语法分析程序设计与实现

实验内容及要求

编写语法分析程序，实现对算术表达式的语法分析。要求所分析算数表达式由如下的文法产生:

```
E→E+T | E—T | T
T→T*F | T/F | F
F→(E) | num
```

在对输入的算术表达式进行分析的过程中，依次输出所采用的产生式.

实现方法

- 编写LL(1)语法分析程序，要求如下
  - 编程实现算法4.2，为给定文法自动构造预测分析表
  - 编程实现算法4.1，构造LL(1)预测分析程序
- 编写语法分析程序实现自底向上的分析，要求如下
  - 构造识别该文法所有活前缀的DFA
  - 构造该文法的LR分析表
  - 编程实现算法4.3，构造LR分析程序

程序设计说明

文法预处理

从原始文法到进行分析的过程还需要进行一些预先处理,这一过程由手工计算完成,程序的输入是处理后的文法.

消除左递归后得到下列文法(省略了箭头)作为LL(1)分析程序的输入,其中`#`表示epsilon,`num`改用`n`表示.

```
E TG
G +TG -TG #
T FU
U *FU /FU #
F (E) n
```

LR分析表

对于自底向上的分析,构造识别该文法的所有活前缀的DFA和该文法的LR分析表的过程同样由手工计算完成,程序仅包含LR分析过程的实现.

手工计算后得到如下分析表作为LR分析程序的输入.

```
) - n / + ( * $ E T F
I0 0 0 S5 0 0 S4 0 0 1 2 3
I1 0 S7 0 0 S6 0 0 ACC 0 0 0
```

```
I2 0 R3 0 S9 R3 0 S8 R3 0 0 0
I3 0 R6 0 R6 R6 0 R6 R6 0 0 0
I4 0 0 S14 0 0 S13 0 0 10 11 12
I5 0 R8 0 R8 R8 0 R8 R8 0 0 0
I6 0 0 S5 0 0 S4 0 0 0 15 3
I7 0 0 S5 0 0 S4 0 0 0 16 3
I8 0 0 S5 0 0 S4 0 0 0 0 17
I9 0 0 S5 0 0 S4 0 0 0 0 18
I10 S19 S21 0 0 S20 0 0 0 0 0 0
I11 R3 R3 0 S23 R3 0 S22 0 0 0 0
I12 R6 R6 0 R6 R6 0 R6 0 0 0 0
I13 0 0 S14 0 0 S13 0 0 24 11 12
I14 R8 R8 0 R8 R8 0 R8 0 0 0 0
I15 0 R1 0 S9 R1 0 S8 R1 0 0 0
I16 0 R2 0 S9 R2 0 S8 R2 0 0 0
I17 0 R4 0 R4 R4 0 R4 R4 0 0 0
I18 0 R5 0 R5 R5 0 R5 R5 0 0 0
I19 0 R7 0 R7 R7 0 R7 R7 0 0 0
I20 0 0 S14 0 0 S13 0 0 0 25 12
I21 0 0 S14 0 0 S13 0 0 0 26 12
I22 0 0 S14 0 0 S13 0 0 0 0 27
I23 0 0 S14 0 0 S13 0 0 0 0 28
I24 S29 S21 0 0 S20 0 0 0 0 0 0
I25 R1 R1 0 S23 R1 0 S22 0 0 0 0
I26 R2 R2 0 S23 R2 0 S22 0 0 0 0
I27 R4 R4 0 R4 R4 0 R4 0 0 0 0
I28 R5 R5 0 R5 R5 0 R5 0 0 0 0
I29 R7 R7 0 R7 R7 0 R7 0 0 0 0
```

使用Golang作为实现语言,包括三个package:

- **util** 定义文法的格式,提供相关的基本函数以及FIRST和FOLLOW集合的计算方法
- **ll1** 提供LL(1)语法的预测分析表构造和预测分析程序
- **lr** 提供LR分析程序

util/types.go

- Grammar-表示文法的数据结构

```
type Grammar struct {
  N mapset.Set[rune]
  T mapset.Set[rune]
  P map[rune][]string
  S rune
}
```

- func NewGrammar-从文件中读取文法并返回一个Grammar对象的指针

- func NewProducers-从文件中读取产生式序列作为LR分析程序的输入

util/first.go

- first-表示FIRST集合的数据结构

```
var first map[rune]mapset.Set[rune]
```

- func First(a)-返回FIRST(a)

util/follow.go

- func Follow(a)-返回FOLLOW(a)

ll1/predTable.go

- table-预测分析表

```
var table map[pos]mapset.Set[string]
```

- pos-表示预测分析表中的一个位置

```
var table map[pos]mapset.Set[string]
```

- func calculateTable(g)-使用算法4.2计算文法g的预测分析表,存储到table中
- func GetTable(g)-返回文法g的预测分析表,供LL(1)预测分析程序使用

ll1/analysis.go

- tb,buf,ptr,st-分别为预测分析表,输入缓冲区,缓冲区指针,分析栈
- func Analysis(g, s)-使用算法4.1在文法g上分析输入符号串s

lr/actionTable.go

从文件中读取预先计算好的LR分析表.

lr/analysis.go

- act,buf,ptr,statStack,symbolStack-分别为LR分析表,输入缓冲区,缓冲区指针,状态栈和符号栈
- func Analysis(ps, s)-实现算法4.3,基于产生式序列ps和分析表act对符号串s进行分析

测试

使用下面的驱动程序进行测试:

```
package main

import (
  "bufio"
  "fmt"
  "os"
```

```
  "synt/ll1"
  "synt/lr"
  "synt/util"
)

func task2() {
  g := util.NewGrammar("in1.dat")
  g.ShowNTPS()
  g.ShowFirst()
  g.ShowFollow()
  ll1.ShowTable(g)

  fmt.Println("string to analysis:")
  scan := bufio.NewScanner(os.Stdin)
  scan.Scan()
  ll1.Analysis(g, scan.Text())
}

func task3() {
  g := util.NewGrammar("in0.dat")
  ps := util.NewProducers("in0.dat")
  lr.InitAction("action.dat")
  lr.ShowAction(g)

  fmt.Println("string to analysis:")
  scan := bufio.NewScanner(os.Stdin)
  scan.Scan()
  lr.Analysis(ps, scan.Text())
}

func main() {
  task2()
  task3()
}
```

1. LL(1)语法分析程序-task2()

- 输入

```
E TG
G +TG -TG #
T FU
U *FU /FU #
F (E) n
```

- 运行结果

第一组:分析`1+3/2-(0*9)-123/9`

```
reading grammar from in1.dat
N: U F E G T
T: / ( ) n + - # *
P: U -> *FU /FU #
   F -> (E) n
   E -> TG
   G -> +TG -TG #
   T -> FU
S: E
First(T omitted):
  T: ( n
  F: ( n
  E: ( n
  G: + - #
  U: * / #
Follow:
  E: $ )
  G: ) $
  T: $ ) + -
  U: ) $ + -
  F: * / + - $ )
Table:
  *   /   (   )   n   +   -   #   $
E err   err   E->TG   err   E->TG   err   err   err   err
G err   err   err   G->#    err   G->+TG    G->-TG    G->#    G->#
T err   err   T->FU   err   T->FU   err   err   err   err
U U->*FU    U->/FU    err   U->#    err   U->#    U->#    U->#    U->#
F err   err   F->(E)    err   F->n    err   err   err   err
string to analysis:
1+3/2-(0*9)-123/9

Stack   Buffer    Producer
-----------------------------------------------
$E    1+3/2-(0*9)-123/9$    E->TG
$GT   1+3/2-(0*9)-123/9$    T->FU
$GUF    1+3/2-(0*9)-123/9$    F->n
$GUn    1+3/2-(0*9)-123/9$
$GU   +3/2-(0*9)-123/9$   U->#
$G    +3/2-(0*9)-123/9$   G->+TG
$GT+    +3/2-(0*9)-123/9$
$GT   3/2-(0*9)-123/9$    T->FU
$GUF    3/2-(0*9)-123/9$    F->n
$GUn    3/2-(0*9)-123/9$
$GU   /2-(0*9)-123/9$   U->/FU
$GUF/   /2-(0*9)-123/9$
$GUF    2-(0*9)-123/9$    F->n
$GUn    2-(0*9)-123/9$
$GU   -(0*9)-123/9$   U->#
$G    -(0*9)-123/9$   G->-TG
```

```
$GT-    -(0*9)-123/9$
$GT   (0*9)-123/9$    T->FU
$GUF    (0*9)-123/9$    F->(E)
$GU)E(    (0*9)-123/9$
$GU)E   0*9)-123/9$   E->TG
$GU)GT    0*9)-123/9$   T->FU
$GU)GUF   0*9)-123/9$   F->n
$GU)GUn   0*9)-123/9$
$GU)GU    *9)-123/9$    U->*FU
$GU)GUF*    *9)-123/9$
$GU)GUF   9)-123/9$   F->n
$GU)GUn   9)-123/9$
$GU)GU    )-123/9$    U->#
$GU)G   )-123/9$    G->#
$GU)    )-123/9$
$GU   -123/9$   U->#
$G    -123/9$   G->-TG
$GT-    -123/9$
$GT   123/9$    T->FU
$GUF    123/9$    F->n
$GUn    123/9$
$GU   /9$   U->/FU
$GUF/   /9$
$GUF    9$    F->n
$GUn    9$
$GU   $   U->#
$G    $   G->#
$   $
Accepted
```

输入符号串被该文法接受.

第二组:分析`1/2-0)`

```
...

string to analysis:
1/2-0)

Stack   Buffer    Producer
-------------------------------------------------
$E    1/2-0)$   E->TG
$GT   1/2-0)$   T->FU
$GUF    1/2-0)$   F->n
$GUn    1/2-0)$
$GU   /2-0)$    U->/FU
$GUF/   /2-0)$
$GUF    2-0)$   F->n
$GUn    2-0)$
```

```
$GU   -0)$    U->#
$G    -0)$    G->-TG
$GT-    -0)$
$GT   0)$   T->FU
$GUF    0)$   F->n
$GUn    0)$
$GU   )$    U->#
$G    )$    G->#
$   )$    Unaccepted
```

输入符号串不能被该文法接受.

2. 自底向上分析程序-task3()

- 输入

产生式序列:

```
E E+T
E E-T
E T
T T*F
T T/F
T F
F (E)
F n
```

和LR分析表.

- 运行结果

第一组:分析`(9-21)/9+(1*1)`

```
reading grammar from in0.dat
reading producers from in0.dat
reading action table from in.dat ...
  + - * / ( ) n $ E T F
I0  0 0 0 0 S4  0 S5  0 1 2 3
I1  S6  S7  0 0 0 0 0 ACC 0 0 0
I2  R3  R3  S8  S9  0 0 0 R3  0 0 0
I3  R6  R6  R6  R6  0 0 0 R6  0 0 0
I4  0 0 0 0 S13 0 S14 0 10  11  12
I5  R8  R8  R8  R8  0 0 0 R8  0 0 0
I6  0 0 0 0 S4  0 S5  0 0 15  3
I7  0 0 0 0 S4  0 S5  0 0 16  3
I8  0 0 0 0 S4  0 S5  0 0 0 17
I9  0 0 0 0 S4  0 S5  0 0 0 18
I10 S20 S21 0 0 0 S19 0 0 0 0 0
I11 R3  R3  S22 S23 0 R3  0 0 0 0 0
```

```
I12 R6  R6  R6  R6  0 R6  0 0 0 0 0
I13 0 0 0 0 S13 0 S14 0 24  11  12
I14 R8  R8  R8  R8  0 R8  0 0 0 0 0
I15 R1  R1  S8  S9  0 0 0 R1  0 0 0
I16 R2  R2  S8  S9  0 0 0 R2  0 0 0
I17 R4  R4  R4  R4  0 0 0 R4  0 0 0
I18 R5  R5  R5  R5  0 0 0 R5  0 0 0
I19 R7  R7  R7  R7  0 0 0 R7  0 0 0
I20 0 0 0 0 S13 0 S14 0 0 25  12
I21 0 0 0 0 S13 0 S14 0 0 26  12
I22 0 0 0 0 S13 0 S14 0 0 0 27
I23 0 0 0 0 S13 0 S14 0 0 0 28
I24 S20 S21 0 0 0 S29 0 0 0 0 0
I25 R1  R1  S22 S23 0 R1  0 0 0 0 0
I26 R2  R2  S22 S23 0 R2  0 0 0 0 0
I27 R4  R4  R4  R4  0 R4  0 0 0 0 0
I28 R5  R5  R5  R5  0 R5  0 0 0 0 0
I29 R7  R7  R7  R7  0 R7  0 0 0 0 0
string to analysis:
(9-21)/9+(1*1)

Stack     Buffer      Action
--------------------------------------------------------------------
I0      (9-21)/9+(1*1)$     Shift 4
(
I0 I4       9-21)/9+(1*1)$      Shift 14
(n
I0 I4 I14       -21)/9+(1*1)$     reduce by F n
(F
I0 I4 I12       -21)/9+(1*1)$     reduce by T F
(T
I0 I4 I11       -21)/9+(1*1)$     reduce by E T
(E
I0 I4 I10       -21)/9+(1*1)$     Shift 21
(E-
I0 I4 I10 I21       21)/9+(1*1)$      Shift 14
(E-n
I0 I4 I10 I21 I14       )/9+(1*1)$      reduce by F n
(E-F
I0 I4 I10 I21 I12       )/9+(1*1)$      reduce by T F
(E-T
I0 I4 I10 I21 I26       )/9+(1*1)$      reduce by E E-T
(E
I0 I4 I10       )/9+(1*1)$      Shift 19
(E)
I0 I4 I10 I19       /9+(1*1)$     reduce by F (E)
F
I0 I3       /9+(1*1)$     reduce by T F
T
```

```
I0 I2       /9+(1*1)$     Shift 9
T/
I0 I2 I9      9+(1*1)$      Shift 5
T/n
I0 I2 I9 I5       +(1*1)$     reduce by F n
T/F
I0 I2 I9 I18      +(1*1)$     reduce by T T/F
T
I0 I2       +(1*1)$     reduce by E T
E
I0 I1       +(1*1)$     Shift 6
E+
I0 I1 I6      (1*1)$      Shift 4
E+(
I0 I1 I6 I4       1*1)$     Shift 14
E+(n
I0 I1 I6 I4 I14       *1)$      reduce by F n
E+(F
I0 I1 I6 I4 I12       *1)$      reduce by T F
E+(T
I0 I1 I6 I4 I11       *1)$      Shift 22
E+(T*
I0 I1 I6 I4 I11 I22       1)$     Shift 14
E+(T*n
I0 I1 I6 I4 I11 I22 I14       )$      reduce by F n
E+(T*F
I0 I1 I6 I4 I11 I22 I27       )$      reduce by T T*F
E+(T
I0 I1 I6 I4 I11       )$      reduce by E T
E+(E
I0 I1 I6 I4 I10       )$      Shift 19
E+(E)
I0 I1 I6 I4 I10 I19       $     reduce by F (E)
E+F
I0 I1 I6 I3       $     reduce by T F
E+T
I0 I1 I6 I15      $     reduce by E E+T
E
I0 I1       $     Accepted
```

输入符号串被该文法接受.

第二组:分析`(9-21)//9+(1*1)`

```
...

string to analysis:
(9-21)//9+(1*1)
```

```
Stack     Buffer      Action
--------------------------------------------------------------------
I0      (9-21)//9+(1*1)$      Shift 4
(
I0 I4       9-21)//9+(1*1)$     Shift 14
(n
I0 I4 I14       -21)//9+(1*1)$      reduce by F n
(F
I0 I4 I12       -21)//9+(1*1)$      reduce by T F
(T
I0 I4 I11       -21)//9+(1*1)$      reduce by E T
(E
I0 I4 I10       -21)//9+(1*1)$      Shift 21
(E-
I0 I4 I10 I21       21)//9+(1*1)$     Shift 14
(E-n
I0 I4 I10 I21 I14       )//9+(1*1)$     reduce by F n
(E-F
I0 I4 I10 I21 I12       )//9+(1*1)$     reduce by T F
(E-T
I0 I4 I10 I21 I26       )//9+(1*1)$     reduce by E E-T
(E
I0 I4 I10       )//9+(1*1)$     Shift 19
(E)
I0 I4 I10 I19       //9+(1*1)$      reduce by F (E)
F
I0 I3       //9+(1*1)$      reduce by T F
T
I0 I2       //9+(1*1)$      Shift 9
T/
I0 I2 I9      /9+(1*1)$     error
```

输入符号串不能被该文法接受.

除了上面的测试之外,对其他多种输入串也进行了测试,结果符合对算术表达式进行分析的预期.