中山大学数据科学与计算机学院本科生实验报告

课程名称：编译器构造实验 任课教师：陈炬桦 教学助理（TA）：

|  |  |  |  |
| --- | --- | --- | --- |
| 学年度 | 2018-2019 | 学期 | 第二学期 |
| 年级 | 2016级 | 专业（方向） | 计算机科学与技术 |
| 学号 | 16337233 | 姓名 | 王凯祺 |
| 电话 | 13434386182 | Email | [Wangkq3@mail2.sysu.edu.cn](mailto:Wangkq3@mail2.sysu.edu.cn) |
| 开始日期 | 2019/5/17 | 完成日期 | 2019/5/17 |

1. 实验题目

**LR(K) 语法分析程序**

**1.1 Description**

输入开始符号，非终结符，终结符，产生式  
输出LR(k)优先分析过程  
以拓广算术表达式G[A]: 为例  
A→E  
E→E+T | T  
T→T\*F | F  
F→(E) | a

**1.2 Input**

非终结符个数，非终结符，空格符分隔；  
终结符个数，终结符，空格符分隔；  
产生式的个数，各产生式的序号，产生式的左边和右边符号，空格符分隔；  
状态数，ACTION列数，GOTO列数，空格符分隔；  
状态，ACTION矩阵(k 0 表示空 A 0 表示接收)，GOTO矩阵(0表示 空)，空格符分隔；  
输入分析字符串，#结束

**1.3 Output**

用“ & ”分隔，左边表示栈底到栈顶符号；右边表示尚未分析的字符串。

2. 算法描述(介绍程序模块功能；流程图)

本题考查的是给定SLR(1)分析表，写一个语法分析程序。

我们知道，GOTO表的元素GOTO[Si, Xj] 表示的是状态Si面临符号Xj时所应到达的下一状态。它实际上时以文法的终结符和非终结符作为其输入符号的一个确定有穷自动机的转换表。ACTION表每个元素ACTION[Sm,Ai]所规定的动作是下述四个动作之一：1）移进s；2）归约A🡪b；3）接收；4）错误处理。

LR分析器由两部分组成：栈中符号串和待扫描输入串。

1. 若ACTION[Sm, Ai]=移进s，则分析器执行一个移进动作。分析器将输入符号Ai和下一状态S=GOTO[Sm, Ai]移进栈，再将Ai+1变成新的当前输入符号。
2. 若ACTION[Sm, Ai]=归约A🡪B，则分析器执行一个归约动作，先弹出B.length() \* 2个元素，然后加入A和GOTO[Sm-r, A]。若仍满足归约条件，则继续归约，直至不能归约为止。
3. 若ACTION[Sm, Ai]=接收，表示分析工作正常完成。
4. 若ACTION[Sm, Ai]=错误，表示分析器发现输入串有错，进行出错处理。

我们用两个循环，外层循环是扫描输入串，内层循环是执行循环归约。扫描输入串新增一个字符时，我们先检查是否能归约，若能归约则循环归约，直到不能归约为止。然后依次检查是否为错误、接收、移进。若为移进，则移进栈，然后继续外层循环，扫描下一字符。

3. 测试数据

样例输入

A

4 E T F A

6 a + \* ( ) #

7

0 A E

1 E E+T

2 E T

3 T T\*F

4 T F

5 F (E)

6 F a

12 6 3

0 s 5 k 0 k 0 s 4 k 0 k 0 1 2 3

1 k 0 s 6 k 0 k 0 k 0 A 0 0 0 0

2 k 0 r 2 s 7 k 0 r 2 r 2 0 0 0

3 k 0 r 4 r 4 k 0 r 4 r 4 0 0 0

4 s 5 k 0 k 0 s 4 k 0 k 0 8 2 3

5 k 0 r 6 r 6 k 0 r 6 r 6 0 0 0

6 s 5 k 0 k 0 s 4 k 0 k 0 0 9 3

7 s 5 k 0 k 0 s 4 k 0 k 0 0 0 10

8 k 0 s 6 k 0 k 0 s 11 k 0 0 0 0

9 k 0 r 1 s 7 k 0 r 1 r 1 0 0 0

10 k 0 r 3 r 3 k 0 r 3 r 3 0 0 0

11 k 0 r 5 r 5 k 0 r 5 r 5 0 0 0

(a+a)\*a#

样例输出

#0 & (a+a)\*a#

#0(4 & a+a)\*a#

#0(4a5 & +a)\*a#

#0(4F3 & +a)\*a#

#0(4T2 & +a)\*a#

#0(4E8 & +a)\*a#

#0(4E8+6 & a)\*a#

#0(4E8+6a5 & )\*a#

#0(4E8+6F3 & )\*a#

#0(4E8+6T9 & )\*a#

#0(4E8 & )\*a#

#0(4E8)11 & \*a#

#0F3 & \*a#

#0T2 & \*a#

#0T2\*7 & a#

#0T2\*7a5 & #

#0T2\*7F10 & #

#0T2 & #

#0E1 & #

4. 程序清单

#include <bits/stdc++.h>

#define N 100

**using** **namespace** std;

**char** S, Vn[N], Vt[N], st[N], act1[N][N];

**int** n, m, k, na, nb, nc, act2[N][N], go[N][N];

pair<**char**, string> tr[N];

vector<**int**> stk;

**int** Vnid(**char** x) {

**for** (**int** i = 0; i < n; ++i)

**if** (Vn[i] == x)

**return** i;

**return** -1;

}

**int** Vtid(**char** x) {

**for** (**int** i = 0; i < m; ++i)

**if** (Vt[i] == x)

**return** i;

**return** -1;

}

**void** out(**int** x) {

**for** (**int** i = 0; i < (**int**)stk.size(); ++i) {

**if** (i % 2 == 0)

printf("%c", **char**(stk[i]));

**else**

printf("%d", stk[i]);

}

printf(" & ");

printf("%s\n", st + x);

}

**int** main() {

#ifdef DEBUG

freopen("a.in", "r", stdin);

#endif

scanf("%s", st);

S = st[0];

scanf("%d", &n);

**for** (**int** i = 0; i < n; ++i) {

scanf("%s", st);

Vn[i] = st[0];

}

scanf("%d", &m);

**for** (**int** i = 0; i < m; ++i) {

scanf("%s", st);

Vt[i] = st[0];

}

scanf("%d", &k);

**for** (**int** i = 0; i < k; ++i) {

**int** id;

**char** x;

string y;

scanf("%d", &id);

scanf("%s", st);

x = st[0];

scanf("%s", st);

y = st;

tr[i] = make\_pair(x, y);

}

scanf("%d%d%d", &na, &nb, &nc);

assert(nb == m);

assert(nc == n - 1);

**for** (**int** i = 0; i < na; ++i) {

**int** id;

scanf("%d", &id);

**for** (**int** j = 0; j < nb; ++j) {

scanf("%s", st);

act1[i][j] = st[0];

scanf("%d", &act2[i][j]);

}

**for** (**int** j = 0; j < nc; ++j) {

scanf("%d", &go[i][j]);

}

}

scanf("%s", st);

**int** len = strlen(st);

stk.push\_back(**int**('#'));

stk.push\_back(0);

**int** i = 0;

out(i);

**while** (i < len) {

**while** (act1[stk.back()][Vtid(st[i])] == 'r') {

**int** Act2 = act2[stk.back()][Vtid(st[i])];

**for** (**int** j = 0; j < (**int**)tr[Act2].second.length() \* 2; ++j)

stk.pop\_back();

**int** top = stk.back();

stk.push\_back(**int**(tr[Act2].first));

stk.push\_back(go[top][Vnid(tr[Act2].first)]);

out(i);

}

**if** (act1[stk.back()][Vtid(st[i])] == 'A') {

**break**;

}

**if** (act1[stk.back()][Vtid(st[i])] == 's') {

**int** Act2 = act2[stk.back()][Vtid(st[i])];

stk.push\_back(**int**(st[i]));

stk.push\_back(Act2);

++ i;

out(i);

}

assert(act1[stk.back()][Vtid(st[i])] != 'k');

}

}