第 2 章　形式语言基础（2）

【内容提要】

2.1　形式语言是符号串集合
2.2　形式语言是由文法定义的
2.3　主要语法成分的定义
2.4　两类特性文法
2.5　文法变换方法
2.6　关于形式语言的分类问题

※ 上节课主要内容回顾：

❖ 形式语言是符号串集合且由文法定义！

1. 文法 是规则集合，四元组：$G(Z)=(V_N,\ V_T,\ Z,\ P)$
2. 文法所定义的语言：

$$L(G)=\{\ x\ |\ Z \overset{+}{\Rightarrow} x, x\in V_T^*\ \}$$

3. 文法应用示例：

(1) 简单语言的文法构造：

① 无符号整数文法：G(N):

N -> dN | d

d(数字), ℓ(字母)

② 标识符文法：G(I)：

I->ℓ A|ℓ
A->ℓ A|dA|ε

(2) 求解文法所定义的语言(或句子)方法：

① 正规方程组迭代求语言（如 标识符文法）

② 直接由定义求解句子(如 算术表达式文法)。

2.3 主要语法成分的定义

2.3.1 文法的运算问题

- 文法有两种基本运算：推导，归约。

设 x，y∈$(V_N+V_T)^*$，A->α∈P

Ⅰ．直接推导（ => ）：xAy => xαy

即：指用产生式的右部符号串替换左部非终结符。

直接推导算符

❖ $\overset{+}{\Rightarrow}$ $\overset{+}{\Rightarrow}$

（当且仅当$\alpha\Rightarrow\alpha_1\Rightarrow\alpha_2\Rightarrow\ldots\Rightarrow\beta$）

即：指一步或一步以上的直接推导运算。

加推导算符

❖ 星推导（ $\overset{*}{\Rightarrow}$ ）：α $\overset{*}{\Rightarrow}$ β

（当且仅当α=β 或$\alpha\Rightarrow\alpha_1\Rightarrow\alpha_2\Rightarrow\ldots\beta$）

即：指零步或零步以上的直接推导运算。

星推导算符

$\Rightarrow$ $\Rightarrow$

即：直接归约是直接推导的**逆运算**，用产生式的左部非终结符**替换**右部符号串。

直接归约算符

$\overset{+}{\Rightarrow}$ $\overset{+}{\Rightarrow}$

（当且仅当 $\alpha \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \cdots \Rightarrow \beta$）

即：指一步或一步以上的直接归约运算。

加归约算符

❖ 星归约（$\overset{*}{\Rightarrow}$）：$\alpha \overset{*}{\Rightarrow} \beta$

（当且仅当 $\alpha=\beta$ 或 $\alpha \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \cdots \Rightarrow \beta$）

即：指零步或零步以上的直接归约运算。

星归约算符

※ 实用中最常见的两种运算：

这是相应的算符

- 最左推导（$\overset{+}{\Rightarrow}_{lm}$）—每次推导皆最左非终结符优先；
- 最左归约（$\overset{+}{\Rightarrow}_{lm}$）—每次归约皆最左可归约串优先。

➢文法运算示例：

【例2.8】

算数表达式文法：

G(E)：E->T|E+T|E-T
T->F|T*F|T/F
F->i |(E)

- 给定一个符号串 i+i*i ：

最左非终结符

1. 最左推导(从开始符号出发)过程：

E => E+T => T+T => F+T => i+T => i+T*F
=> i+F*F => i+i*F => i+i*i

∴ E $\overset{+}{\Rightarrow}$ i+i*i

最左可归约串

2. 最左归约(从符号串出发)过程：

i+i*i => F+i*i => T+i*i => E+i*i => E+F*i
=> E+T*i => E+T*F => E+T => E

∴ i*i+i $\overset{+}{\Rightarrow}$ E

2.3 主要语法成分的定义(续1)

2.3.2 句型、句子和语法树

设有文法 $G(Z)=(V_N, V_T, Z, P)$，则：

1. 句型

若有 $Z \overset{+}{\Rightarrow} \alpha$ ，则 α 是句型；

即：句型是由文法开始符号加推导出的任一符号串。

2. 句子

若有 $Z \overset{+}{\Rightarrow} \alpha$ 且 $\alpha \in V_T^*$ ，则 α是句子；

即：句子是由开始符号加推导出的任一终结符号串。

3. 语法树

句型(句子)产生过程的一种树结构表示；

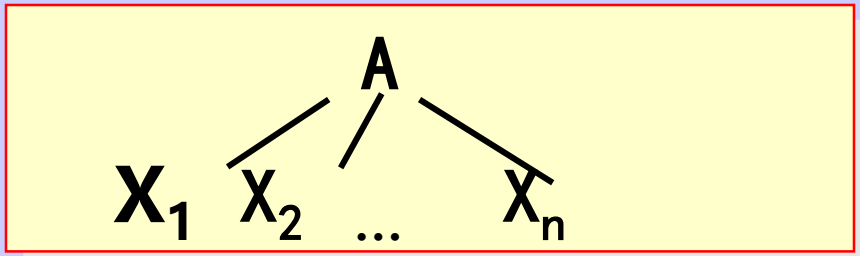
树根--开始符号；树叶—给定的句型(句子)。

2.3 主要语法成分的定义(续2)

【语法树的构造算法】：

① 置树根为开始符号；

② 若采用了推导产生式： $A \rightarrow x_1x_2...x_n$，则有子树：



③ 重复步骤②，直到再没有推导产生式为止。

※ 如此构造的语法树，其全体树叶（自左至右）恰好是给定的句型。

※ 句型、句子和语法树示例：

【例2.10】
算术表达式
文法：

E -> T | E + T | E - T
T -> F | T * F | T / F
F -> i | (E)

(1) 证明 (T/F+F)*i 是一个句型(表达式型)；
(2) 画出该句型的语法树。

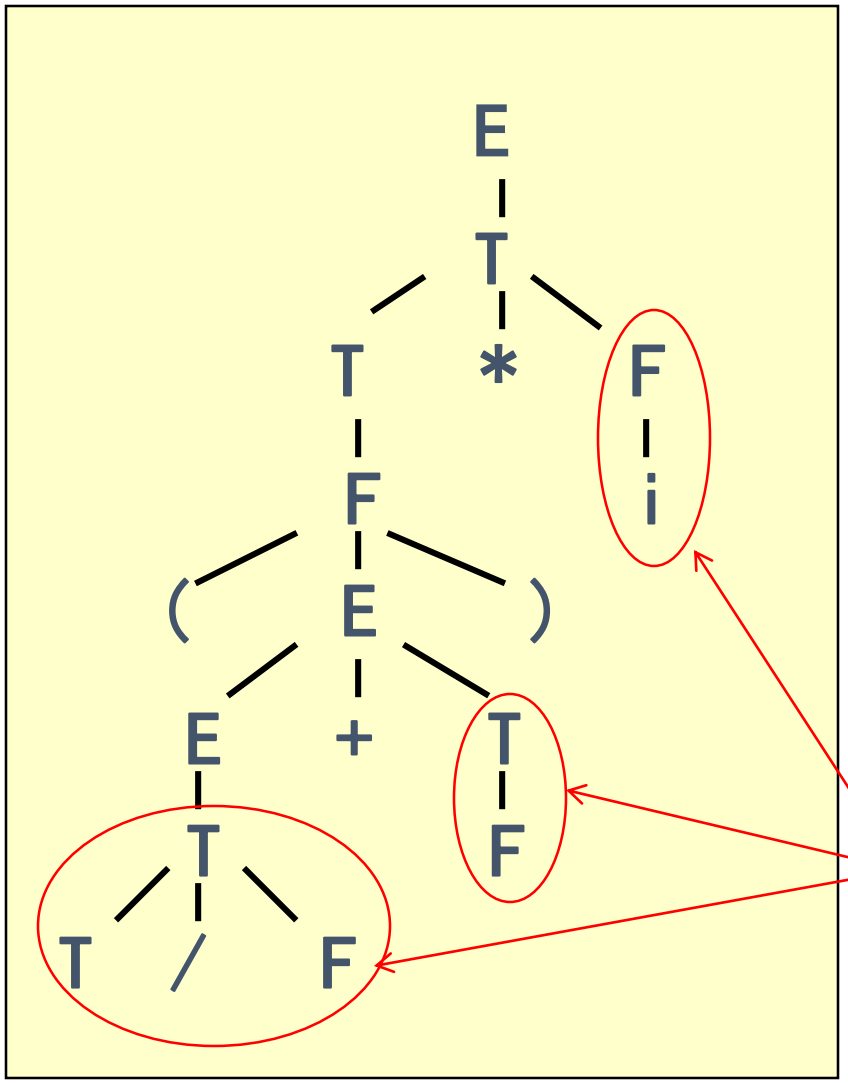
【证】

E =>T =>T*F =>F*F =>(E)*F =>(E+T)*F
=>(T+T)*F =>(T/F+T)*F =>(T/F+F)*F
=>(T/F+F)*i

即：$E \overset{+}{\Rightarrow} (T/F+F)*i$

证闭

※ 句型 (T/F+F)*i 的语法树构造：



E ->T |E+T |E-T
T ->F |T* F |T/F
F -> i | (E)

【注】关于语法树：

- **子树**：以任何具有分支的结点为根所形成的树，称为原树的子树。
- **简单子树**：仅具有单层分支的子树。

2.3.3 短语、简单短语和句柄

【例2.11】图2.2为一个中文句型的语法树：

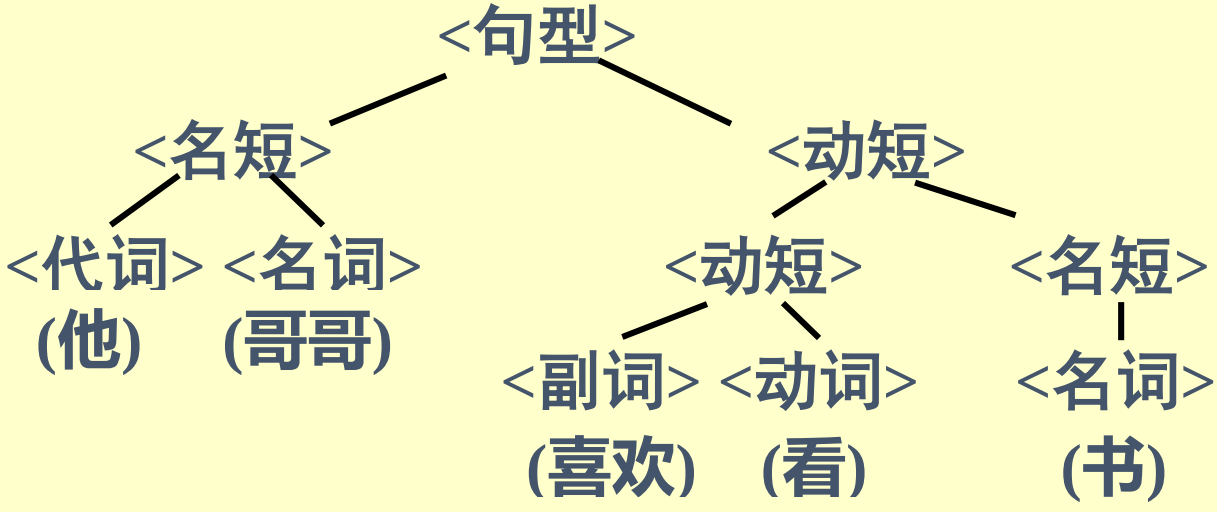
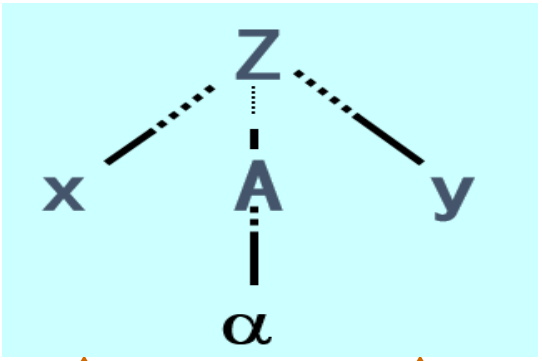


图2.2 他哥哥喜欢看书的语法树

- 短 语 -- 他哥哥<名短>，喜欢看<动短>，书<名词>，
  喜欢看书<动短>，他哥哥喜欢看书<句子>
- 简单短语 -- 他哥哥，喜欢看，书
- 句 柄 -- 他哥哥（最左边的简单短语！）

2.3 主要语法成分的定义(续3)

2.3.3 短语、简单短语和句柄



※ 设 文法 G(Z)，xαy 是一个句型则:

1. 短语 – 若 $Z \overset{+}{\Rightarrow} xAy \overset{+}{\Rightarrow} x\alpha y$ ，

则 α是句型 xαy 关于A的短语(A是α的名字)；

※任一子树的树叶全体(具有共同祖先的叶节点符号串)皆为短语；

2. 简单短语 – 若 $Z \overset{+}{\Rightarrow} xAy \Rightarrow x\alpha y$ ，

则 α是句型 xαy 关于A的简单短语(A是α的名字)；

※任一简单子树的树叶全体(具有共同父亲的叶节点符号串)皆为简单短语；

3. 句柄 – 一个句型的最左简单短语称为该句型的句柄。

※短语、简单短语和句柄示例

【例2.12】图2.3为一个算术表达式(型)的语法树：

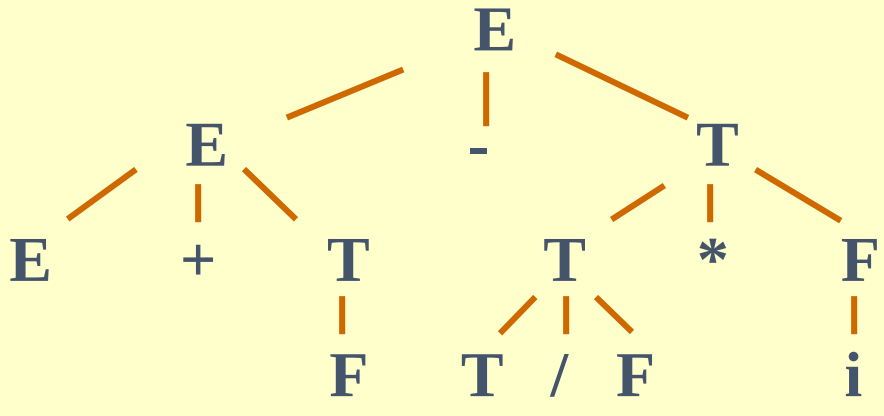


图2.3　E+F−T/F*i　的语法树

- 句型：　E+F-T/F*i
- 短语:　E+F-T/F*i，E+F，F，T/F*i，T/F，i
- 简单短语: F，T/F，i
- 句柄:　F

※ 一类典型的综合例题：

【例2.13】G(S)： S->aAcBe ； A->Ab|b ； B->d

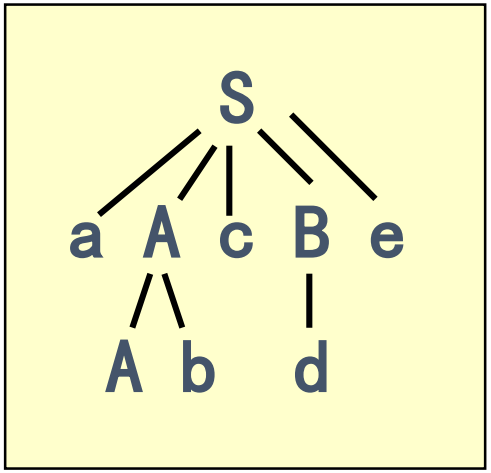
※ 给定符号串α： aAbcde

(1) 证明 α= aAbcde 是一个句型；
(2) 画出句型α 的语法树；
(3) 指出α中的短语、简单短语和句柄。

【题解】

(1) S=>aAcBe=>aAbcBe=> aAbcde

(2) 语法树如右图：



(3) 短语、简单短语和句柄：

- 短语： aAbcde ，Ab ， d
- 简单短语： Ab ， d
- 句柄： Ab

2.4 两种特性文法1

设有文法：$G(Z)=(V_N, V_T, Z, P)$

2.4.1 递归文法

【定义】 设 $A \in V_N$，x，$y \in (V_N+V_T)^*$，则；

若 $A \overset{+}{\Rightarrow} xAy$，：称文法具有**递归性**；

特别： 若 $A \to A\alpha$ ，称文法具有**直接左递归性**；

$A \to \alpha A$ ，称文法具有**直接右递归性**。

如：G1(S)：S -> S b | a ---- **直接左递归文法**；

G2(S)：S -> b S | a ---- **直接右递归文法**。

※ 递归文法是定义无限语言的工具（递归文法定义的语言有无限个句子）！！

2.4　两种特性文法2

2.4.2　二义性文法

【定义】 若文法中存在这样的句型，它具有两棵不同的语法树，则称该文法是二义性文法。

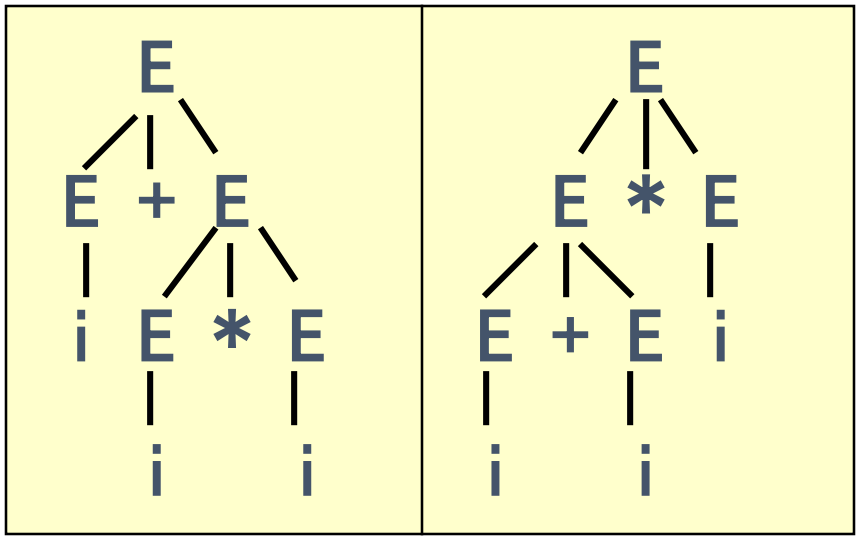
【例2.14】 算术表达式的另一种文法：

G´(E)：E -> E+E|E-E|E*E|E/E|(E)|i　;

i (变量或常数)

∵ 句型 i+i*i 有两棵不同的语法树(右图)：

∴ G`(E) 是二义性文法。



二义性文法会引起歧义，应尽量避免之！

谢谢

※ 递归文法示例

【例2.15】 G(Z)：

Z -> aAbB | cZ

A -> bBc | ε

B -> BbAc |a

∵ Z -> cZ ∴ 直接右递归性；

B -> BbAc ∴ 直接左递归性；

A =>bBc => bBbAcc

即 A =>+ αAβ ∴ 具有递归性

（ α≠ε 且 β≠ε 又称为A具有自嵌套性）

∴ 可以统称文法G(Z)具有递归性。

➢基本图形库

P:
$E \rightarrow T \mid E+T \mid E-T$
$T \rightarrow F \mid T*F \mid T/F$
$F \rightarrow i \mid (\ E\ )$

=>*, =>+ , =>.* , =>.+ , =>l* , =>l+ , =>.l+ , =>.l*