$

试证明 $L(G) = \{a^n b^m \mid n \geq m \geq 0\}$

– 考虑由下列产生式定义的上下文无关文法 G：

$$S \to 1 \mid 1T1$$
$$T \to 0 \mid 0S0$$

试证明 若$S \Rightarrow^* w$，则有occur(1, w) = occur(0, w) + 1。
其中，occur(a, w) 表示终结符a 在串w 中出现的次数。

That's all for today.

Thank You