第 二 讲

✧ 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 $\Sigma = \{ 0, 1 \}$， $L = \{ 0^n1^n \mid n \geq 1\}$，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:

$S \rightarrow 01$

$S \rightarrow 0S1$

✧ 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>

产生式集合

$E \rightarrow EOE$

$E \rightarrow (E)$

$E \rightarrow v$

$E \rightarrow d$

$O \rightarrow +$

$O \rightarrow *$

开始符号

非终结符

终结符

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 $A \in V$, $\alpha \in (V \cup T)^*$

✧ 上下文无关文法举例

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

$$S \rightarrow 01$$
$$S \rightarrow 0S1$$

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

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

✧ 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

$$\boxed{\begin{array}{l} S \to 01 \\ S \to 0S1 \end{array}} \Longrightarrow \boxed{S \to 01 \mid 0S1}$$

$$\boxed{\begin{array}{l} E \to EOE \\ E \to (E) \\ E \to v \\ E \to d \\ O \to + \\ O \to * \end{array}} \Longrightarrow \boxed{\begin{array}{l} E \to EOE \mid (E) \mid v \mid d \\ O \to + \mid * \end{array}}$$

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

归约与推导

✧ 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

$$v*(v+d) \xrightarrow{(4)} v*(v+E) \xrightarrow{(6)} vO(v+E) \xrightarrow{(3)} vO(E+E)$$

$$\xrightarrow{(5)} vO(EOE) \xrightarrow{(1)} vO(E) \xrightarrow{(2)} vOE \xrightarrow{(3)} EOE \xrightarrow{(1)} E$$

✧ 推导过程举例

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）$O \rightarrow *$

从开始符号到字符串 $v*(v+d)$ 的一个推导过程为

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

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

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 $\underset{G}{\Rightarrow}$ 描述. 设 $\alpha,\beta \in (V \cup T)^*$，$A \rightarrow \gamma$ 是一个产生式，则定义

$$\alpha A \beta \underset{G}{\Rightarrow} \alpha\gamma\beta .$$

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 $\overset{*}{\underset{G}{\Rightarrow}}$，可归纳定义如下：

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

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

$$\alpha \overset{*}{\underset{G}{\Rightarrow}} \gamma .$$

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:

$$E \overset{*}{\underset{lm}{\Rightarrow}} EOE \overset{*}{\underset{lm}{\Rightarrow}} vOE \overset{*}{\underset{lm}{\Rightarrow}} v*E$$

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

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

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

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：

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

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

归约与推导

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

若 $S \underset{rm}{\overset{*}{\Rightarrow}} \alpha$，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*）.

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.

上下文无关语言

✧ 文法设计

例 给出语言 $L = \{ 0^n1^n \mid n \geq 1 \}$ 的一个文法。

如下是一个可接受该语言的上下文无关文法 G[S]:

$$S \rightarrow 01$$
$$S \rightarrow 0S1$$

课堂练习？

✧ 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.

– 一个例子

见定理5.7.

上下文无关语言

✧ 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:

$$S \rightarrow aSbS \mid bSaS \mid \varepsilon$$

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$w \in L(G)$.
  （通过对|w|进行归纳来证明w在 L(G) 中，即$S \Rightarrow^* w$）
- 若 $w \in L(G)$，即$S \Rightarrow^* w$，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 ）

（留作练习）

上下文无关语言

✧ 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 $\alpha \rightarrow \beta$, 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 $|\alpha| \leq |\beta|$，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 $A \rightarrow \beta$, 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.

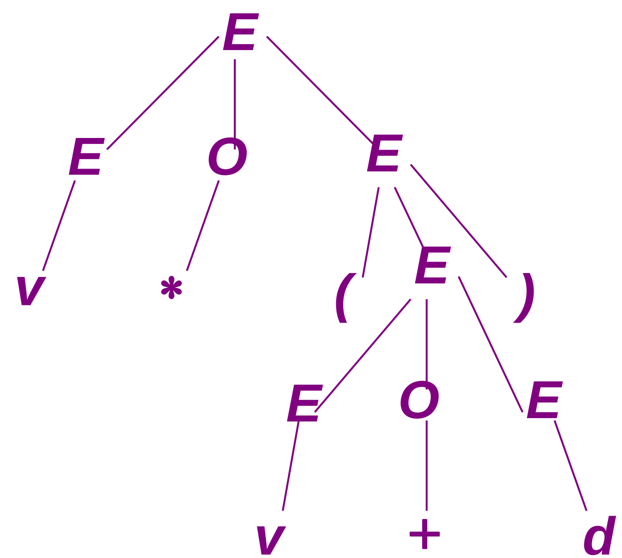
能够用 3 型文法定义的语言称为3 型语言，即正规语言.

✧ 结论

3 型文法的能力等价于有限状态自动机.

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

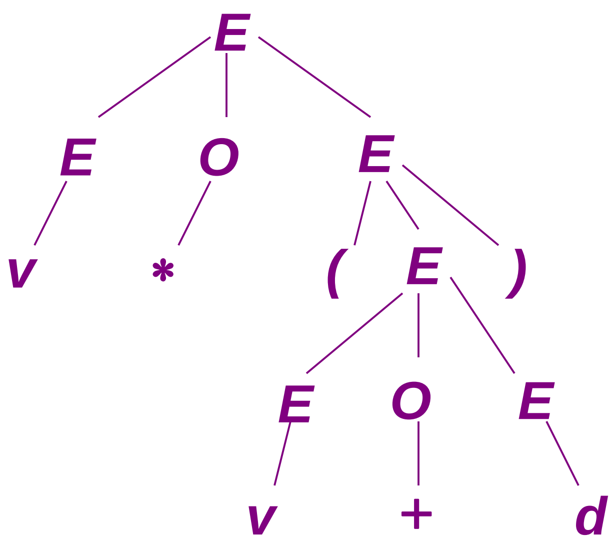
$$v*(v+d) \xrightarrow{(4)} v*(v+E) \xrightarrow{(6)} vO(v+E) \xrightarrow{(3)} vO(E+E)$$

$$\xrightarrow{(5)} vO(EOE) \xrightarrow{(1)} vO(E) \xrightarrow{(2)} vOE \xrightarrow{(3)} EOE \xrightarrow{(1)} E$$

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

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

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

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 $A \rightarrow X_1X_2\ldots X_k$ 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 $A \rightarrow \varepsilon$ 是P 中的一个产生式.

✧ 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) $A \overset{*}{\Rightarrow} w$;

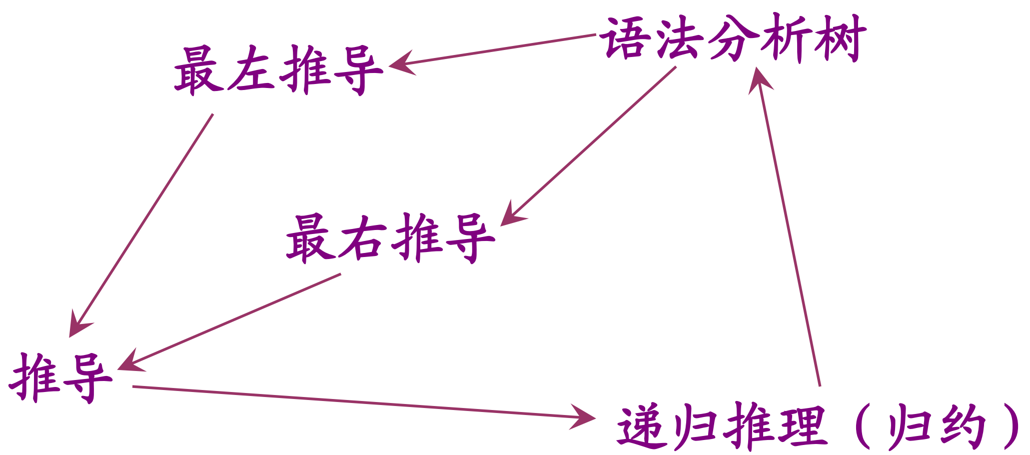
(3) $A \underset{lm}{\overset{*}{\Rightarrow}} w$;

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

(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



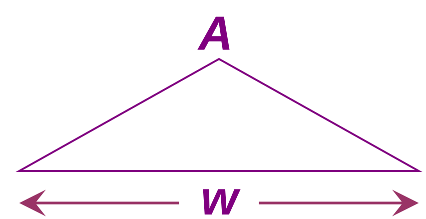
归约、推导与分析树之间关系

✧ 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.

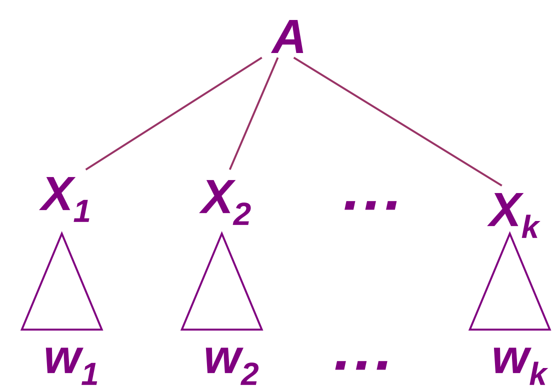


基础 步数为 1. 一定有产生式 $A \rightarrow w$. 存在右上图所示的分析树.

归纳 设步数大于 1，且最后一步归约使用了产生式

$$A \rightarrow X_1X_2\ldots X_k .$$

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 $A \overset{*}{\Rightarrow} w$， $A \underset{lm}{\overset{*}{\Rightarrow}} w$，$A \underset{rm}{\overset{*}{\Rightarrow}} w$.

✧ 证明思路

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

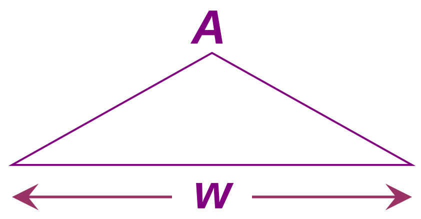
同时也证明了 $A \overset{*}{\Rightarrow} w$.

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

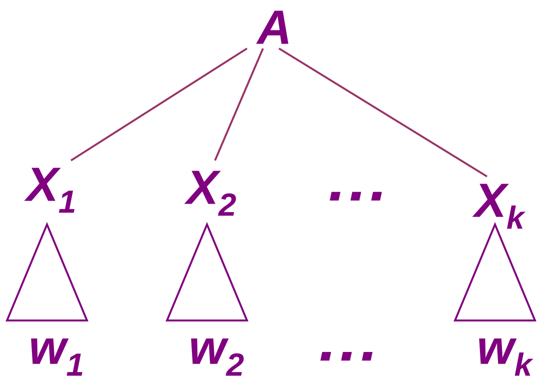
✧ 从分析树到最左推导

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

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



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 $A \rightarrow X_1X_2\ldots X_k$. 存在 w_1, w_2, …, 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，$X_i \underset{lm}{\overset{*}{\Rightarrow}} w_i$（$1 \le i \le k$）. 在此基础上易证得 $A \underset{lm}{\overset{*}{\Rightarrow}} w$.



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 $A \rightarrow w$. w 可以归约到 A.

归纳 设步数大于 1，第一步使用了产生式$A \rightarrow X_1X_2 \ldots X_k$. 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 $w_i = X_i$.

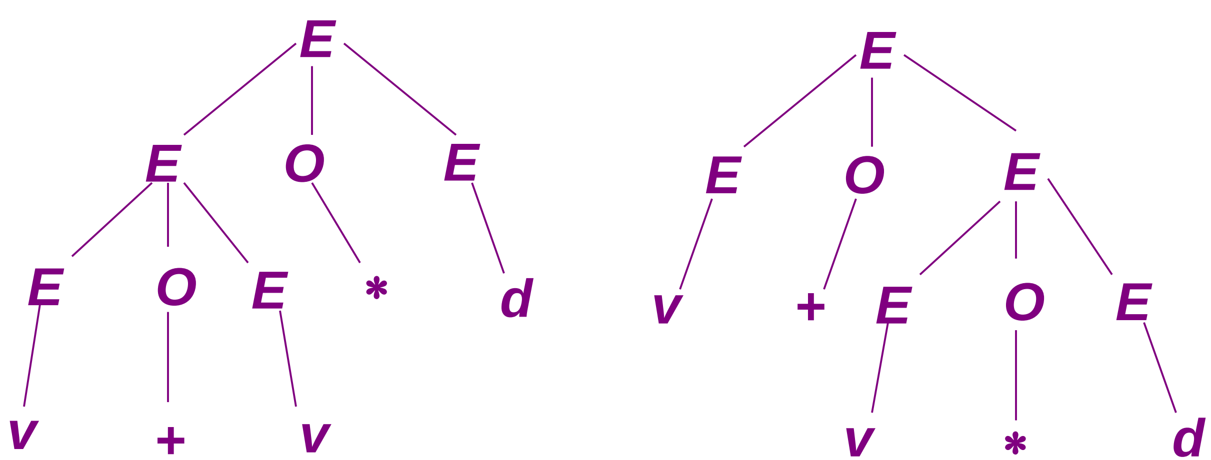
(b) 若 X_i 为非终结符，则 $X_i \overset{*}{\Rightarrow} w_i$. 由归纳假设， w_i 可以归约到 X_i .

这样，w_i 或者为X_i，或者可以归约到 X_i，使用产生式$A \rightarrow X_1X_2 \ldots X_k$，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) $E \rightarrow EOE$
(2) $E \rightarrow (E)$
(3) $E \rightarrow v$
(4) $E \rightarrow d$
(5) $O \rightarrow +$
(6) $O \rightarrow *$

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*

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

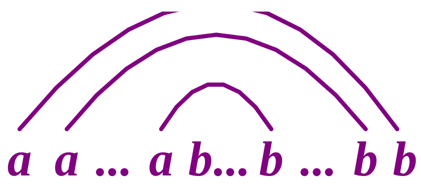
– 推论　没有通用的办法可以消除文法的二义性.

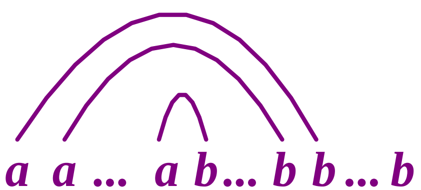
文法和语言中的二义性

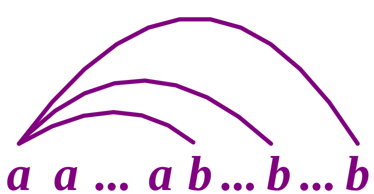
✧ 无二义文法的设计

– 课堂思考问题 给出下列语言 L 的一个无二义文法：

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







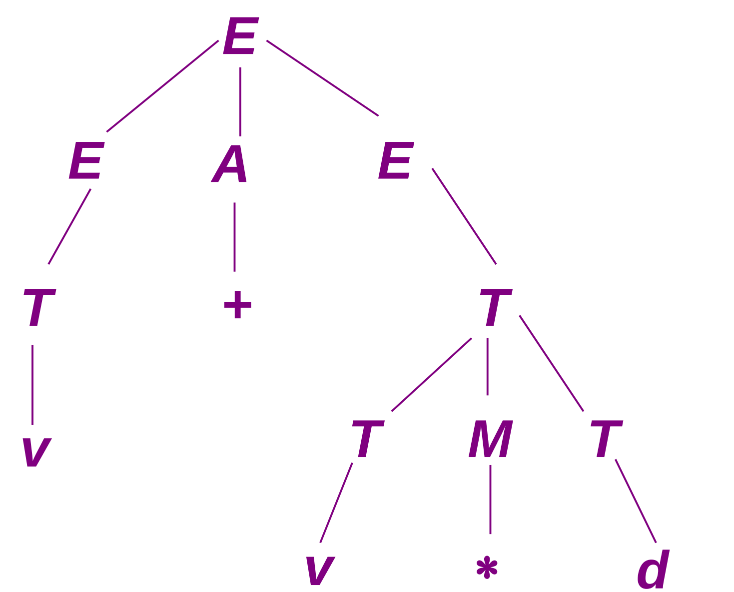
文法和语言中的二义性

✧ 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.

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

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

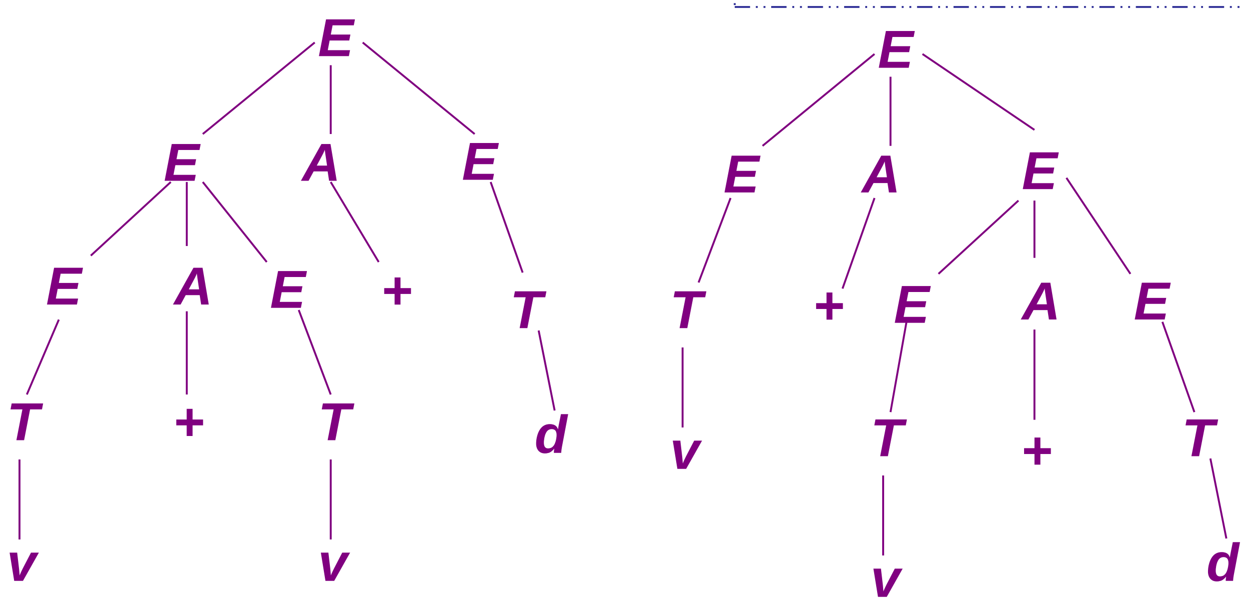


文法和语言中的二义性

✧ 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）.

$$
\begin{aligned}
E &\to E\,A\,E \mid T \\
T &\to T\,M\,T \mid (E) \mid v \mid d \\
A &\to + \\
M &\to *
\end{aligned}
$$



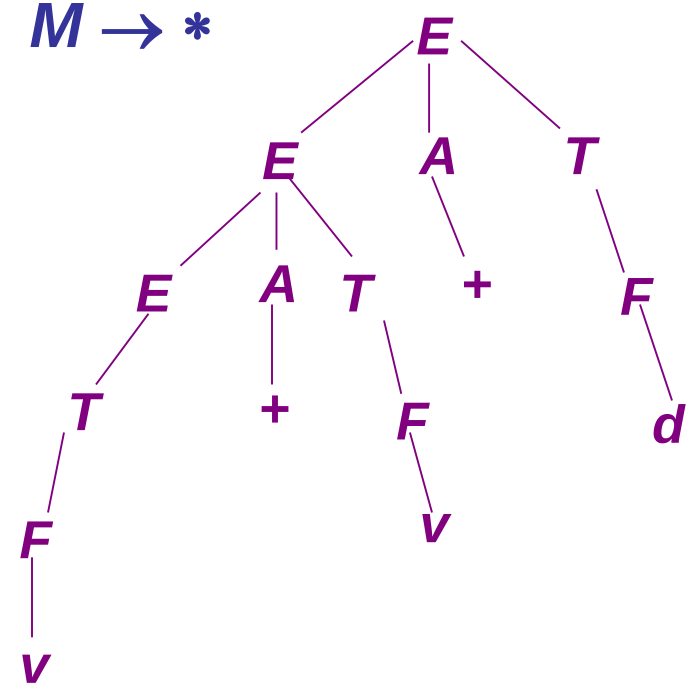
文法和语言中的二义性

✧ 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）.

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

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



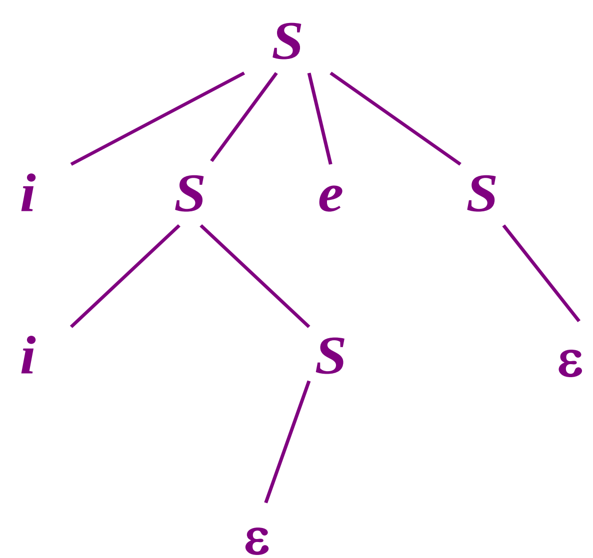
文法和语言中的二义性

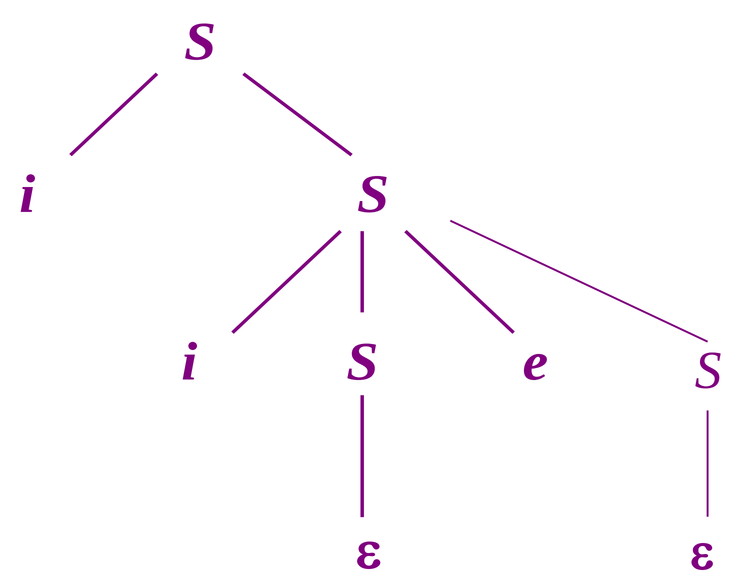
✧ 消除二义性的几种文法变换方法

– 悬挂else二义性

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

iie





文法和语言中的二义性

✧ 消除二义性的几种文法变换方法

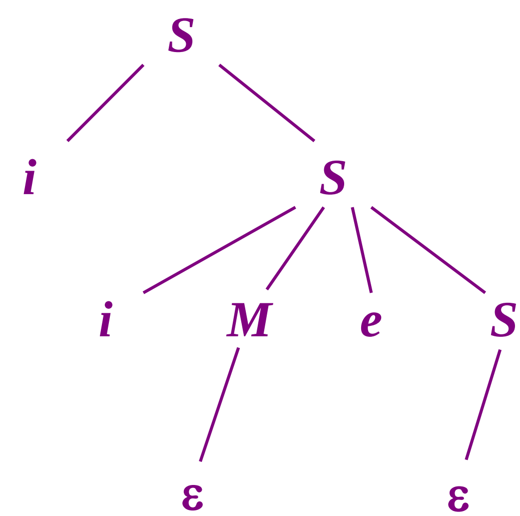
– 采用最近嵌套匹配方法消除悬挂 else 二义性

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

将右上部的文法变换为下面的文法

$$S \to \varepsilon \mid i\,S \mid i\,M\,e\,S$$
$$M \to \varepsilon \mid i\,M\,e\,M$$

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

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: $S \rightarrow SaS \mid SbS \mid ScS \mid d$

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。

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$\}$

  - 考虑由下列产生式定义的上下文无关文法 G:

    $$S \rightarrow 0S1 \mid \varepsilon.$$

    证明$L(G)=\{0^n1^n \mid n\geq 0\}$

That's all for today.

Thank You