第二章　编译基础知识

廖力
xobjects@seu.edu.cn
3793235

2.0 编译基础知识

一、高级语言

程序语言是一个记号系统

- 语法
- 语义

2.0 编译基础知识（续）

二、语法

- 任何语言程序都可以看成是一定字符集（字母表）上的字符串。
- 语法使得这串字符形成一个形式上正确的程序。
- 语法＝词法规则＋语法规则
- 例如：0.5*x1+c
  - 0.5、x1、c、*、+是语言的单词符号
  - 0.5*x1+c是语言的语法单位

2.0 编译基础知识（续）

二、语法

1、单词符号

– 语言中具有独立意义的最基本结构。

- 词法规则
  - 词法规则规定了字母表中那些字符串是单词符号。
  - 单词符号一般包括：常数、标识符、基本字、算符、界限符等。
  - 我们用正规式和有限自动机理论来描述词法结构和进行词法分析。

2.0 编译基础知识（续）

二、语法

1、单词符号

2、语法单位

– 表达式、子句、语句、函数、过程、程序

- 语法规则
  - 规定了如何从单词符号来形成语法单位。
  - 现在多数程序语言使用上下文无关文法来描述语法规则。
  - 语言的词法规则和语法规则定义了程序的形式结构，是判断输入字符串是否构成一个形式上正确的程序的依据。

2.0 编译基础知识（续）

三、语义

- 对于一个语言来说，不仅要给出它的词法、语法规则，而且要定义它的单词符号和语法单位的意义。
- 离开语义，语言只是一堆符号的集合。
- 各种语言中有形式上完全相同的语法单位，含义却不尽相同。
- 对某种语言，可以定义一个程序的意义的一组规则称为语义规则。
- 目前，大多数编译程序使用基于属性文法的语法制导翻译方法来分析语义。

2.0 编译基础知识（续）

- 对于高级程序设计语言及其编译程序来说，语言的语法定义是很重要的。本章主要介绍语法结构的形式描述问题，编译原理的主要内容也可以归结为应用形式语言理论，并将它贯串于词法分析和语法分析两个阶段。
- 本章重点讨论正规文法、上下文无关文法及其对应的有限自动机和下推自动机，以及在构造编译程序时如何应用。

2.1　字母表与符号串

2.1 字母表与符号串

一、相关概念

1、字母表

- 是符号的非空有穷集合。
- 用Σ、V表示。

2、符号：

是语言中最基本的不可再分的单位。

3、符号串：

- 符号串是字母表中符号组成的有穷序列。
- 空串：不含有任何符号的串称作空串，记作ε。

2.1　字母表与符号串

一、相关概念

4、句子：

字母表上符合某种规则构成的串。

5、语言：

字母表上句子的集合。

- 注：约定用a,b,c…表示符号；用α,β,γ…表示符号串；用A,B,C…表示其集合。

2.1 字母表与符号串

二、符号串集合的运算

1、连接（乘积）运算：

– 定义：若串集$A=\{\alpha_1, \alpha_2, \ldots\}$，串集$B=\{\beta_1, \beta_2, \ldots\}$，则乘积$AB=\{\alpha\beta | \alpha \in A \text{ and } \beta \in B\}$

注：1）串集的自身乘积称作串集的方幂

2）$A^0=\{\varepsilon\}$

3）字母表A的n次方幂是字母表A上所有长度为n的串集

2.1　字母表与符号串

二、符号串集合的运算

- 例如：A={a,b};　B={c,e,d}
- 则AB={ac,ae,ad,bc,be,bd}
- 例如：串集A＝{a}的各次方幂定义为：
  - $A^0=\{\varepsilon\}$
  - $A^1=A=\{a\}$
  - ……
  - $A^n=AA^{n-1}(n>0)=\{a\ldots a\}$

2.1 字母表与符号串

三、字母表的闭包与正闭包

- 1）字母表A的闭包（A^*）：

  $A^*=A^0 \cup A^1 \cup A^2 \cup \ldots$

  – 即：由A上符号组成的所有串的集合（包括空串ε）。

- 2）字母表A的正闭包（A^+）：

  – $A^+= A^1 \cup A^2 \cup \ldots = A^* - \{\varepsilon\}$

  – 即：由A上符号组成的所有串的集合（不包括空串ε）。

- 3）语言：是字母表上符合某种规则的语句组成的。

  – 字母表上语言：是字母表上正闭包的子集。

2.2 文法与语言的关系

2.2 文法与语言的关系

一、文法的概念

1、文法

文法是描述语言的语法结构的形式规则。

- 例如：我们写这样一个句子：
  - Young men like pop music.

2.2 文法与语言的关系

一、文法的概念

1、文法

- 其语法规则如下：
  - <句子>→<主语><谓语>
  - <主语> →<形容词><名词>
  - <谓语> →<动词><宾语>
  - <宾语> → <形容词><名词>
  - <形容词> →Young | pop
  - <名词> →men | music
  - <动词> →like

2.2 文法与语言的关系

一、文法的概念

2、相关概念

- (1)非终结符
  - 出现在规则的左部、用<>括起来、表示一定语法概念的词。
  - 非终结符集合用V_N表示。
- (2)终结符
  - 语言中不可再分割的字符串(包括单个字符组成的串)。注：终结符是组成句子的基本单位。
  - 终结符集合用V_T表示。

2.2 文法与语言的关系

一、文法的概念

2、相关概念

- (3)开始符号
  - 表示所定义的语法范畴的非终结符。
  - 注：开始符号又称为识别符号。
- (4)产生式
  - 是用来定义符号串之间关系的一组(语法)规则。
  - 形式：$A \rightarrow \alpha$ （A产生α）

2.2 文法与语言的关系

一、文法的概念

2、相关概念

- （5）推导
  - 推导是从开始符号开始，通过使用产生式的右部取代左部，最终能产生语言的一个句子的过程。
  - 最左（右）推导：每次使用一个规则，以其右部取代符号串最左（右）非终结符。

注：最左推导和最右推导称为规范推导。

2.2 文法与语言的关系

一、文法的概念

2、相关概念

- （6）归约
  - 归约是推导的逆过程，即，从给定的源语言的句子开始，通过规则的左部取代右部，最终达到开始符号的过程。
  - 最左(右)归约是最右(左)推导的逆过程。
  - 注：最左归约和最右归约称为规范归约。

2.2 文法与语言的关系

一、文法的概念

2、相关概念

(7)句型、句子和语言

- 句型：
  - 句型是从文法的开始符号S开始，每步推导（包括0步推导）所得到的字符串α。
  - 记作：$S \xrightarrow{*} \alpha$，其中$\alpha \in (V_N \cup V_T)^*$
- 句子：是仅含终结符的句型。

•例如：根据英语的语法规则，看能否用最左推导得出“Young men like pop music”是个语法正确的句子；在推导过程中有哪些句型。

- 语法规则为：
  - <句子>→<主语><谓语>
  - <主语> →<形容词><名词>
  - <谓语> →<动词><宾语>
  - <宾语> → <形容词><名词>
  - <形容词> →Young | pop
  - <名词> →men | music
  - <动词> →like

对上句的最左推导

<句子>→<主语><谓语>

→ <形容词><名词> <谓语>

→ Young <名词> <谓语>

→Young men <谓语>

→Young men <动词><宾语>

→Young men like<宾语>

→Young men like <形容词><名词>

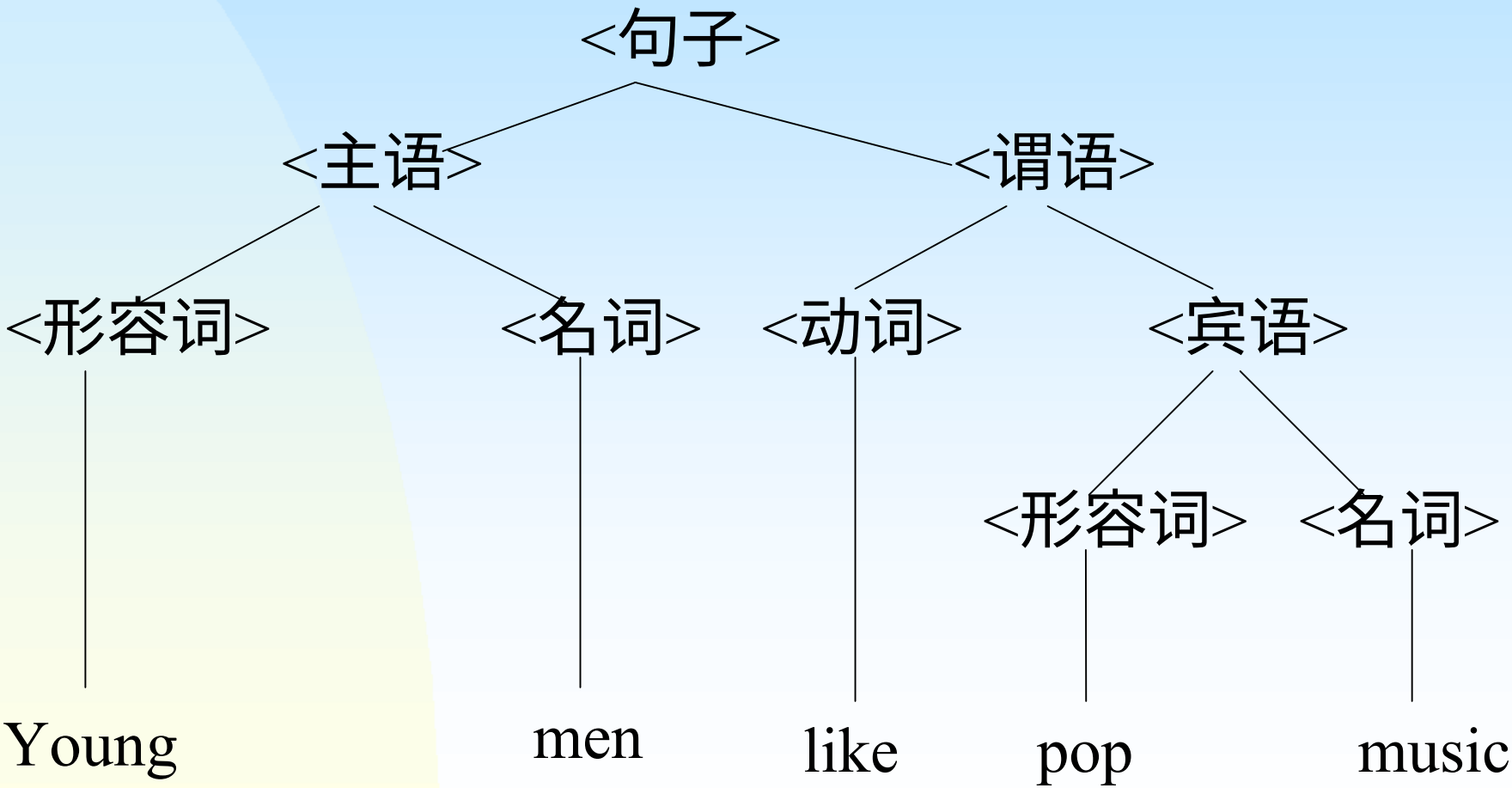
→Young men like pop <名词>

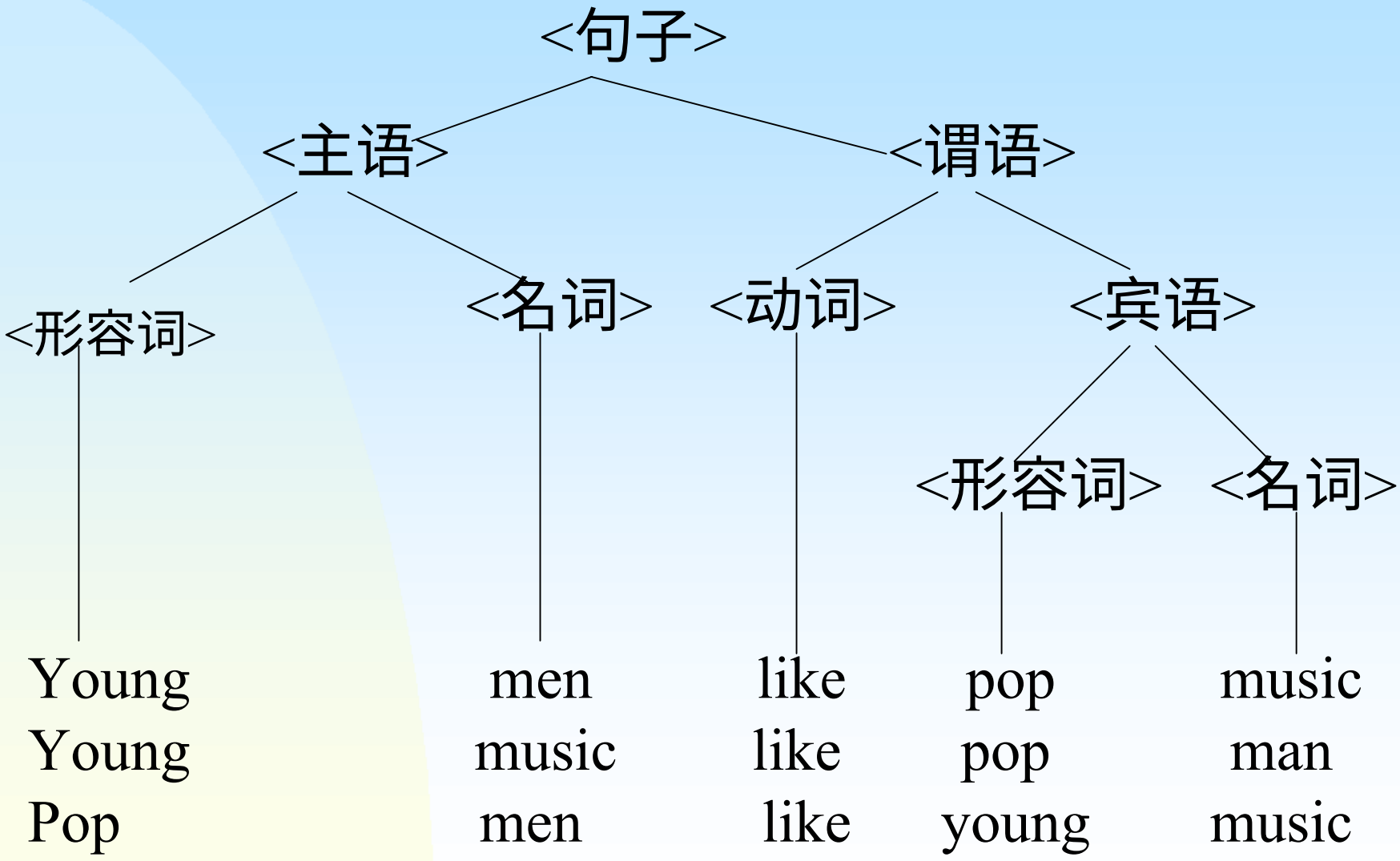
→Young men like pop music

最左推导

最右归约

用图示化方式表示




<句子>
<主语>
<谓语>
<形容词>
<名词>
<动词>
<宾语>
<形容词>
<名词>
Young
Young
Pop
men
music
men
like
like
like
pop
pop
young
music
man
music

……

2.2 文法与语言的关系

一、文法的概念

2、相关概念

(7)句型、句子和语言

- 语言：
  - 语言是由S开始通过1步或1步以上推导所得的句子的集合。
  - 记为：L(G)。$L(G)=\{\alpha|S\xrightarrow{+}\alpha，且\alpha\in V_T^*\}$

(8)文法规则的递归定义

- 非终结符的定义中包含了非终结符自身。
- 使用文法的递归定义要谨慎

- 例如：字母表A＝{0,1},

  语法规则为：

  <整数> → <数字><整数>|<数字>

  <数字> →0|1

- 再如：字母表A＝{0,1},

  语法规则为：

  <整数> → <数字><整数>

  <数字> → 0|1

2.2 文法与语言的关系

一、文法的概念

2、相关概念

(8)文法规则的扩充表示

——扩充的BNF表示

- () ——提因子
  - 例:把U→ax|ay|az 改写为U→a(x|y|z)
- {} ——重复次数的指定
  - 例：<标识符>→<字母>{<字母>|<数字>}$^{5}_{0}$.
- [] ——任选符号
  - 例:<整数>→[+|-]<数字>{<数字>}

2.2 文法与语言的关系

一、文法的概念

2、相关概念

(9)元语言符号

用来说明文法符号之间关系的符号，如，“→”和“|”称为元语言符号。

2.2 文法与语言的关系

二、文法与语言的形式定义

1、Chomsky对文法的定义

从形式上说文法G是一个四元式(V_N,V_T,P,S)

2、Chomsky对文法的分类

根据对产生式施加的限制，可分为

- 0型文法
- 1型文法
- 2型文法
- 3型文法

2.2 文法与语言的关系
二、文法与语言的形式定义

2、Chomsky对文法的分类

(1) 0型文法(短语文法或无限制文法)

- P中产生式$\alpha \to \beta$ 其中$\alpha \in V^+$ 并至少含有一个非终结符, $\beta \in V^*$.
- 注：a)识别0型语言的自动机称为图灵机(TM).
- b)0型文法是对产生式限制最少的文法。
- c)任何0型语言都是递归可枚举的。
- d)对0型文法产生式的形式作某些限制，可得到其他类型文法的定义。

2.2 文法与语言的关系
二、文法与语言的形式定义

2、Chomsky对文法的分类

(2) 1型文法

- P中产生式$\alpha \to \beta$,除可能有$S \to \varepsilon$ 外均有$|\beta| >= |\alpha|$,若有$S \to \varepsilon$，规定S不得出现在产生式右部 .
- 或者，P中产生式$\alpha \to \beta$,除可能有$S \to \varepsilon$外均有$\alpha A \beta \to \alpha \gamma \beta$,其中$\alpha, \beta \in V^*$,$A \in V_N$,$\gamma \in V^+$.
- 注：a)1型文法又称为长度增加文法、上下文有关文法; b)识别1型语言的自动机称为线性界限自动机(LBA) ； c)1型文法意味着，对非终结符进行替换时务必考虑上下文，并且，一般不允许替换成ε，除非是开始符号产生ε

1型文法举例

- 设文法G＝(V_N,V_T,P,S)，
- V_N ＝{S,B,E}, V_T ＝{a,b,e}
- 其中P为: (0) S →aSBE
- (1) S →aBE
- (2) EB →BE
- (3) aB →ab
- (4) bB →bb
- (5) bE →be
- (6) eE →ee

2.2 文法与语言的关系
二、文法与语言的形式定义

2、Chomsky对文法的分类

(3) 2型文法

- P中产生式具有形式$A\rightarrow\beta$ 其中$A\in V_N$， $\beta\in V^*$.
- 注：a)2型文法对产生式的要求是：产生式左部一定是非终结符，产生式右部可以是V_N、 V_T或ε ；非终结符的替换不必考虑上下文;
- b)识别2型语言的自动机称为下推自动机(PDA);
- c) 2型文法也称为上下文无关文法。

2型文法举例

- 设文法G＝(V_N,V_T,P,S)，
- V_N ＝{S,A,B}, V_T ＝{a,b}
- 其中P为: (0) S →aB
- (1) S →bA
- (2) A →a
- (3) A →aS | bAA
- (4) B →b
- (5) B →bS | aBB

2.2 文法与语言的关系

二、文法与语言的形式定义

2、Chomsky对文法的分类

(4) 3型文法

- P中产生式具有形式$A\rightarrow \alpha B$， $A\rightarrow \alpha$ ，或者$A\rightarrow B\alpha$，$A\rightarrow \alpha$， 其中A，$B\in V_N$, $\alpha\in V_T^*$。
- 注： a)3型文法也称为正规文法RG、右线性文法或左线性文法；
- b) 3型文法中的产生式要么均是右线性产生式，要么是左线性产生式，不能既有左线性产生式，又有右线性产生式；若所有产生式均是左线性，则称为左线性文法；若所有产生式均是右线性，则称为右线性文法；
- c)识别3型语言的自动机称为有限状态自动机。

2.2 文法与语言的关系

二、文法与语言的形式定义

3、i型语言

- 由i型文法生成的语言成为i型语言。
- 记为：$L(G)$；$L(G)=\{w|w \in V_T^*$，且$S \overset{+}{\rightarrow} w\}$

- 例1：设文法$G_1=(\{S\},\{a,b\},P,S)$
- 其中P为: (0) $S \rightarrow aS$
- (1) $S \rightarrow a$
- (2) $S \rightarrow b$

$L(G_1)=\{a^i(a|b)|i>=0\}$

- 例2：设文法$G_2=(\{S\},\{a,b\},P,S)$
- 其中P为: (0) $S \rightarrow aSb$
- (1) $S \rightarrow ab$

$L(G_2)=\{a^nb^n|n>=1\}$

2.2 文法与语言的关系

二、文法与语言的形式定义

- 注意：在词法分析和语法分析中对产生式有限制：
  - 不存在P $\rightarrow$P产生式
  - 产生式中出现的任何非终结符P必须有用。
    - 从开始符号S出发，存在推导S $\overset{*}{\rightarrow}\alpha P\beta$
    - P必须能推导出终结符串。即: P $\overset{+}{\rightarrow}\gamma$; $\gamma \in V_T^*$

2.3 文法构造与文法简化

2.3 文法构造与文法简化

一、如何由语言构造文法

- 例2.6：设$L_1=\{a^{2n}b^n|n>=1$ 且$a,b \in V_T\}$
  试构造生成L_1的上下文无关文法G_1

解：设n=1， L_1 =aab

n=2， L_1 =aaaabb

n=3， L_1 =aaaaaabbb

……

所以得：$S \rightarrow aaSb$

– $S \rightarrow aab$

2.3 文法构造与文法简化

一、如何由语言构造文法

- 例2.7：设$L_2=\{a^ib^jc^k \mid i,j,k>=1$ 且$a,b,c \in V_T\}$试构造生成L_2的文法G_2

解：$S \rightarrow aS$

$S \rightarrow aB$

$B \rightarrow bB$

$B \rightarrow bC$

$C \rightarrow cC \mid c$

2.3 文法构造与文法简化

一、如何由语言构造文法

- 例2.8：设$L_3=\{\omega \mid \omega \in (a,b)^*$ 且ω中含有相同个数的a和b$\}$试构造生成L_3的上下文无关文法G_3

解：$S \to \varepsilon$

$S \to bB \mid aA$

$A \to bS \mid aAA$

$B \to aS \mid bBB$

(0) $S \to \varepsilon$

(1) $S \to aSbS$

(2) $S \to bSaS$

2.3 文法构造与文法简化

一、如何由语言构造文法

例：设$L_4=\{\omega \mid \omega \in(0,1)^*$ 且ω中1的个数为偶数$\}$试构造生成L_4的文法G_4

解：$S \to \varepsilon$

$S \to 0S$，$S \to 1A$

$A \to 0A$，$A \to 1S$

2.3 文法构造与文法简化

二、文法的简化

1、为什么要进行文法的简化？

- 由于同一语言可以用不同的文法来描述，显然应当选择产生式的个数最少，最符合语言特征的来描述。
- 在文法中，有些产生式对推导不起作用，要删除掉：
  - 如某个产生式在推导过程中永远不会被用到，即由开始符号推导，永远推不到的左部的非终结符。
  - 再如永远导不出终结符串的产生式。
  - 如形如$P \rightarrow P$的产生式。

2.3 文法构造与文法简化

二、文法的简化

2、简化步骤：

- 查找有无形如P→P的产生式，若有则删除；
- 若某个产生式在推导过程中永远不会被用到，删除它；
- 若某个产生式在推导过程中不能从中导出终结符，删除它；
- 最后，整理所有剩余产生式，就得到简化的文法。

2.3　文法构造与文法简化

二、文法的简化

- 例10：化简下面的文法。
  - (0)S → Be　(1)S → Ec　(2)A → Ae　(3)A →e
  - (4)A →A　(5)B →Ce　(6)B →Af　(7)C →Cf
  - (8)D →f
- (0) S → Be　(1)A → Ae　(2)A →e　(3)B →Af

2.3 文法构造与文法简化

三、构造无ε产生式的上下文无关文法

1、为什么要构造无ε产生式的上下文无关文法？

2、无ε产生式的上下文无关文法要满足条件

– P中要么不含有ε产生式，要么只有$S \to \varepsilon$；

– 若$S \to \varepsilon$，则S不出现在任何产生式右部。

3、构造无ε产生式的上下文无关文法变换算法：

– $G=(V_N,V_T,P,S) \Rightarrow G'=(V'_N,V'_T,P',S')$

（1）由文法G找出所有经过若干步推导能推出ε的非终结符，放在V_0集合中。

（2）再按下列步骤构造G'的产生式集合P'；

2.3 文法构造与文法简化
三、构造无ε产生式的上下文无关文法

3、构造算法：

（2）再按下列步骤构造G’的产生式集合P’

- A)若V_0集合中的某元素出现在某产生式的右端，则将它变成两个产生式：分别以ε和其原型代入；将新产生式加入P’
- B)不满足上一条的P中其他产生式除去ε产生式后也加入P’
- C)如果P中有产生式$S \rightarrow \varepsilon$，将它在P’中改为$S' \rightarrow \varepsilon \mid S$，S’是新的开始符号，把它加入$V_N$，形成$V'_N$

- 例11：设G1=({S},{a,b},P,S),其中
  - P: (0) S → ε (1) S →aSbS (2) S →bSaS
- (1)V_0={S}
- (2)P’: (1) → S →abS|aSbS|aSb|ab
- (2) → S →baS|bSaS|bSa|ba
- (0) → S’ → ε | S
- 故：文法G1’=({S’,S},{a,b},P’,S’),其中
- P’: (0) S’ → ε | S
- (1) S →abS|aSbS|aSb|ab
- (2) S →baS|bSaS|bSa|ba

2.4 语法树与文法的二义性

2.4 语法树与文法的二义性

一、语法树

1、定义：

– 用来表示语言句子结构的树。

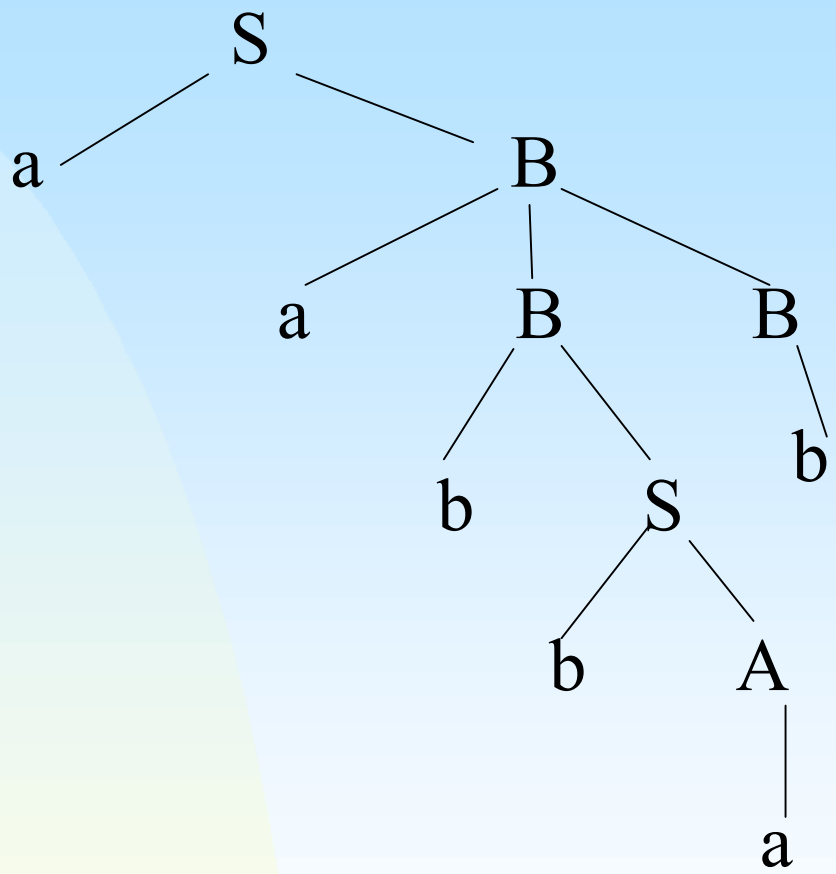
2、作用：

– 使用语法树可以使语法分析过程直观、形象，易于判断文法二义性。

2.4 语法树与文法的二义性

一、语法树

- 例如，有一个2型文法
  - G= ({S,A,B},{a,b},P,S),其中P:
  - (0) S → aB|bA　(1)A →a|aS|bAA　(2)B →b|bS|aBB
- 采用最左推导产生句子aabbab:
  - S → aB → aaBB → aabSB → aabbAB → aabbaB → aabbab



back

$S \rightarrow aB \rightarrow aaBB \rightarrow aabSB \rightarrow aabbAB \rightarrow aabbaB \rightarrow aabbab$

2.4 语法树与文法的二义性

一、语法树

3、语法树中的概念

(1) 子树：除叶子结点之外的任意结点连同它的所有子孙结点构成子树。

(2) 修剪子树：剪去子树树根的所有孩子。

(3) 句型：在一棵语法树生长过程中的任何时刻，所有那些叶子结点排列起来就是一个句型。

2.4 语法树与文法的二义性

一、语法树

3、语法树中的概念

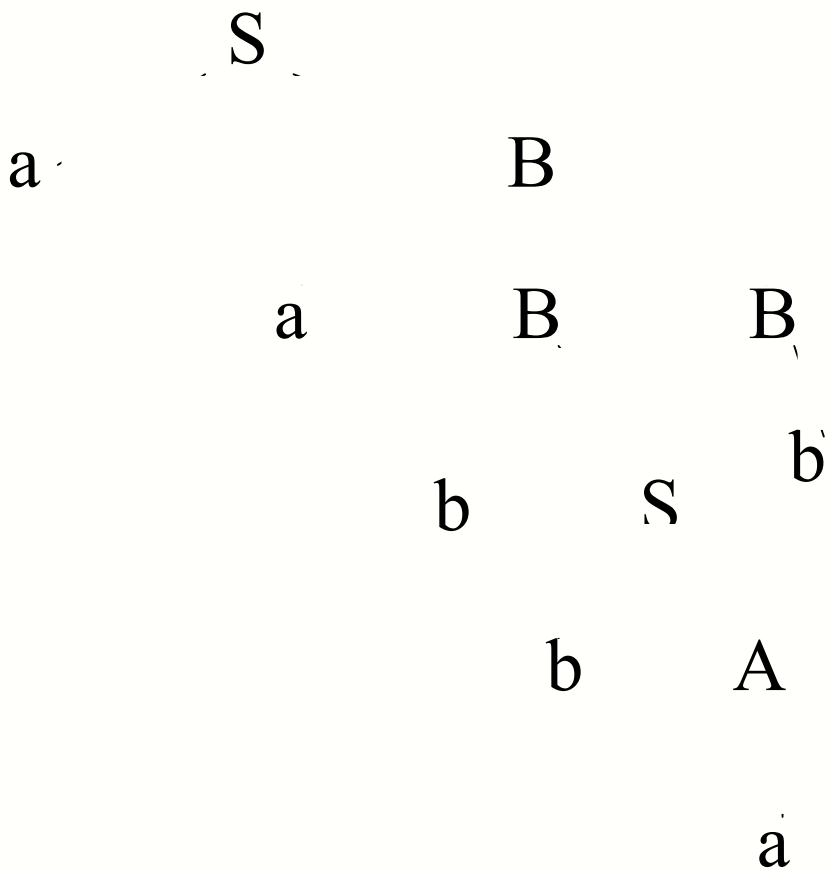
（4）短语：子树的末端符号自左到右连成串，相对于子树树根而言称为短语。

- 简单短语（直接短语）：若短语是某子树根经过1步推导得到的，则称之为该子树根的简单短语。
- 句型的短语：该句型中哪些符号串可构成某子树根的短语。

（5）句柄：句型中的最左简单短语。

注：句柄是最左归约时要寻找的简单短语。

back



2.4 语法树与文法的二义性

二、文法的二义性

1、句子二义性：

– 如果文法的一个句子存在对应的两棵或两棵以上的语法树，则该句子是二义的。

2、文法二义性：

• 包含二义性句子的文法是二义文法。

2.4 语法树与文法的二义性

二、文法的二义性

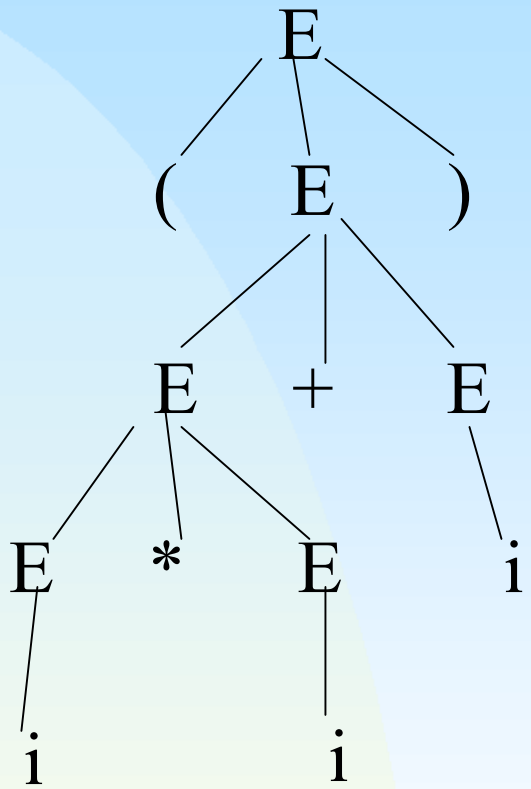
- 例如：文法G=({E}，{+,*,(,),i}，P，E)

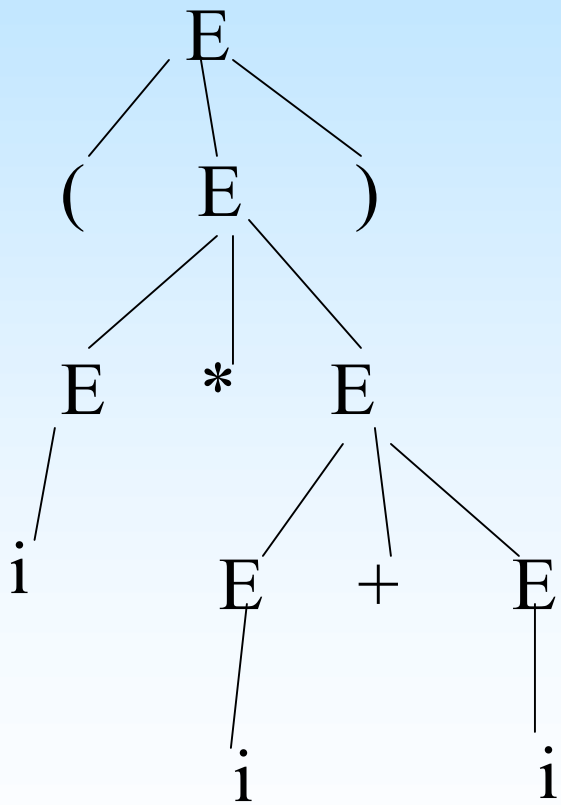
  其中：E →E+E | E*E | (E) | i

  问：对于句子（i* i+ i)有几种最左推导

解：1） E →(E) →(E+E) →(E*E+E) →(i*E+E) →(i*i+E) →(i* i+ i)

2） E →(E) →(E*E) →(i*E) →(i*E+E) →(i*i+E) →(i* i+ i)


E
(
E
)
E
+
E
E
*
E
i
i
i


E
(
E
)
E
*
E
i
E
+
E
i
i

2.4 语法树与文法的二义性

二、文法二义性

注：1）二义性会给语法分析带来不确定性。

2）文法的二义性是不可判定的，即不存在算法，能够在有限步数内确切判定一个文法是否为二义文法。

3）若要证明是二义性，只要举出一例即可。

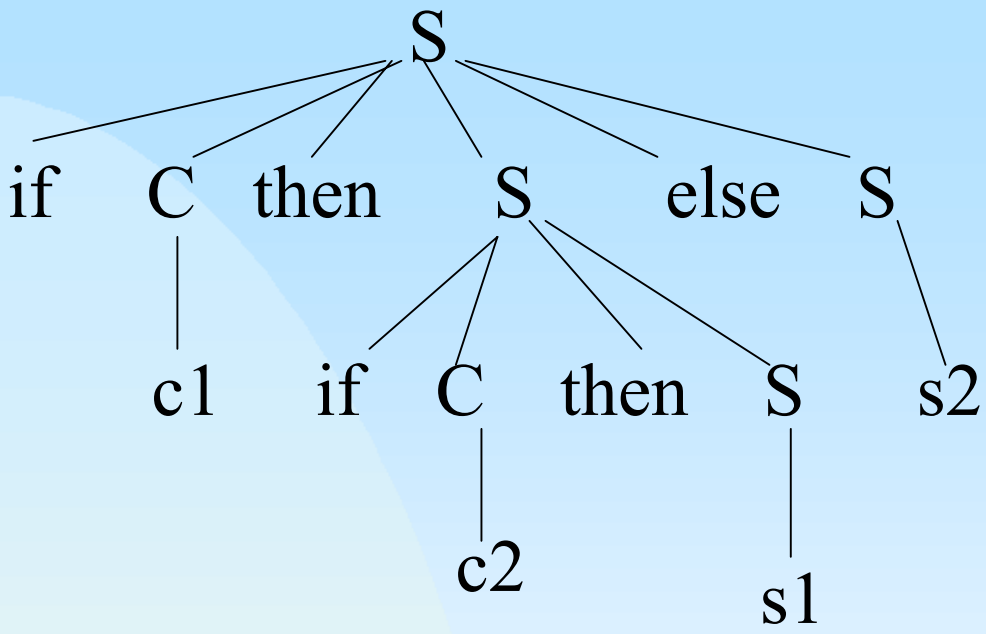
4）若能控制文法的二义性，即加入人为的附加条件，则二义文法的存在并非坏事。

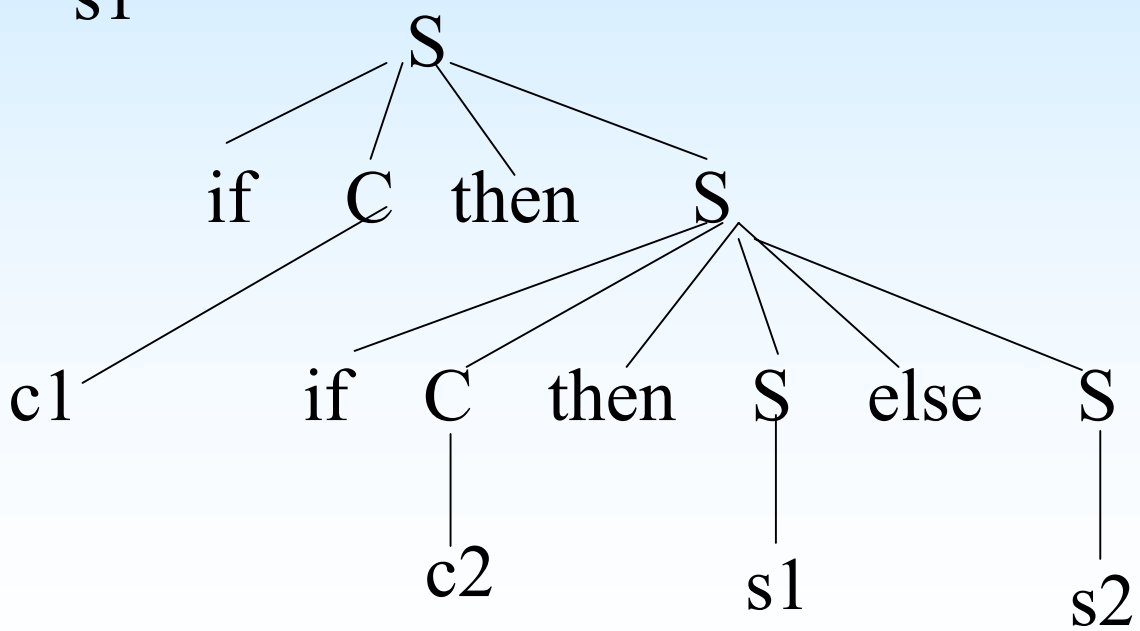
2.4 语法树与文法的二义性

二、文法的二义性

- 例如：if语句结构采用下面的产生式：
  - S →if C then S else S
  - S →if C then S
  - S →其他语句　　C →具体条件表达式

  判断它是不是二义性文法？
- 解法：只要找到一个例子，一个句子有两个语法树，就可以证明它是二义性文法。
  - 例如：if c1 then if c2 then s1 else s2


S
if
C
then
S
else
S
c1
if
C
then
S
s2
c2
s1


S
if
C
then
S
c1
if
C
then
S
else
S
c2
s1
s2

小结

1、Chomsky文法：（重点）
   – 主要掌握上下文无关文法和正规文法及所对应的自动机类型。
2、由语言构造文法：（重点＋难点）
   – 包括上下文无关文法和正规文法
   – 将上下文无关文法改写为正规文法
3、根据算法构造无ε产生式的上下文无关文法。
4、判断文法二义性。