# 第一章作业

1. **解释下列术语:编译程序,源程序,目标程序,编译程序的前端、后端和遍。**

**答：**

**(1)编译程序：源语言为高级语言，目标语言为汇编语言或机器语言的翻译程序称为编译程序。**

**(2)源程序：待翻译的程序，即高级程序设计语言编写的程序。**

**(3)目标程序：翻译后的程序，即汇编语言或机器语言。**

**(4)前端：编译过程中主要依赖于源语言与目标机无关的阶段，包括词法分析、语法分析、语义分析和中间代码生成。某些优化工作也可在前端做，也包括与前端每个阶段相关的出错处理工作和符号表管理等工作。**

**(5)后端：指依赖于目标机而一般不依赖源语言，只与中间代码有关的阶段，即目标代码生成，以及相关出错处理和符号表操作。**

**(6)遍：是对源程序或其等价的中间语言程序从头到尾扫视并完成规定任务的过程。**

1. **编译程序有哪些主要构成成分?各自的主要功能是什么?**

**答：**

**编译程序的主要构成成分有: 词法分析程序、语法分析程序、语义分析程序、中间代码生成程序、代码优化程序、目标代码生成程序、表格管理程序及出错处理程序。**

**(1)词法分析程序：从左到右扫描源程序，识别出各个单词，确定单词的类型并将其转换成单词串；同时查词法错误，进行标识符登记，即符号表管理。**

**(2)语法分析程序：识别由词法分析给出的单词符号串是否是给定文法的句子**

**(3)语义分析程序：审查源程序是否有语义错误，为代码生成阶段收集类型信息，当不符合规范的时候报错。**

**(4)中间代码生成程序：将源程序转换成一种内部表示形式，如三地址指令或四元式。**

**(5)中间代码优化程序：对中间代码进行等价变换处理以提高执行效率。**

**(6)目标代码生成程序：将优化的中间代码转换成目标机上的机器指令代码或汇编代码。**

**(7)表格管理程序：管理各种符号表（常数、标号、变量、过程、结构……），查、填（登记、查找）源程序中出现的符号和编译程序生成的符号，为编译的各个阶段提供信息。**

**(8)错误处理程序：进行各种错误的检查、报告、纠正，以及相应的续编译处理。**

1. **什么是解释程序?它与编译程序的主要不同是什么?**

**答：**

**解释程序直接执行源程序给出运行结果。工作模式：一个个的获取、分析并执行源程序语句，一旦第一个语句分析结束，源程序便开始运行并且生成结果。**

**不同：编译程序将源高级语言程序翻译成汇编或机器语言程序，而解释程序则是分析处理源高级语言程序直接计算结果，不生成目标语言程序。**

1. **对下列错误信息,请指出可能是编译的哪个阶段(词法分析、语法分析、语义分析、代码生成)报告的。**

**（1）else没有匹配的if。**

**（2）数组下标越界。**

**（3）使用的函数没有定义。**

**（4）在数中出现非数字字符。**

**答：**

|  |  |
| --- | --- |
| **没有匹配的** | **语法分析** |
| **数组下标越界** | **语义分析** |
| **使用的函数没有定义** | **语义分析** |
| **在数中出现非数字字符** | **词法分析** |

# 第二章作业

1. **文法，其中为**



**写出的全部元素。**

**答：**

1. **文法为**

**的语言是什么?**

**答：**

**,其中，即长度大于等于*1*的数字串**

**3. 为只包含数字、加号和减号的表达式,例如、、等构造一个文法。**

**答：**

**(注：表达同一语言的文法可能不唯一)**

**5. 已知文法：**

**写出的全部元素。**

**答：**

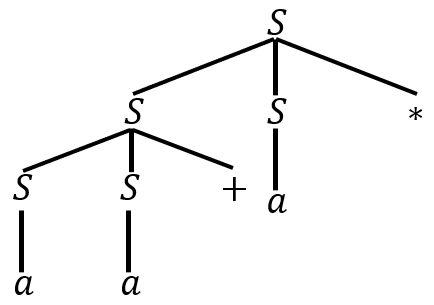
**8. 考虑下面的上下文无关文法:**

**(1)表明通过此文法如何生成串,并为该串构造语法树。**

**(2)该文法生成的语言是什么?**

**答：(1)推导：**

**语法树如下图所示：**



**(2)该文法生成的语言是符号的和运算的后缀表达式。**

**9. 已知文法。**

**(1)该文法生成的语言是什么?**

**(2)该文法是二义的吗?说明理由。**

**答：**

**(1) 该文法生成的语言是匹配的括号**

**(2) 是二义的,因为对于可以构造两棵不同的语法树**

**10. 令文法为**

**证明是它的一个右句型,指出这个句型的所有短语、直接短语和句柄。**

**答：**

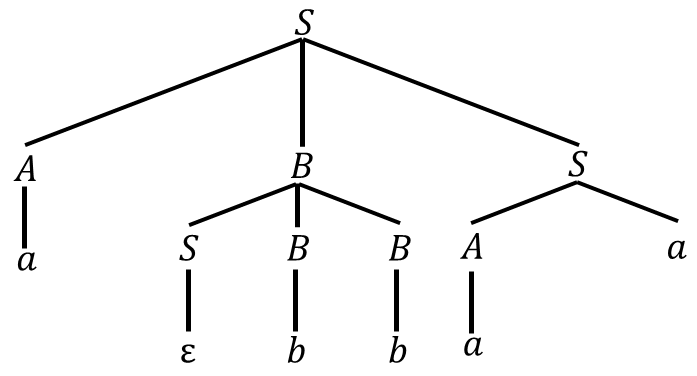
**最右推导为：**，**所以是该文法的一个右句型。**

**短语:**

**直接短语：**

**句柄:**

**11.一个上下文无关文法生成句子的唯一语法树如下:**

****

**(1)给出该句子相应的最左推导和最右推导。**

**(2)该文法的产生式集合可能有哪些元素?**

**(3)找出该句子的所有短语、简单短语、句柄。**

**答：**

**(1)最左推导：**

**最右推导:**

**(2)产生式有：**

**(3)短语：,,,,,**

**直接短语：,,**

**句柄：**

**12.构造产生如下语言的上下文无关文法各一个:**

**(1)**

**(2)**

**(3)**

**(4)**

**(5) 且**

**(6) ,其中,表示的反向串,其含义是将中的字母依次反转,首尾字母交换位置,下同**

**(7)**

**(8)**

**答：**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
| **且** |  |
|  |  |
|  |  |
|  |  |

# 第三章作业

1.构造下列正规式相应的DFA。

**（1）**

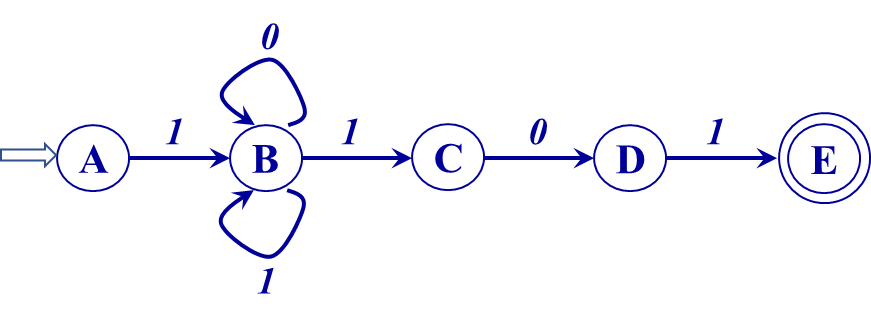
**（2）**

（3）

**（4）**

**答**：

**(1)正规式对应的NFA如下：**

****

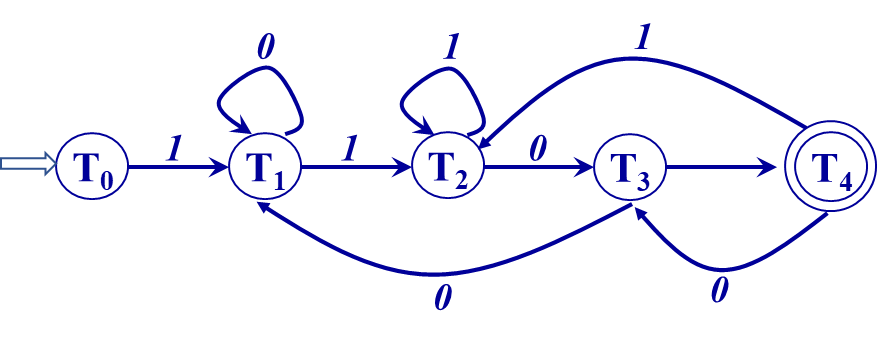
**用子集法将此NFA确定化的状态转换表如下：**

|  |  |  |
| --- | --- | --- |
| 输入符号  状态集 | 1 | 0 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**初态：**

终态：

**生成的如下为：**

****

**分割法最小化**

**（2）正规式**对应的NFA如下：



**用子集法将此NFA确定化的状态转换表如下：**

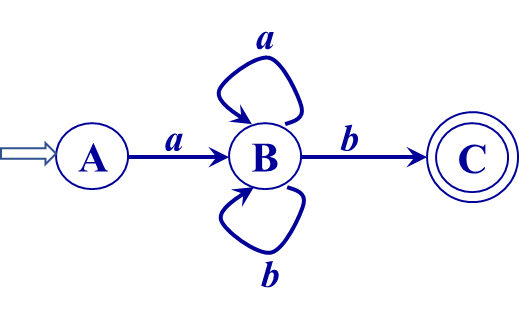
|  |  |  |  |
| --- | --- | --- | --- |
| 状态集 | 符号 |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**初态：**

终态：,,,,

**生成的如下为：（图略）**

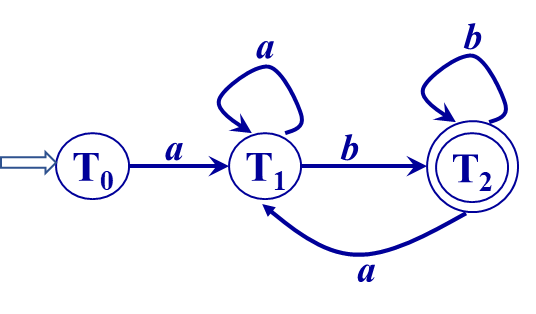
**(3)正规式对应的NFA如下：**

****

**用子集法将此NFA确定化的状态转换表如下：**

|  |  |  |
| --- | --- | --- |
| 状态集 | 符号 |  |
|  |  |  |
|  |  |  |
|  |  |  |

**生成的如下为：**



**分割法最小化**

**（4）正规式**对应的NFA为：



|  |  |  |
| --- | --- | --- |
| 输入符号  状态集 | a | b |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

初态：

终态：

**分割法最小化**

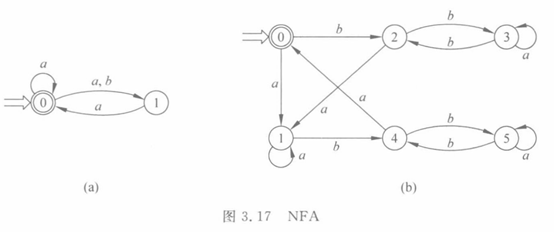
和等价

令:

**生成的如下为：**



4.把图3.17(a)和(b)中的NFA分别确定化和最小化。



**（a）用子集法将此NFA确定化的状态转换表如下：**

|  |  |  |
| --- | --- | --- |
| 状态集 | 符号 |  |
|  |  |  |
|  |  |  |
|  |  |  |

**初态：**

终态：,

**分割法最小化**

和等价,删掉用代替出现的位置。

**生成的最简如下为：**



**（b）此FA是DFA。**

采用分割法进行最小化：

第一次分割：{1,2,3,4,5} {0}

状态4输入为a时结果为0，将状态4分割出去。

第二次分割：{1,2,3,5}{4}{0}

对于划分{1,2,3,5}输入为b时将其划分为{1,5}{2,3}

第三次分割：{1,5}{2,3}{4}{0}

对于划分{2,3}输入为a时可分割为{2}{3}

最终结果为：{1,5}{2}{3}{4}{0}

删除状态5，用状态1代替状态5 （状态转换图略）

5.构造一个DFA,它接受上所有满足如下条件的字符串：每个1都有0直接跟在右边。然后构造该语言的正规文法。

正规式为：（0|10）\*

正规文法：

构造的DFA为：



1. **为正规文法**

**T**

**T**

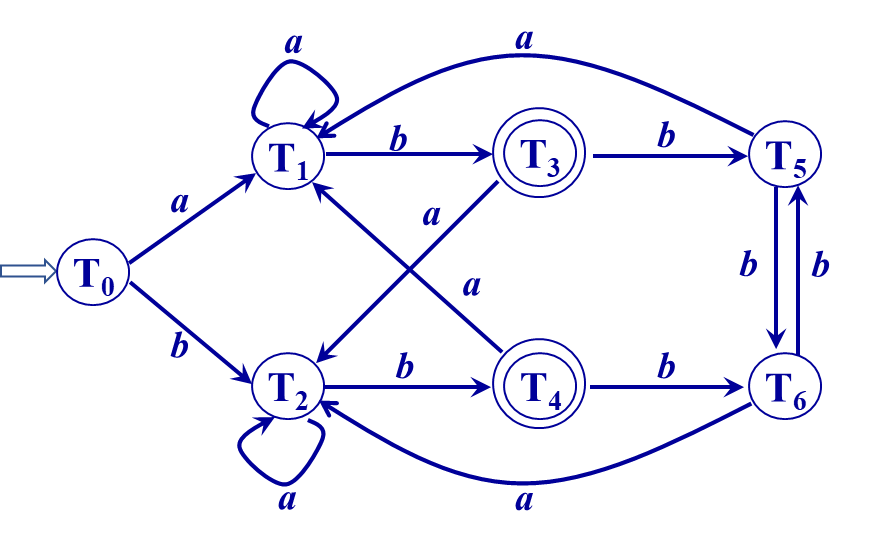
**T T是状态图中的终态**

**构造相应的最小的**

**答：用子集法将此NFA确定化的状态转换表如下：**

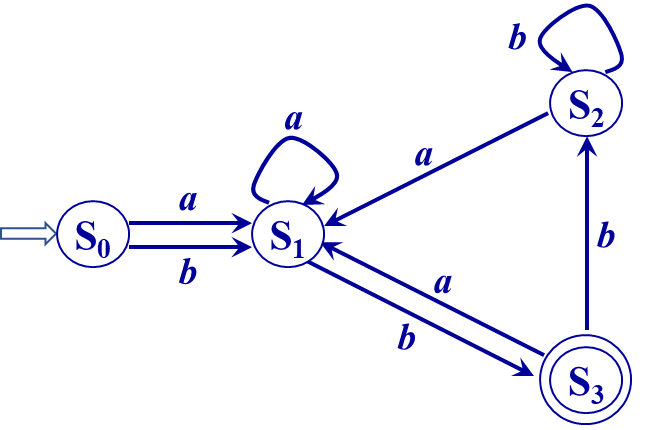
|  |  |  |
| --- | --- | --- |
| 状态集 | 符号 |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**生成的如下为：**

****

**分割法最小化：**

**令**，**则最小化的：**



1. **给出下述正规文法所对应的正规式：**

**答：将A和B的产生式代入S中：**

**对S进行求解：**

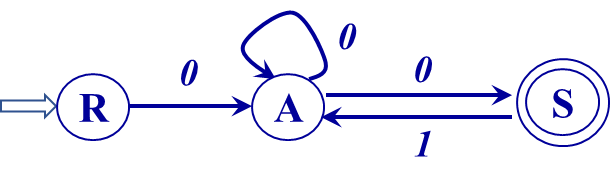
1. **构造下述文法的自动机：**

**该自动机是确定的吗？若不确定，则对它确定化。该自动机相应的语言是什么？**

**说明：产生式形式为或的文法也是正规文法，并称为左线性文法。为左线性文法构造的规则如下：**

1. **字母表与G的终结符集相同。**
2. **中的每个非终结符生成的一个状态。**
3. **G的开始符对应的终态。**
4. **增加一个新的状态，作为的初态。**
5. **对G中的形如的产生式，构造的转换函数；对，构造。**

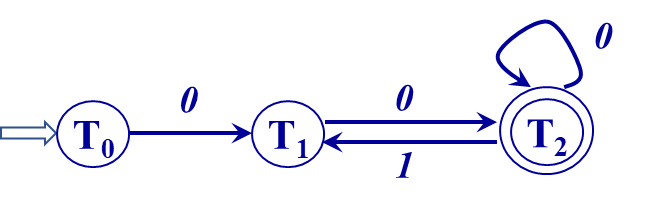
**答：NFA如下：**

****

**用子集法将此NFA确定化的状态转换表如下：**

|  |  |  |
| --- | --- | --- |
| 状态集 | 符号 |  |
|  |  |  |
|  |  |  |
|  |  |  |

**DFA如下：**

****

**此DFA所识别的语言的正规式为：**

12.文法G[<单词>]为

<单词>→<标识符> | <整数>

<标识符>→<标识符><字母> | <标识符><数字> | <字母>

<整数>→<整数><数字> | <数字>

<字母>→A|B|…|Y|Z

<数字>→0|1|2|…|8|9

改写G为G',使G'为与G等价的正规文法。 给出相应的有穷自动机。

**答：**令符号W表示单词，I表示标识符，D表示整数，a表示字母类的终结符，b表示数字类的终结符

则文法为：

W→I|D

I→Ia|Ib|a

D→Db|b

该文法为一左线性文法。构造自动机如下所示：



**用子集法将此NFA确定化的状态转换表如下：**

|  |  |  |  |
| --- | --- | --- | --- |
| 状态集 | 符号 |  |  |
|  |  |  |  |
|  | c |  |  |
|  |  |  |  |

**初态：**

终态：

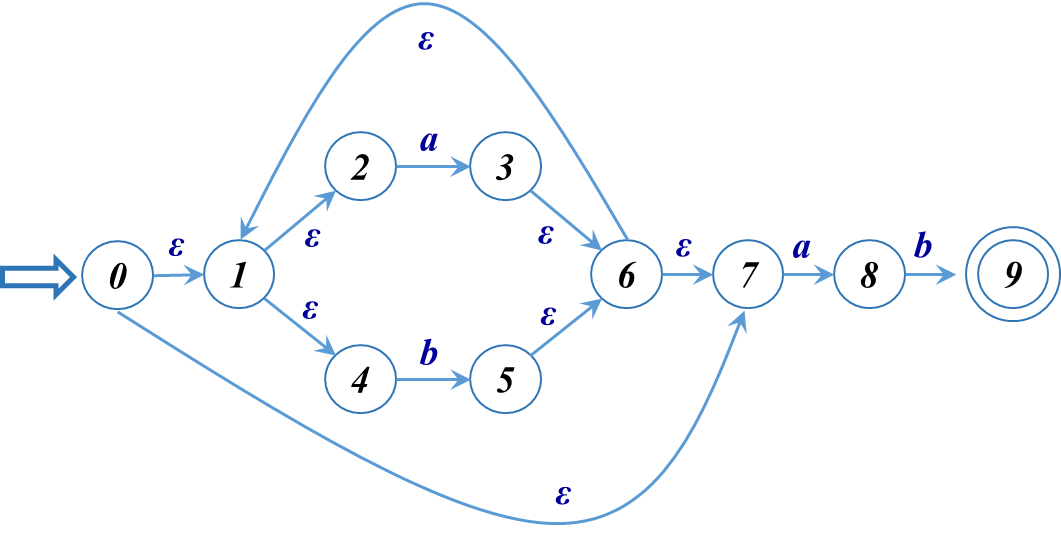
已经是最简化的DFA

**生成的如下图为：**



补充：将PPT中NFA的确定化

(1)



**答：用子集法将此NFA确定化的状态转换表如下：**

|  |  |  |  |
| --- | --- | --- | --- |
| 状态集 | 符号 |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**初态：**

终态：

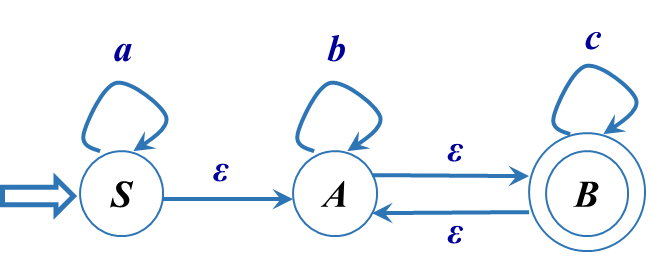
**分割法最小化：**

令

**生成的如下图为：**



(2)



**答：用子集法将此NFA确定化的状态转换表如下：**

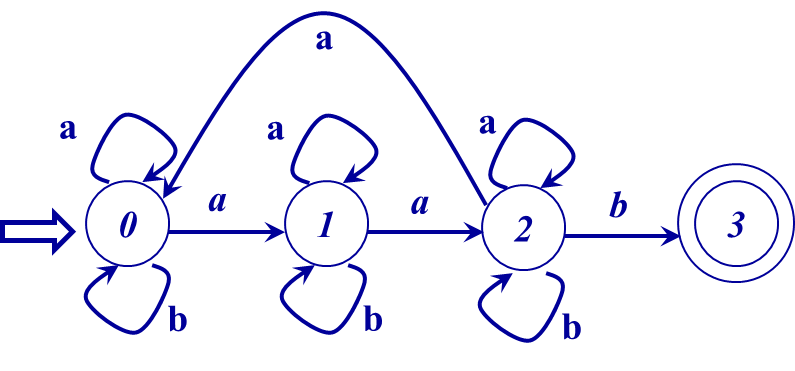
|  |  |  |  |
| --- | --- | --- | --- |
| 状态集 | 符号 |  |  |
|  | c |  |  |
|  | c |  |  |

**已经是状态数最小的DFA**

**生成的如下图为：**



(3)



**答：用子集法将此NFA确定化的状态转换表如下：**

|  |  |  |
| --- | --- | --- |
| 状态集 | 符号 |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**初态：**

终态：,

**分割法最小化：**

**DFA如下：**



# 第四章作业

**1.对文法G[S]**

**S→a||(T)**

**T→T,S|S**

**(1)给出(a,(a,a))和(((a,a),,(a)),a)的最左推导。**

**(2)对文法G进行改写，然后对每个非终结符写出不带回溯的递归子程序。**

**(3)经改写后的文法是否是LL(1)的？给出它的预测分析表。**

**(4)给出输入串(a,a)#的分析过程，并说明该串是否为G的句子。**

**答：**

**(1)(a,(a,a))和(((a,a),,(a)),a)的最左推导。**

**(((a,a),** **,(a)),a)的最左推导：**

S ( T )

( T , S )

( S , S )

( ( T ) , S )

( ( T , S ) , S )

( ( T , S , S ) , S )

( ( S , S , S ) , S )

( ( ( T ) , S , S ) , S )

( ( ( T, S ) , S , S ) , S )

( ( ( S , S ) , S , S ) , S )

( ( ( a , S ) , S , S ) , S )

( ( ( a , a ) , S , S ) , S )

( ( ( a , a ) , , S ) , S )

( ( ( a , a ) , , ( T ) ) , S )

( ( ( a , a ) , , ( S ) ) , S )

( ( ( a , a ) , , ( a ) ) , S )

( ( ( a , a ) , , ( a ) ) , a )

**(a,(a,a)) 的最左推导：**

S ( T )

( T , S )

( S , S )

( a , S )

( a , ( T ) )

( a , ( T , S ) )

( a , ( S , S ) )

( a , ( a , S ) )

( a , ( a , a ) )

**（2）消除文法的左递归：**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(char expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error \n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符T:T→ST’ 14. **void** ParseT() 15. { 16. **if** (lookahead == 'a' || lookahead == '⋀' || lookahead == '(') 17. ParseS(); 18. ParseT′(); 19. **else** { 20. printf("syntax error \n"); 21. exit(0); 22. } 23. } 24. // 解析非终结符T’:T’→,ST’| 25. **void** ParseT′() 26. { 27. **if** (lookahead == ',') 28. { 29. MatchToken(','); 30. ParseS(); 31. ParseT′(); 32. } 33. **else** **if** (lookahead == ')') 34. { 35. } 36. **else** { 37. printf("syntax error \n"); 38. exit(0); 39. } 40. } 41. // 解析非终结符S→a||(T) 42. **void** ParseS() 43. { 44. **switch** (lookahead) 45. { 46. **case** 'a': 47. MatchToken('a');; 48. **break**; 49. **case** '⋀': 50. MatchToken('⋀');; 51. **break**; 52. **case** '(': 53. MatchToken('('); 54. ParseT(); 55. MatchToken(')'); 56. **break**; 57. **default**: 58. printf("syntax error \n"); 59. exit(0); 60. } |

**(3)经改写后的文法是否是LL(1)的？给出它的预测分析表。**

**计算文法的每个非终结符的集和集**

|  |  |  |
| --- | --- | --- |
| **非终结符** | **集** | **集** |
|  |  |  |
|  |  |  |
|  |  |  |

**证明文法是的**

**因为**

**=**

**所以该文法是文法**

**(4)给出输入串(a,a)#的分析过程，并说明该串是否为G的句子。**

**构造它的预测分析表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **步骤** | **分析栈** | **输入缓冲区** | **选择的产生式或匹配** |
| **1** | **#S** | **(a,a)#** |  |
| **2** | **#)T(** | **(a,a)#** | **匹配 (** |
| **3** | **#)T** | **a,a)#** |  |
| **4** | **#)S** | **a,a)#** |  |
| **5** | **#) a** | **a,a)#** | **匹配 a** |
| **6** | **#)** | **,a)#** |  |
| **7** | **#) S,** | **,a)#** | **匹配 ,** |
| **8** | **#) S** | **a)#** |  |
| **9** | **#) a** | **a)#** | **匹配 a** |
| **10** | **#) T’** | **)#** |  |
| **11** | **#)** | **)#** | **匹配 )** |
| **12** | **#** | **#** | **成功** |

**2. 对下面的文法：**

1. **计算这个文法的每个非终结符的集和集**
2. **证明这个文法是的**
3. **构造它的预测分析表**
4. **构造它的递归下降分析程序**

**答：**

**(1) 计算这个文法的每个非终结符的集和集**

|  |  |  |
| --- | --- | --- |
| **非终结符** | **集** | **集** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**(2)证明这个文法是的**

**因为**

**所以该文法是文法**

**(3)构造它的预测分析表**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

**(4)构造它的递归下降分析程序**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(char expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error \n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符E 14. **void** ParseE() 15. { 16. **if** (lookahead == '(' || lookahead == 'a' || lookahead == 'b' || lookahead == '⋀') 17. { 18. ParseT(); 19. ParseE′(); 20. } 21. **else** 22. { 23. printf("syntax error \n"); 24. exit(0); 25. } 26. } 27. // 解析非终结符E′ 28. **void** ParseE′() 29. { 30. **if** (lookahead == '+') 31. { 32. MatchToken(')'); 33. ParseE(); 34. } 35. **else** **if** (lookahead == ')' || lookahead == '#') 36. { 37. } 38. **else** 39. { 40. printf("syntax error \n"); 41. exit(0); 42. } 43. } 44. // 解析非终结符T 45. **void** ParseT() 46. { 47. **switch** (lookahead) 48. { 49. **case** '(', 'a', 'b', ' ⋀': 50. ParseF(); 51. ParseT′(); 52. **break**; 53. **default**: 54. printf("syntax error \n"); 55. exit(0); 56. } 57. } 58. // 解析非终结符T′ 59. **void** ParseT′() 60. { 61. **switch** (lookahead) 62. { 63. **case** '(', 'a', 'b', ' ⋀': 64. ParseT(); 65. **break**; 66. **case** '+', ' )', '#': 67. **break**; 68. **default**: 69. printf("syntax error \n"); 70. exit(0); 71. } 72. } 73. // 解析非终结符F 74. **void** ParseF() 75. { 76. **switch** (lookahead) 77. { 78. **case** '(', 'a', 'b', ' ⋀': 79. ParseP(); 80. **break**; 81. **default**: 82. printf("syntax error \n"); 83. exit(0); 84. } 85. } 86. // 解析非终结符F′ 87. **void** ParseF′() 88. { 89. **switch** (lookahead) 90. { 91. **case** '\*': 92. MatchToken('\*'); 93. ParseF’(); 94. **break**; 95. **case** '+', '(', ')', 'a', 'b', ' ⋀', '#': 96. **break**; 97. **default**: 98. printf("syntax error \n"); 99. exit(0); 100. } 101. } 102. // 解析非终结符P 103. **void** ParseP() 104. { 105. **switch** (lookahead) 106. { 107. **case** '(': 108. MatchToken('('); 109. ParseE(); 110. MatchToken(')'); 111. **break**; 112. **case** 'a': 113. MatchToken('a'); 114. **break**; 115. **case** 'b': 116. MatchToken('b'); 117. **break**; 118. **case** '⋀': 119. MatchToken('⋀'); 120. **break**; 121. **default**: 122. printf("syntax error \n"); 123. exit(0); 124. } 125. } |

**3. 已知文法：**

**判断是否是文法，如果是，构造分析表。**

**答：非终结符对应的集和集**

|  |  |  |
| --- | --- | --- |
| **非终结符** | **集** | **集** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

***因为***

**所以文法是文法**

**对应预测分析表如下：**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

**4. 证明下述文法不是LL(1)的**

**能否构造一等价的文法，使其是LL(1)的？并给出判断过程。**

**答：**

**因为与拥有左公共因子，所以该文法不是文法。**

**因此，对这两个产生式 提取左公共因子，构造等价的文法如下：**

**证明该文法是LL(1)文法**

**首先给出该文法中非终结符的FIRST集与FOLLOW集**

|  |  |  |
| --- | --- | --- |
| **非终结符** | **FIRST集** | **FOLLOW集** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**因为**

**所以，无法构造一个等价文法使其是文法。**

**5.** 文法G如下：

**<程序>→**begin**<语句表>** end

**<语句表>→<语句>|<语句表>；<语句>**

**<语句>→<无条件语句>|<条件语句>**

**<无条件语句>→***a*

**<条件语句>→<如果语句>|<如果语句>else<语句>**

**<如果语句>→<如果子句><无条件语句>**

**<如果子句>→** if *b* then

试将G改写为LL(1)文法，并构造其预测分析表，判断改写后的文法是否为LL(1)文法。

**答：**

**将文法符号化：<程序>P，<语句表>L，<语句>S，<无条件语句>U，<条件语句>F，<如果语句>C, <如果子句> I**

**P→**begin **L** end

**L→S|L;S**

**S→U|F**

**U→***a*

**F→C|C else S**

**C→I U**

**I→** if *b* then

**对该文法消除左递归、提取左因子：**

**P→**begin **L** end

**L→SL’**

**L’→;SL’|**

**S→U|F**

**U→***a*

**F→CF’**

**F’→ | else S**

**C→I U**

**I→** if *b* then

|  |  |  |  |
| --- | --- | --- | --- |
| **文法产生式** | **First** | **Follow** | **Select** |
| **P→**begin **L** end  **L→SL’**  **L’→;SL’**    **S→U**  **F**  **U→***a*  **F→CF’**  **F’→**  **else S**  **C→I U**  **I→** if *b* then | **begin**  **a, if**  **;**  **a**  **if**  **a**  **if**  **else**  **if**  **if** | **#**  **end**  **end**  **; end**  **; end**  **; end**  **else ; end**  **a** | **begin**  **a, if**  **;**  **end**  **a**  **if**  **a**  **if**  **; end**  **else**  **if**  **if** |

**Select（L’→;SL’） Select（L’→）=**

**Select（F’→ else S） Select（F’→）=**

**所以该文法是LL（1）文法。**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **begin** | **end** | **if** | **then** | **else** | **a** | **;** | **#** |
| **P** | begin **L** end |  |  |  |  |  |  |  |
| **L** |  |  | **SL’** |  |  | **SL’** |  |  |
| **L’** |  |  |  |  |  |  | **;SL’** |  |
| **S** |  |  | **F** |  |  | **U** |  |  |
| **U** |  |  |  |  |  | **a** |  |  |
| **F** |  |  | **CF’** |  |  |  |  |  |
| **F’** |  |  |  |  | **else S** |  |  |  |
| **C** |  |  | **I U** |  |  |  |  |  |
| **I** |  |  | if *b* then |  |  |  |  |  |

**6. 判断下面哪些文法是LL(1)的，哪些能改写为LL(1)文法，并对每个LL(1)文法设计相应的递归下降识别器。**

**(2)**

**答：在该文法中产生式与产生式存在左公共因子，所以该文法不是文法。现提取左公共因子构造等价文法：**

**分析表如下：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**构造它的递归下降分析程序：**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(char expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error\n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符S 14. **void** ParseS() 15. { 16. **if** (lookahead == 'd' || lookahead == 'b' || lookahead == 'a' || lookahead == '#') 17. { 18. ParseB(); 19. ParseS′(); 20. } 21. **else** 22. { 23. printf("syntax error\n"); 24. exit(0); 25. } 26. } 27. // 解析非终结符S′ 28. **void** ParseS′() 29. { 30. **switch** (lookahead) 31. { 32. **case** 'a': 33. MatchToken('a'); 34. **break**; 35. **case** '#': 36. **break**; 37. **default**: 38. printf("syntax error\n"); 39. exit(0); 40. } 41. } 42. // 解析非终结符B 43. **void** ParseB() 44. { 45. **switch** (lookahead) 46. { 47. **case** 'a', 'd', 'b', '#': 48. ParseD(); 49. ParseB′(); 50. **break**; 51. **default**: 52. printf("syntax error\n"); 53. exit(0); 54. } 55. } 56. // 解析非终结符B′ 57. **void** ParseB′() 58. { 59. **switch** (lookahead) 60. { 61. **case** 'b': 62. MatchToken('b'); 63. **break**; 64. **case** '#', 'a': 65. **break**; 66. **default**: 67. printf("syntax error\n"); 68. exit(0); 69. } 70. } 71. // 解析非终结符D 72. **void** ParseD() 73. { 74. **switch** (lookahead) 75. { 76. **case** 'd': 77. MatchToken('d'); 78. **break**; 79. **case** 'b', 'a', '#': 80. **break**; 81. **default**: 82. printf("syntax error\n"); 83. exit(0); 84. } 85. } |

**(4)**

**答：在该文法中存在直接左递归，所以该文法不是文法。现消除左递归构造等价文法：**

**分析表如下：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**构造它的递归下降分析程序：**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(char expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error\n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符S 14. **void** ParseS() 15. { 16. **switch** (lookahead) 17. { 18. **case** 'i': 19. MatchToken('i'); 20. **break**; 21. **case** '(': 22. MatchToken('('); 23. ParseE(); 24. MatchToken(')'); 25. **break**; 26. **default**: 27. printf("syntax error\n"); 28. exit(0); 29. } 30. } 31. // 解析非终结符E 32. **void** ParseE() 33. { 34. **switch** (lookahead) 35. { 36. **case** 'i', '(': 37. ParseS(); 38. ParseE’(); 39. **break**; 40. **default**: 41. printf("syntax error\n"); 42. exit(0); 43. } 44. } 45. // 解析非终结符E′ 46. **void** ParseE′() 47. { 48. **switch** (lookahead) 49. { 50. **case** '+': 51. MatchToken('+'); 52. ParseS(); 53. ParseE′(); 54. **break**; 55. **case** '-': 56. MatchToken('-'); 57. ParseS(); 58. ParseE′(); 59. **break**; 60. **case** ')': 61. **break**; 62. **default**: 63. printf("syntax error\n"); 64. exit(0); 65. } 66. } |

**(6)**

**答：在该文法中产生式存在直接左递归，产生式存在左公共因子，所以该文法不是文法。现消除左递归构造等价文法：**

**分析表如下：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**构造它的递归下降分析程序：**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(**char** expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error\n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符M 14. **void** ParseM() 15. { 16. **switch** (lookahead) 17. { 18. **case** 'b', '(': 19. ParseH(); 20. ParseM′(); 21. **break**; 22. **default**: 23. printf("syntax error\n"); 24. exit(0); 25. } 26. } 27. // 解析非终结符M′ 28. **void** ParseM′() 29. { 30. **switch** (lookahead) 31. { 32. **case** 'a': 33. MatchToken('a'); 34. ParseH(); 35. ParseM′(); 36. **break**; 37. **case** ')','#': 38. **break**; 39. **default**: 40. printf("syntax error\n"); 41. exit(0); 42. } 43. } 44. // 解析非终结符H 45. **void** ParseH() 46. { 47. **switch** (lookahead) 48. { 49. **case** 'b': 50. MatchToken('b'); 51. ParseH’(); 52. **break**; 53. **case** '(': 54. MatchToken('('); 55. ParseM(); 56. MatchToken(')'); 57. **break**; 58. **default**: 59. printf("syntax error\n"); 60. exit(0); 61. } 62. } 63. // 解析非终结符H′ 64. **void** ParseH′() 65. { 66. **switch** (lookahead) 67. { 68. **case** '(': 69. MatchToken('('); 70. ParseM(); 71. MatchToken(')'); 72. **break**; 73. **case** ')', 'a', '#': 74. **break**; 75. **default**: 76. printf("syntax error\n"); 77. exit(0); 78. } 79. } |

**（1）S→ A | B**

**A→ aA | a**

**B→ bB | b**

**文法中存在左公因子A→ aA | a，所以不是LL（1）文法。**

对文法进行消除左公因子：

**S→ A | B**

**A→ aA’**

**A’→ A |** ε

**B→ bB’**

**B’→ B |** ε

**分析表如下：**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**构造它的递归下降分析程序：**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(**char** expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error\n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符S 14. **void** ParseS() 15. { 16. **switch** (lookahead) 17. { 18. **case** 'a':  ParseA(); **break**; 19. **case** 'b': ParseB(); **break**; 20. **default**:   printf("syntax error\n");  exit(0); 21. } 22. } 23. // 解析非终结符A 24. **void** ParseA() 25. { 26. **if** (lookahead == 'a') 27. { MatchToken('a'); 28. ParseA’(); 29. } 30. else 31. **{**   printf("syntax error\n"); 32. exit(0); 33. } 34. } 35. // 解析非终结符B 36. **void** ParseB() 37. { 38. **if** (lookahead == 'b') 39. { MatchToken('b'); 40. ParseB(); 41. } 42. else 43. **{**     printf("syntax error\n"); 44. exit(0); 45. } 46. } 47. **void** ParseA’()  // 解析非终结符A’ 48. { 49. **switch** (lookahead) 50. { 51. **case** 'a':   ParseA();   **break**; 52. **case** '#':    **break**; 53. **default**: 54. printf("syntax error\n"); 55. exit(0); 56. } 57. } 58. // 解析非终结符B′ 59. **void** ParseB′() 60. { 61. **switch** (lookahead) 62. { 63. **case** 'b':   ParseB();   **break**; 64. **case** '#':    **break**; 65. **default**: 66. printf("syntax error\n"); 67. exit(0); 68. } 69. } |

**（3）S→ aAaB | bAbB**

**A→ S | db**

**B→ bB | a**

是LL（1）文法

**分析表如下：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**构造它的递归下降分析程序：**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(**char** expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error\n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符S 14. **void** ParseS() 15. { 16. **switch** (lookahead) 17. { 18. **case** 'a': 19. // 20. MatchToken('a'); 21. ParseA(); 22. MatchToken('a'); 23. ParseB(); 24. **break**; 25. Case 'b': // 26. MatchToken('b'); 27. ParseA(); 28. MatchToken('b'); 29. ParseB(); 30. **break**; 31. **default**: 32. printf("syntax error\n"); 33. exit(0); 34. } 35. } 36. // 解析非终结符A 38. **void** ParseA() 39. { 40. **switch** (lookahead ) 41. **case** 'a':   ParseS();**break**; 42. **case** 'b':   ParseS(); **break**; 43. **case** 'd':   MatchToken('d'); MatchToken('b'); **break**; 44. **default**: 45. **{**   printf("syntax error\n"); 46. exit(0); 47. } 48. } 49. // 解析非终结符B 50. **void** ParseB() 51. { 52. **switch** (lookahead ) 53. { 54. **case** 'a':   MatchToken('a');**break**; 55. **case** 'b':   MatchToken('b'); ParseB(); **break**; 56. **default**: 57. **{**     printf("syntax error\n"); 58. exit(0); 59. } 60. } |

**（5）S→ SaA | bB**

**A→ aB | c**

**B→ Bb | d**

**文法中存在左递归，所以不是LL（1）文法。消除文法的左递归：**

**S→ bBS’**

**S’→ aAS’ |**ε

**A→ aB | c**

**B→ dB’**

**B’ →bB’ |**ε

**分析表如下：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**构造它的递归下降分析程序：**

|  |
| --- |
| **递归下降分析程序** |
| 1. **void** MatchToken(**char** expected) 2. { 3. **if** (lookahead != expected) 4. { 5. printf("syntax error\n"); 6. exit(0); 7. } 8. **else** 9. { 10. lookahead = getToken(); 11. } 12. } 13. // 解析非终结符S 14. **void** ParseS() 15. { 16. **if** (lookahead == 'b') 17. { MatchToken('b'); 18. ParseB(); 19. ParseS’();  } 20. else 21. **{**     printf("syntax error\n"); 22. exit(0); 23. } 24. } 25. // 解析非终结符S’ 26. **void** ParseS’() 27. { 28. **switch** (lookahead) 29. { 30. **case** 'a':   MatchToken('a'); ParseA();ParseS’();**break**; 31. **case** '#':     **break**; 32. **default**: 33. printf("syntax error\n"); 34. exit(0); 35. } 36. } 37. // 解析非终结符A 38. **void** ParseA() 39. { 40. **switch** (lookahead) 41. { 42. **case** 'a':   MatchToken('a'); ParseB();**break**; 43. **case** 'c':    MatchToken('c'); **break**; 44. **default**: 45. printf("syntax error\n"); 46. exit(0); 47. } 48. } 49. **void** ParseB()  // 解析非终结符B 50. { 51. **if** (lookahead=='d') 52. { 53. MatchToken('d'); 54. ParseB(); 55. } 56. **else** 57. { printf("syntax error\n"); 58. exit(0); 59. } 60. } 61. // 解析非终结符B′ 62. **void** ParseB′() 63. { 64. **switch** (lookahead) 65. { 66. **case** 'a', '#':  **break**; 67. **case** 'b':   MatchToken('b'); ParseB’();  **break**; 68. **default**: 69. printf("syntax error\n"); 70. exit(0); 71. } 72. } |

**7.对于一个文法若消除了左递归，提取了左公共因子后是否一定为LL(1)文法？试对下面的文法进行改写，并对改写后的文法进行判断。**

**答：不一定**

**（1）A→ baB | ε**

**B→ Abb | a**

**该文法无左递归，无左公因子**

**Select（A→ baB）={b}**

**Select（A→ε）=follow(A)={#,b}**

**两者有交集，因此不是LL（1）文法。**

**（2）A→ aABe | a**

**B→ Bb | d**

**对该文法消除左递归，提取左公因子：**

**A→ aA’**

**A’→ ABe | ε**

**B→ dB’**

**B’→ bB’ | ε**

**Select（A’→ABe）={a}**

**Select（A’→ε）=follow(A’)={#}**

**两者交集为空集**

**Select（B’→ bB’ ）={b}**

**Select（B’→ε）=follow(B’)={e}**

**两者交集为空集**

**该文法是LL（1）文法。**

**（3）S→ Aa | b**

**A→ SB**

**B→ ab**

**对该文法消除左递归，提取左公因子：**

**S→ bS’**

**S’→ Ba S’ | ε**

**B→ ab**

**Select（S’→ BaS’ ）={a}**

**Select（S’→ε）=follow(S’)={#}**

**两者交集为空集**

**该文法是LL（1）文法。**

# 第五章作业

**1.已知文法G[S]为： S→a|∧|(T) T→T,S|S**

**（1）计算G[S]的FIRSTVT和LASTVT。**

**（2）构造G[S]的算符优先关系表并说明G[S]是否为算符优先文法。**

**（3）计算G[S]的优先函数。**

**（4）给出输入串(a,a)#和(a,(a,a))#的算符优先分析过程。**

**答：**

**（1）计算G[S]的FIRSTVT和LASTVT。**

|  |  |  |
| --- | --- | --- |
|  | **FirstVT** | **LastVT** |
| **S** | **a ∧ (** | **a ∧ )** |
| **T** | **, a ∧ (** | **, a ∧ )** |

**（2）构造G[S]的算符优先关系表并说明G[S]是否为算符优先文法。**

**求文法中的各个优先关系：**

**: ()**

**: T, T) S# （LASTVT）**

**: (T ,S #S （FIRSTVT）**

**算符优先关系表:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **,** | **(** | **)** | **a** | **∧** | **#** |
| **,** |  |  |  |  |  |  |
| **(** |  |  |  |  |  |  |
| **)** |  |  |  |  |  |  |
| **a** |  |  |  |  |  |  |
| **∧** |  |  |  |  |  |  |
| **#** |  |  |  |  |  |  |

**优先关系表中无冲突，所以是算符优先文法。**

**（3）计算G[S]的优先函数。**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **,** | **(** | **)** | **a** | **∧** | **#** |
| **f** | **5** | **3** | **7** | **7** | **7** | **1** |
| **g** | **6** | **4** | **3** | **7** | **8** | **1** |

**（4）给出输入串(a,a)#和(a,(a,a))#的算符优先分析过程。**

**分析(a,(a,a))#**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **步骤** | **分析栈** | **优先关系** | **余留输入串** | **最左素短语** | **产生式** |
| **1** | **#** | **<** | **(a,(a,a))#** |  |  |
| **2** | **#(** | **<** | **a,(a,a))#** |  |  |
| **3** | **#(a** | **>** | **,(a,a))#** | **a** | **S→a** |
| **4** | **#(S** | **<** | **,(a,a))#** |  |  |
| **5** | **#(S,** | **<** | **(a,a))#** |  |  |
| **6** | **#(S,(** | **<** | **a,a))#** |  |  |
| **7** | **#(S,(a** | **>** | **,a))#** | **a** | **S→a** |
| **8** | **#(S,(S** | **<** | **,a))#** |  |  |
| **9** | **#(S,(S,** | **<** | **a))#** |  |  |
| **10** | **#(S,(S,a** | **>** | **))#** | **a** | **S→a** |
| **11** | **#(S,(S,S** | **>** | **))#** | **S,S** | **T→T,S** |
| **12** | **#(S,(T** | **=** | **))#** |  | **H→(S)** |
| **13** | **#(S,(T)** | **>** | **)#** | **(T)** | **S→(T)** |
| **14** | **#(S,S** | **>** | **)#** | **S,S** | **T→T,S** |
| **15** | **#(T** | **=** | **)#** |  |  |
| **16** | **#(T)** | **>** | **#** | **(T)** | **S→(T)** |
| **17** | **#S** |  | **#** | **分析成功** |  |

**分析(a,a)#**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **步骤** | **分析栈** | **优先关系** | **余留输入串** | **最左素短语** | **产生式** |
| **1** | **#** | **<** | **(a,a)#** |  |  |
| **2** | **#(** | **<** | **a,a)#** |  |  |
| **3** | **#(a** | **>** | **,a)#** | **a** | **S→a** |
| **4** | **#(S** | **<** | **,a)#** |  |  |
| **5** | **#(S,** | **<** | **a)#** |  |  |
| **6** | **#(S,a** | **>** | **)#** | **a** | **S→a** |
| **7** | **#(S,S** | **>** | **)#** | **S,S** | **T→T,S** |
| **8** | **#(T** | **=** | **)#** |  |  |
| **9** | **#(T)** | **>** | **#** | **(T)** | **S→(T)** |
| **10** | **#S** |  | **#** | **分析成功** |  |

**4. 已知文法G[S]为：**

**（1）构造G的算符优先关系表，并判定G[S]是否为算符优先文法。**

**（2）给出句型a(T+S);H;(S)的短语、句柄、素短语、最左素短语。**

**（3）给出a;(a+a)和(a+a)的分析过程，说明它们是否为G[S]的句子。**

**（4）给出（3）中输入串的最右推导，分别说明两个输入串是否为G[S]的句子。**

**（5）由（3）和（4）说明了算符优先分析的哪些缺点？**

**（6）算符优先分析过程和规范规约过程都是最右推导的逆过程吗？**

**答：**

**（1）构造G的算符优先关系表，并判定G[S]是否为算符优先文法。**

**解答：**

**求文法各个非终结符的FirstVT集合和LastVT集合：**

|  |  |  |
| --- | --- | --- |
|  | **FirstVT** | **LastVT** |
| **S** | **( a** | **) a** |
| **G** | **( a** | **) a** |
| **H** | **( a** | **) a** |
| **T** | **( a +** | **) a +** |

**求文法中的各个优先关系：**

**: ()**

**: S; G( T) S) T+ S#**

**: ;G (T (S +S #S**

**构造优先关系表如下：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **;** | **(** | **)** | **a** | **+** | **#** |
| **;** |  |  |  |  |  |  |
| **(** |  |  |  |  |  |  |
| **)** |  |  |  |  |  |  |
| **a** |  |  |  |  |  |  |
| **+** |  |  |  |  |  |  |
| **#** |  |  |  |  |  |  |

**（2）给出句型a(T+S);H;(S)的短语、句柄、素短语、最左素短语**

**短语：a，T+S, a(T+S), H, (S), a(T+S);H, a(T+S);H;(S)（7个）**

**句柄: a**

**素短语: a，T+S, (S) （3个）**

**最左素短语: a**

**语法树如下：**

****

**（3）分析a;(a+a)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **步骤** | **分析栈** | **优先关系** | **余留输入串** | **最左素短语** | **产生式** |
| **1** | **#** | **<** | **a;(a+a)#** |  |  |
| **2** | **#a** | **>** | **;(a+a)#** | **a** | **H→a** |
| **3** | **#H** | **<** | **;(a+a)#** |  |  |
| **4** | **#H;** | **<** | **(a+a)#** |  |  |
| **5** | **#H;(** | **<** | **a+a)#** |  |  |
| **6** | **# H;(a** | **>** | **+a)#** | **a** | **H→a** |
| **7** | **# H;(H** | **<** | **+a)#** |  |  |
| **8** | **# H;(H+** | **<** | **a)#** |  |  |
| **9** | **# H;(H+a** | **>** | **)#** | **a** | **H→a** |
| **10** | **# H;(H+H** | **>** | **)#** | **H+H** | **T→T+S** |
| **11** | **# H;(T** | **=** | **)#** |  |  |
| **12** | **# H;(T)** | **>** | **#** | **(T)** | **H→(S)** |
| **13** | **# H;H** | **>** | **#** | **H;H** | **S→S;G** |
| **14** | **#S** |  | **#** | **分析成功** |  |

**分析(a+a)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **步骤** | **分析栈** | **优先关系** | **余留输入串** | **最左素短语** | **产生式** |
| **1** | **#** | **<** | **(a+a)#** |  |  |
| **2** | **#(** | **>** | **a+a)#** | **a** | **H→a** |
| **3** | **#(a** | **<** | **+a)#** |  |  |
| **4** | **#(H** | **<** | **+a)#** |  |  |
| **5** | **#(H+(** | **<** | **a)#** |  |  |
| **6** | **# (H+a** | **>** | **)#** | **a** | **H→a** |
| **7** | **# (H+H** | **<** | **)#** | **H+H** | **T→T+S** |
| **8** | **# (T** | **<** | **)#** |  |  |
| **9** | **#(T)** | **>** | **#** | **（T）** | **H→(S)** |
| **10** | **# H** | **>** | **#** | **分析成功** |  |

**(4) a;(a+a)的最右推导**

**SS;G**

**S;H**

**S;(S)**

**S;(G)**

**S;(H) 出错**

**(a+a)的最右推导**

**SG**

**H**

**(S)**

**(G)**

**(H) 出错**

**（5）算法优先归约的是最左素短语，不是句柄，因此会隐藏语法错误无法发现。**

**（6）算法优先分析过程不是最右推导逆过程；**

**规范归约是最右推导逆过程。**

# 第六章作业

**1. 已知文法**

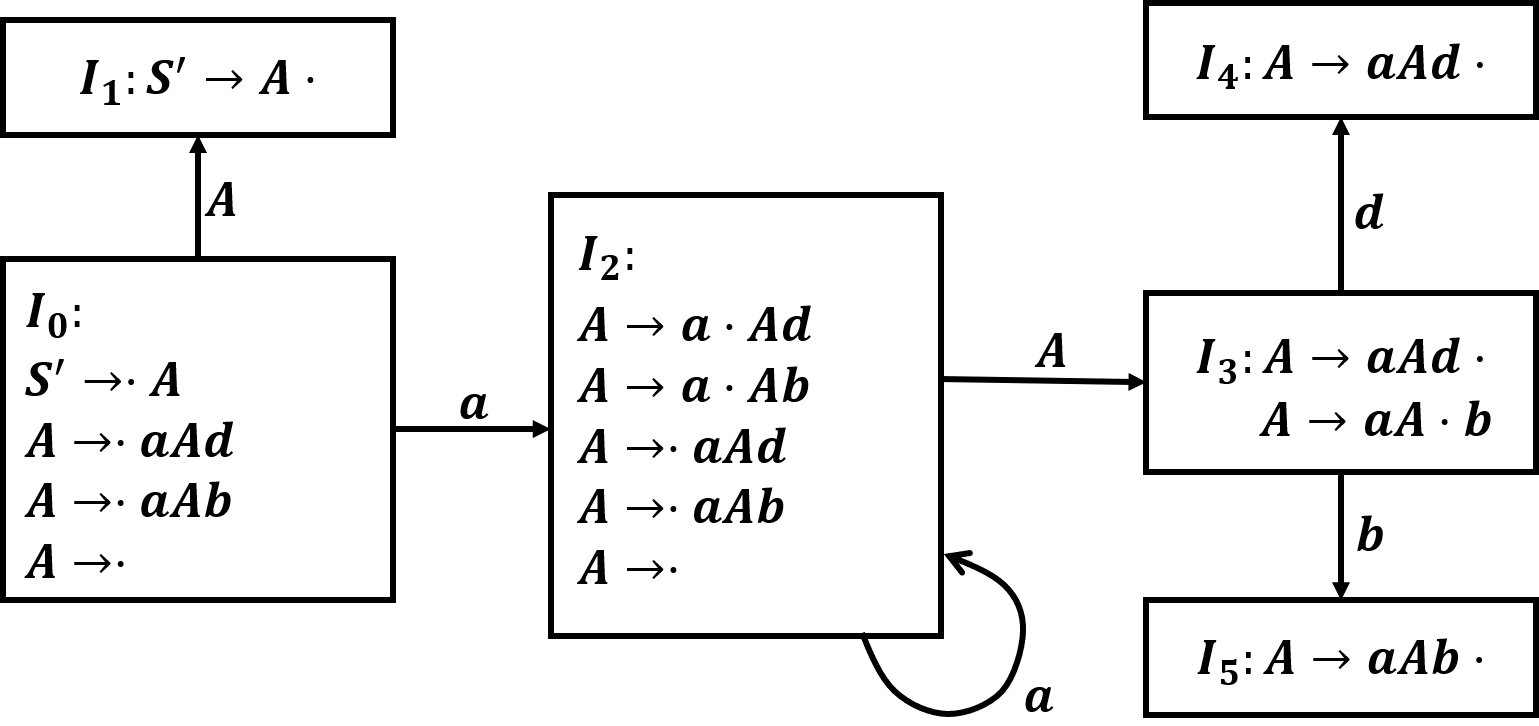
**判断该文法是否是文法，若是，请构造相应分析表，并对输入串给出分析过程。**

**答：拓广文法为，增加产生式，若产生式排序为:**

|  |  |
| --- | --- |
| ***0*** |  |
| ***1*** |  |
| ***2*** |  |
| ***3*** |  |

**由产生式知：**

**的项目集族及识别活前缀的如下图所示：**

****

**在中：和为移进项目，为归约项目，存在移进-归约冲突，因此所给文法不是文法。**

**在、中：所以在、中的移进-归约冲突可以由集解决，所以是文法。**

**构造的分析表如下：**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | **GOTO** |
|  |  |  |  |  |
| **0** |  |  |  |  | **1** |
| **1** |  |  |  |  |  |
| **2** |  |  |  |  | **3** |
| **3** |  |  |  |  |  |
| **4** |  |  |  |  |  |
| **5** |  |  |  |  |  |

**对输入串的分析过程**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **步骤** | **状态栈** | **符号栈** | **输入串** | **分析动作** | **下一状态** |
|  |  |  |  |  |  |
|  |  |  |  |  | **3** |
|  |  |  |  |  |  |
|  |  |  |  |  | **1** |
|  |  |  |  |  |  |

**分析成功，说明输入串是文法的句子。**

**2.若有定义二进制数的文法如下：**

**S→L.L | L**

**L→LB|B**

**B→0|1**

**(1) 试为该文法构造LR分析表，并说明属哪类LR分析表。**

**(2) 给出输入串101.110的分析过程。**

**说明：为了便于画图，用’+’代替’.’。**

**答：拓广文法并对产生式进行编号：**

**0: S’→S**

**1: S→L+L**

**2: S→L**

**3: L→LB**

**4: L→B**

**5: B→0**

**6: B→1**

**的项目集族及识别活前缀的如下图所示：**



**由产生式知：**

**在中：、、为移进项目，为归约项目，存在移进-归约冲突，因此所给文法不是文法。**

**在中：, 所以在中的移进-归约冲突可以由集解决。**

**在中：所以在中的移进-归约冲突可以由集解决**

**综上，该文法是文法。**

**构造的分析表如下：**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | **GOTO** | | |
|  |  |  |  |  |  |  |
| **0** |  |  |  |  | **1** | **2** | **3** |
| **1** |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  | **7** |
| **3** |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  | **8** | **3** |
| **7** |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  | **7** |

**对输入串101.110（101+110）的分析过程**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **步骤** | **状态栈** | **符号栈** | **输入串** | **分析动作** | **下一状态** |
|  |  |  |  |  | **5** |
|  |  |  |  |  | **GOTO[0][B]=3** |
|  | **3** |  |  |  | **GOTO[0][L]=2** |
|  | **2** |  |  |  | **4** |
|  | **4** |  |  |  | **GOTO[2][B]=7** |
|  | **7** |  |  |  | **GOTO[0][L]=2** |
|  |  |  |  |  | **5** |
|  | **5** | **1** |  |  | **GOTO[2][B]=7** |
|  | **7** | **B** |  |  | **GOTO[0][L]=2** |
|  |  |  |  |  | **6** |
|  | **6** | **+** |  |  | **5** |
|  | **65** | **+1** |  |  | **GOTO[6][B]=3** |
|  | **63** |  |  |  | **GOTO[6][L]=8** |
|  | **68** | **+L** |  |  | **5** |
|  |  | **+L1** |  |  | **GOTO[8][B]=7** |
|  | **687** | **+LB** |  |  | **GOTO[6][L]=8** |
|  | **68** | **+L** |  |  | **4** |
|  | **684** | **+L0** |  |  | **GOTO[8][B]=7** |
|  | **687** | **+LB** |  |  | **GOTO[6][L]=8** |
|  | **68** | **+L** |  |  | **GOTO[0][S]=1** |
|  |  |  |  |  |  |

**分析成功，说明输入串101.110（101+110）是文法的句子。**

**3.考虑文法**

**S→AS|b**

**A→SA|a**

**(1) 列岀这个文法的所有LR(0)项目。**

**(2) 按(1)列出的项目构造识别这个文法活前缀的NFA,把这个NFA确定化为DFA, 说明这个DFA的所有状态全体构成这个文法的LR(0)规范族。**

**(3) 这个文法是SLR(l)的吗？若是，构造出它的SLR分析表。**

**(4) 这个文法是LALR(l)或LR(1)的吗？**

**答：**

**(1) 列岀这个文法的所有LR(0)项目。**

1. **S’ →S**
2. **S→AS**
3. **S→b**
4. **A→SA**
5. **A→a**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **(1) S’→·S** | **(3) S→·AS** | **(6) S→·b** | **(8) A→·SA** | **(11) A→·a** |
| **(2) S’→ S·** | **(4) S→A·S** | **(7) S→b·** | **(9) A→S·A** | **(12) A→a·** |
|  | **(5) S→AS·** |  | **(10) A→SA·** |  |

**（2）按(1)列出的项目构造识别这个文法活前缀的NFA，并把这个NFA确定化为DFA：**



|  |  |
| --- | --- |
| 状态 |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**用有效子集法将此NFA确定化的状态转换表如下：**

|  |  |  |  |
| --- | --- | --- | --- |
| 状态集 | 符号 |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**把每个子集写出来，得到如下图所示的DFA**



**(3) 这个文法不是SLR(l)的，分析表中有冲突。**

**由产生式知：**

**构造的分析表如下：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | **GOTO** | | |
|  |  |  |  |  |  |
| **0** |  |  |  | **2** | **1** |  |
| **1** |  |  |  | **3** | **1** |  |
| **2** |  |  |  | **4** | **5** |  |
| **3** |  |  |  | **4** | **5** |  |
| **4** |  |  |  | **4** | **5** |  |
| **5** |  |  |  | **4** | **1** |  |
| **6** |  |  |  |  | **8** |  |
| **7** |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**这个文法不是SLR(l)的，分析表中有冲突。**

**(4) 在构造LR(1)的项目集规范族时，在项目集T5中存在移进-归约冲突，归约符号为{a,b},移进的符号也有a和b，因此冲突无法解决。故文法不是LR（1）的，也不是LALR（1）的。**



**4.下面是一个描述∑={a.b}上的正规式的LALR(l)文法(实际上也是SLR(l)文法)，只不过用+代替丨。**

**E→E+T | T**

**T→TF | F**

**F→ F\* |(E) | a |b 构造这个文法的LALR(l)项目集和分析表。**

**答：文法并对产生式进行编号：**

1. **E’→E**
2. **E→E+T**
3. **E→T**
4. **T→TF**
5. **T→F**
6. **F→F\***
7. **F→(E)**
8. **F→a**
9. **F→b**

**为便于计算，先计算项目集I0如下：**

**I0:**

**E’→·E, #**

**E→·E+T, #/+**

**E→·T, #/+**

**T→·TF, #/+/(/a/b**

**T→·F, #/+/(/a/b**

**F→·F\*,#/+/(/a/b/\***

**F→·(E), #/+/(/a/b/\***

**F→·a, #/+/(/a/b/\***

**F→·b, #/+/(/a/b/\***

**构造的LR（1）项目集规范族如下：**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **E** | **T** | **F** | **+** | **\*** | **(** | **)** | **a** | **b** |
| **I0:** | **I1:**  **E’→ E·, #**  **E→E·+T, #/+** | **I2:**  **E→T·, #/+**  **T→T·F, #/+/(/a/b**  **F→·F\*,#/+/(/a/b/\***  **F→·(E), #/+/(/a/b/\***  **F→·a, #/+/(/a/b/\***  **F→·b, #/+/(/a/b/\*** | **I3:**  **T→F·, #/+/(/a/b**  **F→F·\*,#/+/(/a/b/\*** |  |  | **I8:**  **F→(·E), #/+/(/a/b/\***  **E→·E+T, )/+**  **E→·T,)/+**  **T→·TF, )/+/(/a/b**  **T→·F, )/+/(/a/b**  **F→·F\*,)/+/(/a/b/\***  **F→·(E), )/+/(/a/b/\***  **F→·a, )/+/(/a/b/\***  **F→·b, )/+/(/a/b/\*** |  | **I4:**  **F→a·, #/+/(/a/b/\*** | **I5:**  **F→b·, #/+/(/a/b/\*** |
| **I1:** |  |  |  | **I6:**  **E→E+·T, #/+**  **T→·TF, #/+/(/a/b**  **T→·F, #/+/(/a/b**  **F→·F\*,#/+/(/a/b/\***  **F→·(E), #/+/(/a/b/\***  **F→·a, #/+/(/a/b/\***  **F→·b, #/+/(/a/b/\*** |  |  |  |  |  |
| **I2** |  |  | **I7:**  **T→TF·, #/+/(/a/b**  **F→F·\*,#/+/(/a/b/\*** |  |  | **I8:** |  | **I9:**  **F→a·, )/+/(/a/b/\*** | **I10:**  **F→b·, )/+/(/a/b/\*** |
| **I3** |  |  |  |  | **I11:**  **F→F\*·,#/+/(/a/b/\*** |  |  |  |  |
| **I4** |  |  |  |  |  |  |  |  |  |
| **I5** |  |  |  |  |  |  |  |  |  |
| **I6** |  | **I12:**  **E→E+T·, #/+**  **T→T·F, #/+/(/a/b**  **F→·F\*,#/+/(/a/b/\***  **F→·(E), #/+/(/a/b/\***  **F→·a, #/+/(/a/b/\***  **F→·b, #/+/(/a/b/\*** | **I7** |  |  | **I8** |  | **I4:** | **I5:** |
| **I7** |  |  |  |  | **I11:** |  |  |  |  |
| **I8** | **I13:**  **F→(E·), #/+/(/a/b/\***  **E→E·+T, )/+** | **I14:**  **E→T·,)/+**  **T→T·F, )/+/(/a/b**  **F→·F\*,)/+/(/a/b/\***  **F→·(E), )/+/(/a/b/\***  **F→·a, )/+/(/a/b/\***  **F→·b, )/+/(/a/b/\*** | **I15:**  **T→F·, )/+/(/a/b**  **F→F·\*,)/+/(/a/b/\*** |  |  |  |  | **I9:** | **I10:** |
| **I9** |  |  |  |  |  |  |  |  |  |
| **I10** |  |  |  |  |  |  |  |  |  |
| **I11** |  |  |  |  |  |  |  |  |  |
| **I12** |  |  | **I7** |  |  | **I8:** |  | **I4:** | **I5:** |
| **I13** |  |  |  | **I16:**  **E→E+·T, )/+**  **T→·TF, )/+/(/a/b**  **T→·F, )/+/(/a/b**  **F→·F\*,)/+/(/a/b/\***  **F→·(E), )/+/(/a/b/\***  **F→·a, )/+/(/a/b/\***  **F→·b, )/+/(/a/b/\*** |  |  | **I17;**  **F→(E)·, #/+/(/a/b/\*** |  |  |
| **I14** |  |  | **I15:** |  |  | **I18:**  **F→(·E), )/+/(/a/b/\***  **E→·E+T, )/+**  **E→·T,)/+/(/a/b**  **T→·TF, )/+/(/a/b**  **T→·F, )/+/(/a/b**  **F→·F\*,)/+/(/a/b/\***  **F→·(E), )/+/(/a/b/\***  **F→·a, )/+/(/a/b/\***  **F→·b, )/+/(/a/b/\*** |  | **I9:** | **I10:** |
| **I15** |  |  |  |  | **I19:**  **F→F\*·,)/+/(/a/b/\*** |  |  |  |  |
| **I16** |  | **I20:**  **E→E+T·, )/+**  **T→T·F, )/+/(/a/b**  **F→·F\*,)/+/(/a/b/\***  **F→·(E), )/+/(/a/b/\***  **F→·a, )/+/(/a/b/\***  **F→·b, )/+/(/a/b/\*** | **I15** |  |  | **I18** |  | **I9:** | **I10:** |
| **I17** |  |  |  |  |  |  |  |  |  |
| **I18** | **I21:**  **F→(E·), )/+/(/a/b/\***  **E→E·+T, )/+** | **I14** | **I15** |  |  | **I18** |  | **I9:** | **I10:** |
| **I19** |  |  |  |  |  |  |  |  |  |
| **I20** |  |  | **I22:**  **T→TF·, )/+/(/a/b**  **F→F·\*,)/+/(/a/b/\*** |  |  | **I18** |  | **I9:** | **I10:** |
| **I21** |  |  |  | **I16:** |  |  | **I23:**  **F→(E)·, )/+/(/a/b/\*** |  |  |
| **I22** |  |  |  |  | **I19** |  |  |  |  |
| **I23** |  |  |  |  |  |  |  |  |  |

**构造的LALR（1）项目集如下：**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **E** | **T** | **F** | **+** | **\*** | **(** | **)** | **a** | **b** |
| **I0:** | **I1:**  **E’→ E·, #**  **E→E·+T, #/+** | **I2**  **E→T·, )/#/+**  **T→T·F, )/#/+/(/a/b**  **F→·F\*,)/#/+/(/a/b/\***  **F→·(E), )/#/+/(/a/b/\***  **F→·a, )/#/+/(/a/b/\***  **F→·b, )/ #/+/(/a/b/\*** | **I3**  **T→F·, )/#/+/(/a/b**  **F→F·\*,)/#/+/(/a/b/\*** |  |  | **I8**  **F→(·E), )/#/+/(/a/b/\***  **E→·E+T, )/+**  **E→·T,)/+/(/a/b**  **T→·TF, )/+/(/a/b**  **T→·F, )/+/(/a/b**  **F→·F\*,)/+/(/a/b/\***  **F→·(E), )/+/(/a/b/\***  **F→·a, )/+/(/a/b/\***  **F→·b, )/+/(/a/b/\*** |  | **I4:**  **F→a·, )/#/+/(/a/b/\*** | **I5:**  **F→b·, )/#/+/(/a/b/\*** |
| **I1:** |  |  |  | **I6**  **E→E+·T, )/#/+**  **T→·TF, )/#/+/(/a/b**  **T→·F, )/#/+/(/a/b**  **F→·F\*, )/#/+/(/a/b/\***  **F→·(E), )/#/+/(/a/b/\***  **F→·a, )/#/+/(/a/b/\***  **F→·b, )/#/+/(/a/b/\*** |  |  |  |  |  |
| **I2** |  |  | **I7**  **T→TF·, )/#/+/(/a/b**  **F→F·\*, )/#/+/(/a/b/\*** |  |  | **I8** |  | **I4:** | **I5:** |
| **I3** |  |  |  |  | **I9**  **F→F\*·,)/#/+/(/a/b/\*** |  |  |  |  |
| **I4** |  |  |  |  |  |  |  |  |  |
| **I5** |  |  |  |  |  |  |  |  |  |
| **I6** |  | **I10**  **E→E+T·, )/#/+**  **T→T·F, )/#/+/(/a/b**  **F→·F\*,)/#/+/(/a/b/\***  **F→·(E), )/#/+/(/a/b/\***  **F→·a, )/#/+/(/a/b/\***  **F→·b, )/#/+/(/a/b/\*** | **I3** |  |  | **I8** |  | **I4:** | **I5:** |
| **I7** |  |  |  |  | **I9:** |  |  |  |  |
| **I8** | **I11:**  **F→(E·), )/#/+/(/a/b/\***  **E→E·+T, )/+** | **I2** | **I3** |  |  | **I8** |  | **I4:** | **I5:** |
| **I9** |  |  |  |  |  |  |  |  |  |
| **I10** |  |  | **I3** |  |  | **I8** |  | **I4:** | **I5:** |
| **I11** |  |  |  |  |  |  | **I12**  **F→(E)·,)/#/+/(/a/b/\*** |  |  |
| **I12** |  |  |  |  |  |  |  |  |  |

1. **E’→E**
2. **E→E+T**
3. **E→T**
4. **T→TF**
5. **T→F**
6. **F→F\***
7. **F→(E)**
8. **F→a**
9. **F→b**

**构造的分析表如下：**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** |  | | **ACTION** | | | | | | | **GOTO** | | | |
|  | **\*** | |  |  | **a** | **b** |  |  | |  | **F** |
| **0** |  |  | |  |  |  |  |  | **1** | | **2** | **3** |
| **1** |  |  | |  |  |  |  |  |  | |  |  |
| **2** |  |  | |  |  |  |  |  |  | |  | **7** |
| **3** |  |  | |  |  |  |  |  |  | |  |  |
| **4** |  |  | |  |  |  |  |  |  | |  |  |
| **5** |  |  | |  |  |  |  |  |  | |  |  |
| **6** |  |  | |  |  |  |  |  |  | | **10** | **3** |
| **7** |  |  | |  |  |  |  |  |  | |  |  |
| **8** |  |  | |  |  |  |  |  | **11** | | **2** | **3** |
| **9** |  |  | |  |  |  |  |  |  | |  |  |
| **10** |  |  | |  |  |  |  |  |  | |  | **3** |
| **11** |  |  | |  | **S12** |  |  |  |  | |  |  |
| **12** |  |  | |  |  |  |  |  |  | |  |  |

**6.文法 G=({U,T,S},{a0,c,d,e},P,S),其中 P 为**

**S→UTa|Tb**

**T→S | Sc | d**

**U→US | e**

**(1) 判断G是 LR(0)、SLR(1)、LALR(1)还是 LR(1)的，说明理由。**

**(2) 构造相应的分析表**

**答：**

**拓广文法并对产生式进行编号：**

**0: S’→S**

**1: S→UTa**

**2: S→Tb**

**3: T→S**

**4: T→Sc**

**5: T→d**

**6: U→US**

**7: U→e**

**的项目集族及识别活前缀的如下图所示：**



**在中：和为归约项目, 为移进项目，存在移进-归约冲突，因此所给文法不是文法。**

**由产生式知：**

**在中：, 所以在中的移进-归约冲突可以由集解决。**

**在中：,所以在中的移进-归约冲突可以由集解决**

**综上，该文法是文法。**

**0: S’→S**

**1: S→UTa**

**2: S→Tb**

**3: T→S**

**4: T→Sc**

**5: T→d**

**6: U→US**

**7: U→e**

**构造的分析表如下：**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | | | **GOTO** | | |
|  |  |  |  |  |  |  |  |  |
| **0** |  |  |  |  |  |  | **1** | **3** | **6** |
| **1** |  |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |  |  | **7** |
| **3** |  |  |  |  |  |  | **10** | **3** | **8** |
| **4** |  |  |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  |  | **8** | **3** |
| **7** |  |  |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  |  | **7** |
| **9** |  |  |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |  |  |

**7.证明下面文法不是LR(0)而是SLR(1)**

**S→A**

**A→Ab | bBa**

**B→aAc | a | aAb**

**的项目集族及识别活前缀的如下图所示：**



**由产生式知：**

**在和中存在移进-归约冲突，因此所给文法不是文法。**

**在中：和为归约-移进冲突，，所以在中的移进-归约冲突可以由规则解决。**

**在中：为归约项目，在中的移进-归约冲突可以由规则解决。**

**综上，该文法是文法。**

**8.证明文法(其中$相当于井)**

**S→A$**

**A→BaBb | DbDa**

**B→ɛ**

**D→ɛ**

**是LR(1)而不是SLR(1)的。**

**答：**

**拓广文法并对产生式进行编号：**

**0: S’→S**

**1:S→A**

**2:A→BaBb**

**3:A→DbDa**

**4:B→ɛ**

**5:D→ɛ**

**的项目集族及识别活前缀的状态I0如下图所示：**



**在中：和为归约-归约冲突，，所以在中的移进-归约冲突不能由规则解决，不是SLR(1)文法。**

**的项目集族及识别活前缀的状态如下图所示：**



**0: S’→S 1:S→A 2:A→BaBb 3:A→DbDa 4:B→ɛ 5:D→ɛ**

**构造的分析表如下：**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | **GOTO** | | | |
|  |  |  |  |  |  |  |
| **0** |  |  |  | **1** | **2** | **3** | **4** |
| **1** |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  | **7** |  |
| **3** |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  | **7** |  |
| **6** |  |  |  |  |  |  | **9** |
| **7** |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  | **7** |  |
| **9** |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |

**9.证明下面文法是LR(1)而不是LALR(l)的。**

**S→Aa | bAe |Be | bBa A→d B→d**

**答：拓广文法并对产生式进行编号：**

**0: S’→S 1：S→Aa 2：S→bAe 3：S→Be 4：S→bBa 5：A→d 6：B→d**



**0: S’→S 1：S→Aa 2：S→bAe 3：S→Be 4：S→bBa 5：A→d 6：B→d**

**构造的分析表如下：**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | | **GOTO** | | | |
|  |  | **d** | **e** |  |  |  |  |  |
| **0** |  |  |  |  |  | **1** | **2** | **4** |  |
| **1** |  |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  | **8** | **10** |  |
| **12,7** |  |  |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  | **7** |  |
| **9** |  |  |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |  |  |
| **11** |  |  |  |  |  |  |  |  |  |

**在中和在中，合并同心集之后，分析表有冲突，所以不是LALR(l)的。**

**0: S’→S 1：S→Aa 2：S→bAe 3：S→Be 4：S→bBa 5：A→d 6：B→d**

**构造的分析表如下：**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | | **GOTO** | | | |
|  |  | **d** | **e** |  |  |  |  |  |
| **0** |  |  |  |  |  | **1** | **2** | **4** |  |
| **1** |  |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  | **8** | **10** |  |
| **7** |  |  |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  | **7** |  |
| **9** |  |  |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |  |  |
| **11** |  |  |  |  |  |  |  |  |  |
| **12** |  |  |  |  |  |  |  |  |  |

**LR(1)分析表无冲突，所以是LR(1)的。**

**10. 判断下列6个文法是否为类文法，若是，请说明是、、或的哪一种，并构造相应的分析表；若不是，请说明理由。**

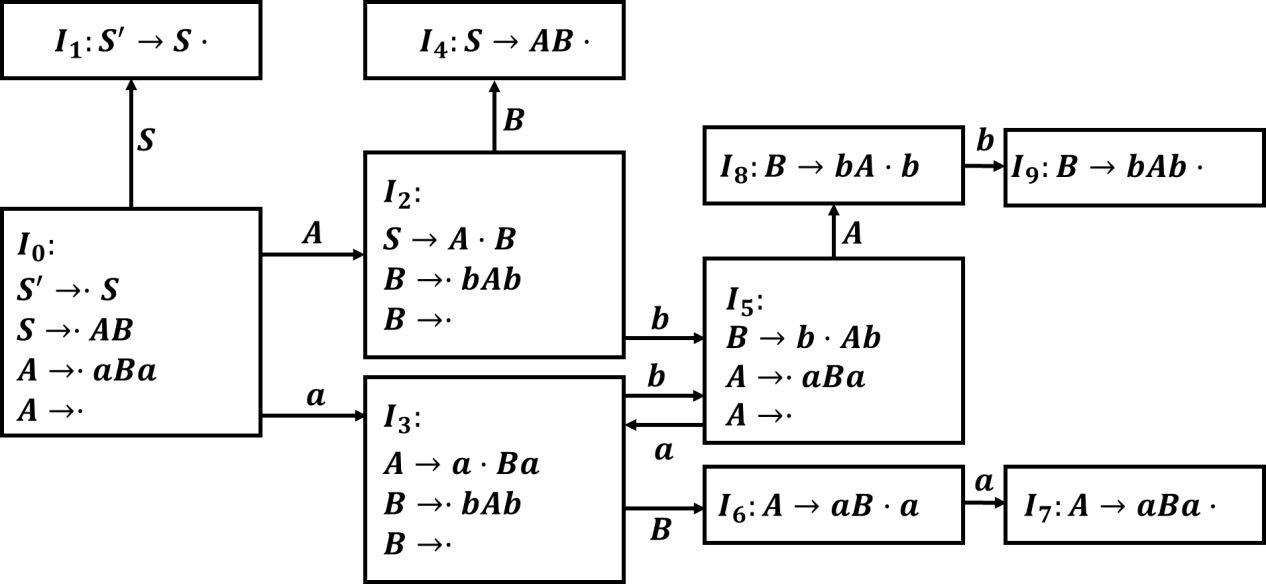
**(1)**

**答：拓广文法为，增加产生式，若产生式排序为：**

|  |  |
| --- | --- |
| ***0*** |  |
| ***1*** |  |
| ***2*** |  |
| ***3*** |  |
| ***4*** |  |
| ***5*** |  |

**由产生式知：**

**的项目集族及识别活前缀的如下图所示：**

****

**在:为移约项目，为归约项目，存在移进-归约冲突，因此所给文法不是文法。用文法能解决冲突，所以构造。**

**构造的分析表如下：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | **GOTO** | | |
|  |  |  |  |  |  |
| **0** |  |  |  | **1** | **2** |  |
| **1** |  |  |  |  |  |  |
| **2** |  |  |  |  |  | **4** |
| **3** |  |  |  |  |  | **6** |
| **4** |  |  |  |  |  |  |
| **5** |  |  |  |  | **8** |  |
| **6** |  |  |  |  |  |  |
| **7** |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |
| **9** |  |  |  |  |  |  |

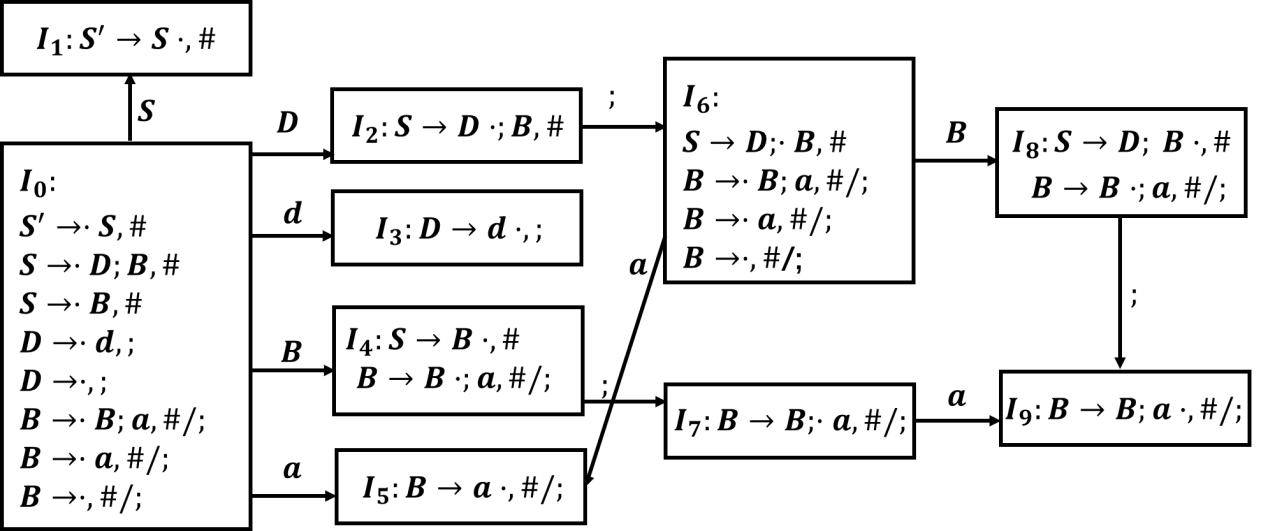
**(2)**

**答：拓广文法为，增加产生式，若产生式排序为：**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***0*** |  | | ***1*** |  | | ***2*** |  | | ***3*** |  | | ***4*** |  | | ***5*** |  | | ***6*** |  | | ***7*** |  | |  |

**的项目集族和项目集族有归约-归约冲突。**

**的项目集族及识别活前缀的如下图所示：**

****

**在:存在归约-归约冲突，所以不是。**

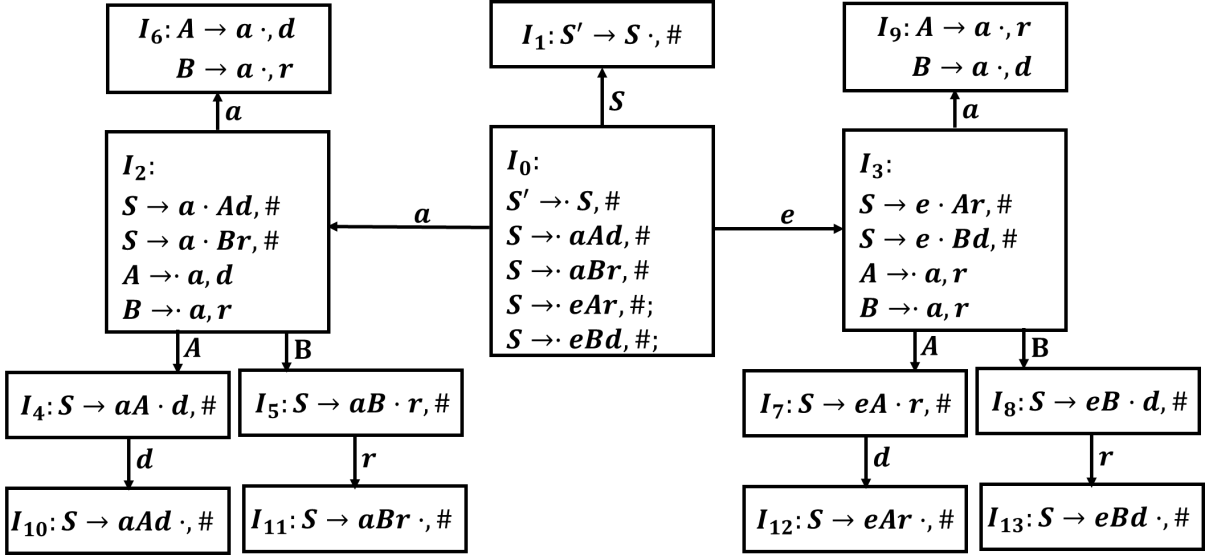
**(3)**

**答：拓广文法为，增加产生式，若产生式排序为：**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***0*** |  | | ***1*** |  | | ***2*** |  | | ***3*** |  | | ***4*** |  | | ***5*** |  | | ***6*** |  | |  |

**的项目集族和项目集族有归约-归约冲突。**

**的项目集族及识别活前缀的如下图所示：**

****

**构造无冲突。而同心集和无法合并，所以是，而不是是**

**构造的分析表如下：**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | | **GOTO** | | |
|  |  |  |  |  |  |  |  |
| **0** |  |  |  |  |  |  |  | **1** |
| **1** |  |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  | **4** | **5** |  |
| **3** |  |  |  |  |  | **7** | **8** |  |
| **4** |  |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  |  |  |
| **7** |  |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  |  |
| **9** |  |  |  |  |  |  |  |  |
| **10** |  |  |  |  |  |  |  |  |
| **11** |  |  |  |  |  |  |  |  |
| **12** |  |  |  |  |  |  |  |  |
| **13** |  |  |  |  |  |  |  |  |

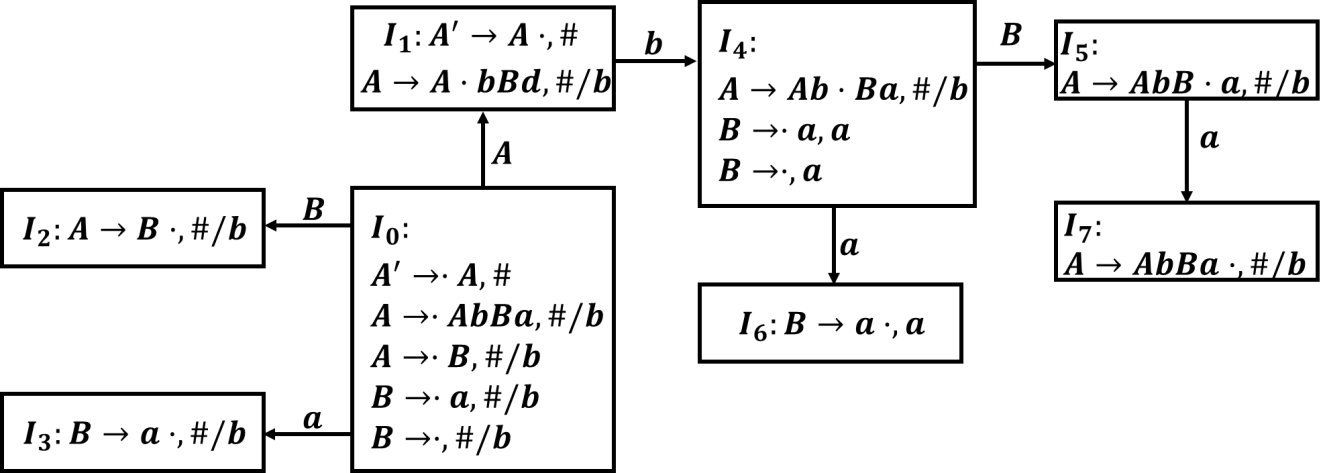
**(4)**

**答：拓广文法为，增加产生式，若产生式排序为：**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***0*** |  | | ***1*** |  | | ***2*** |  | | ***3*** |  | | ***4*** |  | |  |

**的项目集族和项目集族有移进-归约冲突。**

**的项目集族及识别活前缀的如下图所示：**

****

**在中，和有移进-归约冲突，所以也不是文法。**

**16.给定文法：**

1. **构造识别该文法活前缀的**
2. **该文法是的吗？是的吗？说明理由**
3. **若对一些终结符的优先级以及算符的结合规则规定如下：**
   1. **优先性大于**
   2. **服从左结合**
   3. **优先性大于**
   4. **优先性大于**

**请构造该文法的分析表。**

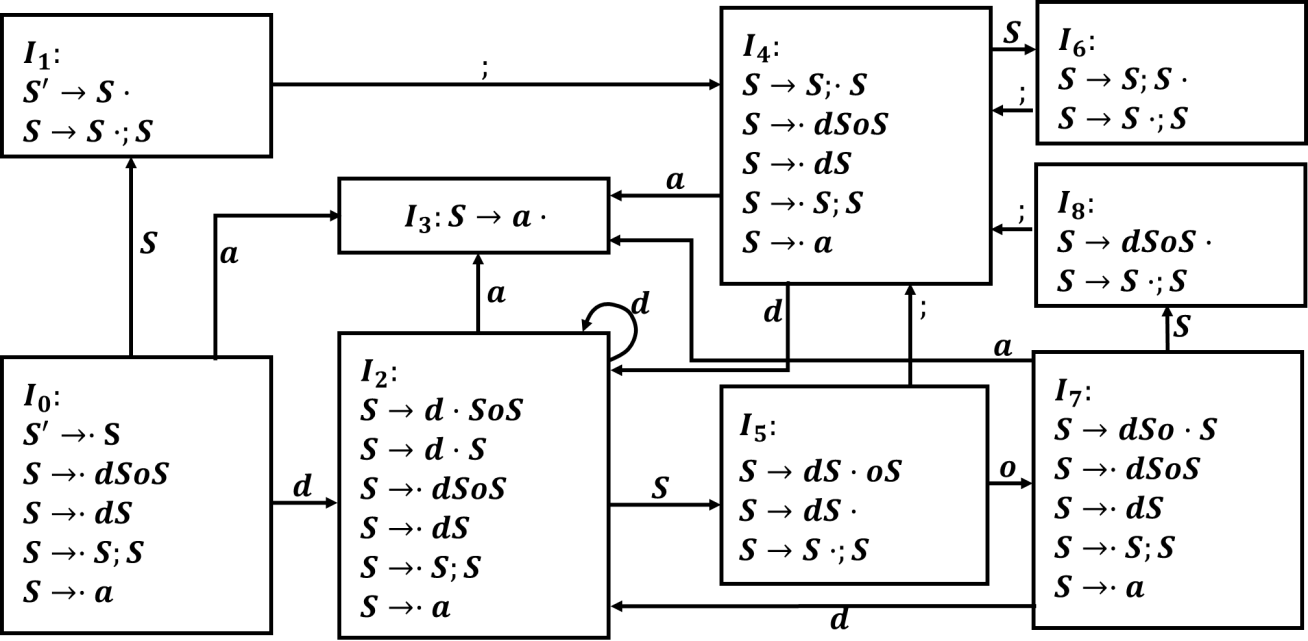
**答：首先化简文法，用代替；用代替；用代替；文法可写成：**

**拓广文法为，增加产生式，若产生式排序为：**

|  |  |
| --- | --- |
| ***0*** |  |
| ***1*** |  |
| ***2*** |  |
| ***3*** |  |
| ***4*** |  |

**由产生式知：**

1. **识别该文法活前缀的如下图：**

****

1. **该文法不是也不是因为：在、和存在移进-归约冲突，因此所给文法不是文法。**

**又由于在和中：**

**，在中：**

**所以该文法也不是*文法。***

**此外很容易证明所给文法是二义性的。**

1. **在中：和优先性都大于 ，所以遇输入符和移进；遇号归约。**

**在中：号服从左结合，所以遇输入符的都应该归约。**

**在中：号优先性大于do，所以遇输入符号移进；遇和号归约。**

**此外，在中：接受和移进可以不看成冲突，因此接受只有遇号。**

**由以上分析，所有存在的移进-归约冲突可用规定的终结符优先级以及算符的结合规则解决，所构造的分析表如下：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | | **GOTO** |
|  |  |  |  |  |  |
| **0** |  |  |  |  |  | **1** |
| **1** |  |  |  |  |  |  |
| **2** |  |  |  |  |  | **5** |
| **3** |  |  |  |  |  |  |
| **4** |  |  |  |  |  | **6** |
| **5** |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  |
| **7** |  |  |  |  |  | **8** |
| **8** |  |  |  |  |  |  |

# 第七章作业

**1. 下面的文法描述由布尔常量,,联结词(合取)、(析取)、(否定)构成的不含括号的二值布尔表达式的集合：**

**试设计一个基于的属性文法，它可以计算出每个二值布尔表达式的取值。如对于句子，输出是true。**

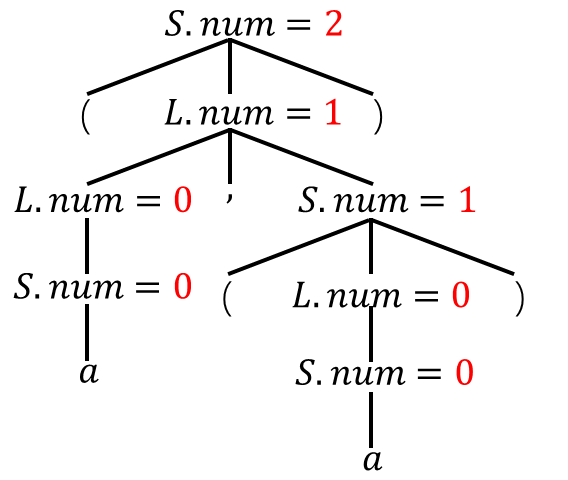
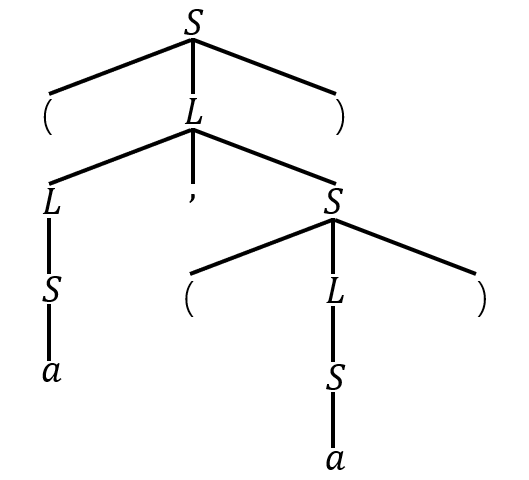
**答：**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**2. 给定文法：**

**如下是相应于的一个属性文法(或翻译模式)：**

**图7.19分别是输入串的语法分析树和对应的带标注语法树，但后者的属性值没有标出，试将其标出(即填写图7.19右图中符号=右边的值)。**

******

**图7.19 题2的语法分析树和带标注语法树**

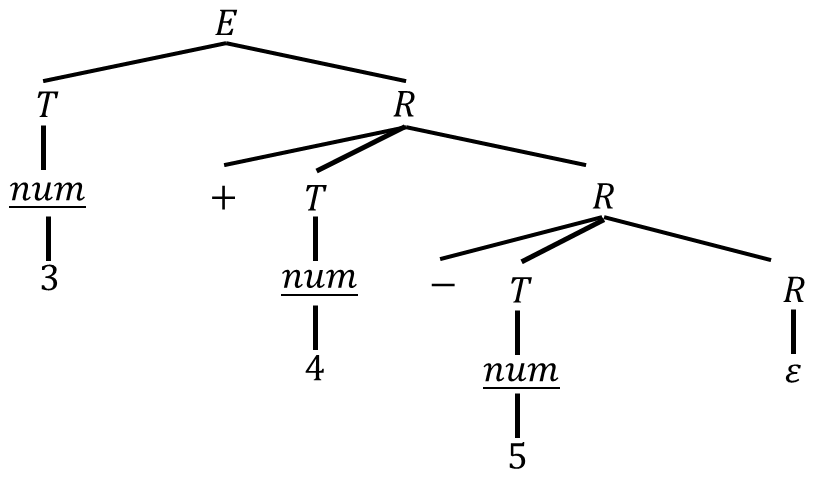
**答：如上图所示**

**4. 以下是简单表达式(只含加、减运算)计算的一个属性文法****:**

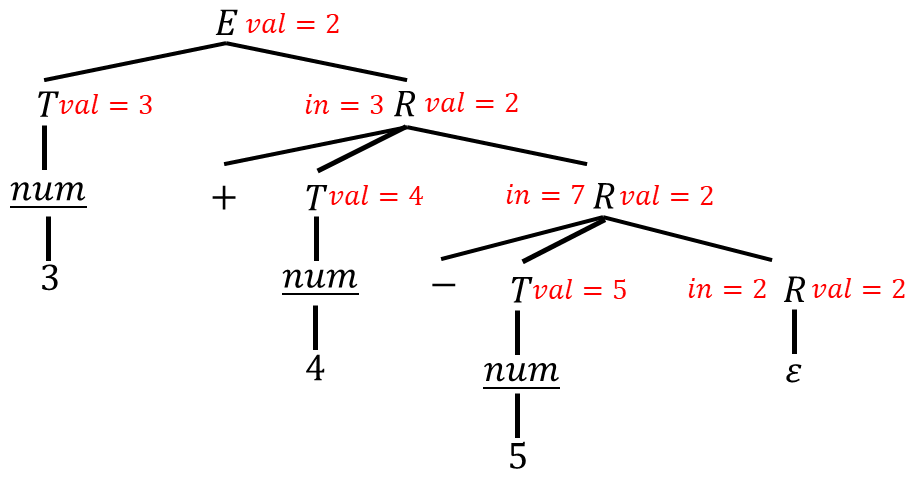
**其中，表示从词法分析程序得到的常数值。**

**试给出表达式的语法分析树和相应的带标注语法分析树。**

**答：语法分析树如下**

****

**带标注的语法树如下：**

****

**5. 题2中所给的的属性文法是一个-属性文法，故可以在自底向上分析过程中增加语义栈来计算属性值，图7.20是的一个分析表，图7.21描述了输入串的分析和求值过程(语义栈中的值对应或)，其中，第14、15行没有给出，试着补全。**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **ACTION** | | | | | **GOTO** | |
|  |  |  |  |  |  |  |
| **0** |  |  |  |  |  |  |  |
| **1** |  |  |  |  |  |  |  |
| **2** |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |
| **5** |  |  |  |  |  |  |  |
| **6** |  |  |  |  |  |  |  |
| **7** |  |  |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  |

**图7.20 题5的分析表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **步骤** | **状态栈** | **语义栈** | **符号栈** | **余留符号串** |
| **1** | **0** | **\_** |  |  |
| **2** | **02** | **\_ \_** |  |  |
| **3** | **023** | **\_ \_ \_** |  |  |
| **4** | **025** | **\_ \_ 0** |  |  |
| **5** | **024** | **\_ \_ 0** |  |  |
| **6** | **0247** | **\_ \_ 0 \_** |  |  |
| **7** | **02472** | **\_ \_ 0 \_ \_** |  |  |
| **8** | **024723** | **\_ \_ 0 \_ \_ \_** |  |  |
| **9** | **024725** | **\_ \_ 0 \_ \_ 0** |  |  |
| **10** | **024724** | **\_ \_ 0 \_ \_ 0** |  |  |
| **11** | **0247246** | **\_ \_ 0 \_ \_ 0 \_** |  |  |
| **12** | **02478** | **\_ \_ 0 \_ 1** |  |  |
| **13** | **024** | **\_ \_ 1** |  |  |
| **14** | **0246** | **\_ \_ 1 \_** |  |  |
| **15** | **01** | **\_ 2** |  |  |
| **16** | **接受** |  |  |  |

**图 7.21 题5的分析和求值过程**

**答：如上图所示**

**7.设题4中属性文法的基础文法为。**

**(1) 说明是文法。**

**(2) 如下是以作为基础文法设计的翻译模式：**

**试针对该翻译模式构造相应的递归下降(预测)翻译程序(如题6,可直接使用例7.9中的函数)。**

**答：**

**每个产生式的集合如下：**

**相同左部产生式的交集为**

**所以该文法为文法。**

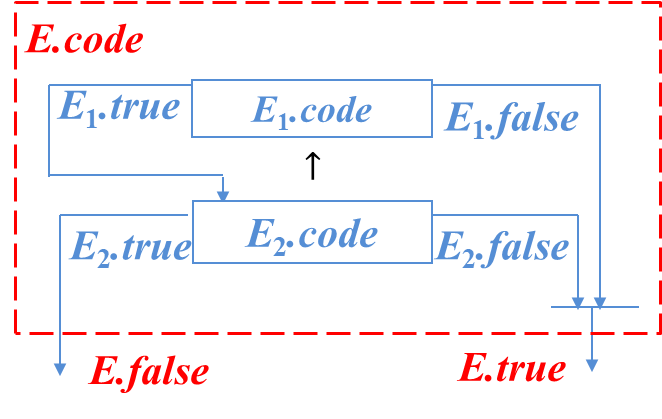
**对应的递归下降翻译程序为**

|  |
| --- |
| **文法对应的递归下降翻译程序** |
| 1. **int** ParseE() 2. { 3. Tv:=ParseT();           //变量Tv对应属性T.value 4. Ri:=Tv;                 //变量Ri对应属性R.in 5. Rv:=ParseR(Ri); 6. Ev:=Rv; 7. **return** Ev； 8. } 9. **int** ParseR(**int** f)   //形参f对应属性R.in 10. { 11. **if**(lookahead == '+')    //lookahead是当前扫描的输入符号 12. { 13. MatchToken('+'); 14. Tv:=ParseT(); 15. R1i:=f+Tv;          //R1i对应属性R1.in 16. R1v:=ParseR(R1i); 17. Rv:=R1v; 18. } 19. **else** **if**(lookahead == '-') 20. { 21. MatchToken('-'); 22. Tv:=ParseT(); 23. R1i=f-Tv; 24. R1v:=ParseR(R1i); 25. Rv:=R1v; 26. } 27. **else** **if**(lookahead == '#') 28. { 29. Rv = f; 30. } 31. **else** 32. { 33. printf("Syntax error"); 34. exit(0); 35. } 36. **return** Rv; 37. } 38. **int** ParseT() 39. { 40. **if**(lookahead == '(lexvalnum)') 41. { 42. MatchToken('(lexvalnum)'); 43. Tv:=lexval(num); 44. } 45. **else** 46. { 47. printf("Syntax error."); 48. exit(0); 49. } 50. **return** Tv; 51. } 52. **void** MatchToken(**int** expected) 53. { 54. **if**(lookahead!= expected) 55. { 56. printf("syntax error\n"); 57. exit(0); 58. } 59. **else** 60. { 61. lookahead=getToken(); 62. } 63. } |

# 第八章作业

**4. 参考8.3.3.4节采用短路代码进行布尔表达式翻译的L-翻译模式片段及用到的语义函数。若在基础文法中增加产生式，试给出与该产生式相应的语义动作集合。其中，代表“与非”逻辑算符，其语义可用其他逻辑运算定义为。**

**答：**

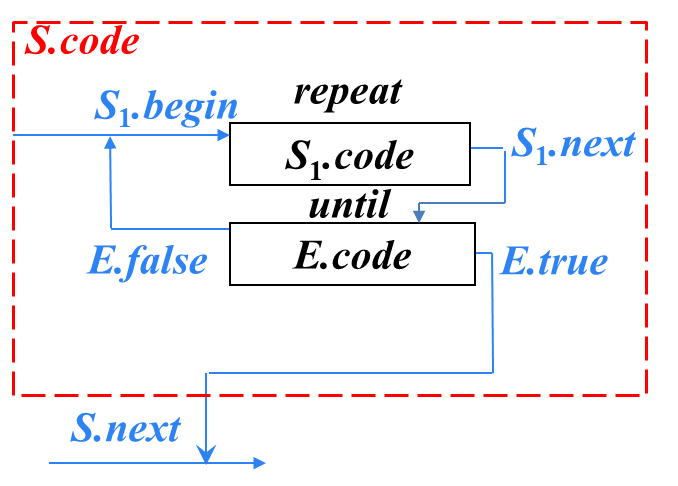
****

**或者**

**5. 参考8.3.3.5节进行控制语句（不含break）翻译的L-翻译模式片段及所用到的语义函数。若在基础文法中增加产生式,试给出与该产生式相应的语义动作集合。**

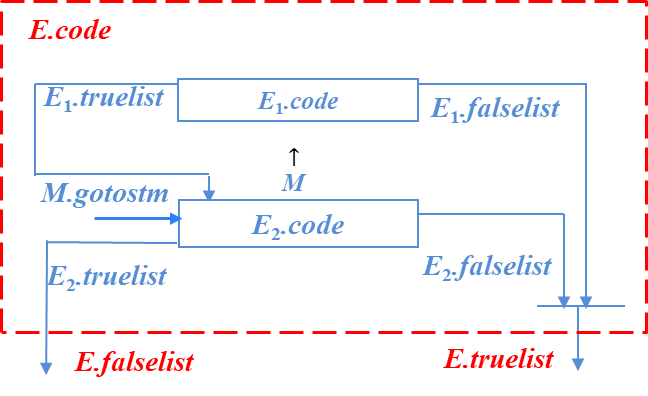
**注：控制语句 的语义为：至少执行<循环体>一次，直到<布尔表达式>成真时结束循环。**

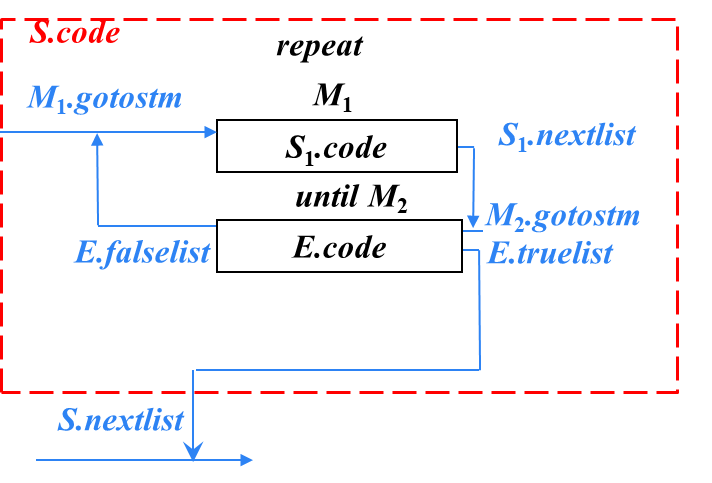
**答：**

****

**或者**

**6. 参考8.3.3.6节采用拉链与代码回填技术进行布尔表达式和控制语句翻译的S-翻译模式片段及所用到的语义函数，重复题4和题5的工作。**

****

****