layout: post
title: 语法分析程序
subtitle: 使用LL(1)文法
date: 2017-11-03
author: Awybupt
header-img: img/page-CL.jpg
catalog: true
tags:
- 编译原理

- C++

实验环境

Mac OS X10.13.6
Xcode
c++

LR语法分析实验报告

- 实验题目
  - 该程序使用实现对算术表达式自底向上的语法分析，并且在对输入表达式进行分析的过程中，输出分析动作，移进或者用哪个产生式进行规约，该程序使用的是LR语法分析程序，手动构造了识别所有活前缀的DFA，为给定文法构造LR分析表，并通过预测分析表对输入的表达式进行分析，并将栈顶状态和预测分析过程详细输出，如果匹配成功则接受，如果匹配不成功则返回错误信息。
  - 特别的是，该程序参照书上129页的有关LR分析的错误处理与恢复表对一些可能出现的错误进行报错和局部恢复，在action表中设置相应的错误处理过程入口，调用相应的过程进行错误处理和恢复，使语法分析能继续进行。
  - 编写语法分析程序,实现对算数表达式的语法分析.
  - 表达式由如下文法产生
    - E->E+T|E-T|T
    - T->T*F|T/F|F

        - F->id|(E)|num

```
    map <int,string> table;
    vector<string>  v{"","+","-","*","/","(",")","N","#","E","T","F"};
    vector<string> anas{"","E->E+T","E->E-T","E->T","T->T*F","T->T/F","T->F","F
->(E)","F->N"};
    string shuru;
    string fenxizhan="0";
    vector<record> re;
```

- 实验过程
    - 首先由生成式确定扩展文法,构造DFA,然后构造出了LR语法分析表
        - 构造map类
            - int的构成是至少有三位数字,三位数第一位数是区分字符,无任何意义.存在是防止7不能表示为07,有1就可以表示为107
            - 前面的一位或者两位是语法分析表中的状态位置
            - string类型的特点就是存储动作,并为这些动作创造特殊的动作,完成从程序到实际的转换
        - 将分析栈中的字符输入一个vector这样的好处是可以动态构造,在后期可以用较小的代价实现修改这个集合
        - 在vector ans中存放生成式,第一个位置存放空字符串表示跳过这个字符串,在对应的时候就可以从1开始避免频繁的转换
        - 在分析栈中存放0,作为栈底
    - 根据算法4.3
        - 对输入根据语法分析表就可以分析并且输出了
        - 严格按照算法4.3,就可以构造下面的代码段实现分析

```
void Lr::fenxi(){
shuru=shuru+"#";
do{
    string a=zuozhanding(shuru);
    string b=youzhanding(fenxizhan);

    cout<<fenxizhan<<"\t\t\t"<<shuru<<"\t\t\t";
    string dongzuo=table[chaxunshu(a,b)];
    //cout<<dongzuo;
    if(dongzuo[0]=='S')
    {
        fenxizhan=fenxizhan+a;
```

```
        string ttt=zuotanzhan(dongzuo);
        fenxizhan=fenxizhan+ttt;
        shuru=zuotanzhan(shuru);
        cout<<"shift->"<<zuotanzhan(dongzuo)<<endl;
    }
    else if(dongzuo[0]=='R')
    {
        string jianshao=anas[stringtoint(zuotanzhan(dongzuo))];
        cout<<"reduce BY->"<<anas[stringtoint(zuotanzhan(dongzuo))]<<endl;
        string A=zuozhanding(jianshao);
        string b=zuotanzhan(zuotanzhan(zuotanzhan(jianshao)));
        fenxizhan=reducestring(fenxizhan, int(b.size()));
        string douo=table[chaxunshu(A,youzhanding(fenxizhan))];
        //cout<<douo<<endl;
        fenxizhan=fenxizhan+A;
        fenxizhan=fenxizhan+shuzitan(douo);

    }
    else if(dongzuo=="ACC"){
        cout<<"ACC"<<endl;
        break;
    }
    else
    {
        cout<<"error!"<<endl;
        break;
    }

}
while(true);
}
```

- 首先进行词法分析,然后再进行语法分析,有效的扩展了程序的可用性

输入输出

- 输入
  - 只包含合法字符的字符串
    - (3+2)
    - (3+2)*(4/2)
    - ...

- 输出
  - 正确的输入

```
请输入待识别字符串:(12+13)/(24*13)
(N+N)/(N*N)0            (N+N)/(N*N)#            shift->4
0(4         N+N)/(N*N)#         shift->5
0(4N5           +N)/(N*N)#          reduce BY->F->N
0(4F3           +N)/(N*N)#          reduce BY->T->F
0(4T2           +N)/(N*N)#          reduce BY->E->T
0(4E10          +N)/(N*N)#          shift->6
0(4E10+6            N)/(N*N)#           shift->5
0(4E10+6N5          )/(N*N)#            reduce BY->F->N
0(4E10+6F3          )/(N*N)#            reduce BY->T->F
0(4E10+6T11         )/(N*N)#            reduce BY->E->E+T
0(4E10          )/(N*N)#            shift->15
0(4E10)15           /(N*N)#         reduce BY->F->(E)
0F3         /(N*N)#         reduce BY->T->F
0T2         /(N*N)#         shift->9
0T2/9           (N*N)#          shift->4
0T2/9(4         N*N)#           shift->5
0T2/9(4N5           *N)#            reduce BY->F->N
0T2/9(4F3           *N)#            reduce BY->T->F
0T2/9(4T2           *N)#            shift->8
0T2/9(4T2*8         N)#         shift->5
0T2/9(4T2*8N5           )#          reduce BY->F->N
0T2/9(4T2*8F13          )#          reduce BY->T->T*F
0T2/9(4T2           )#          reduce BY->E->T
0T2/9(4E10          )#          shift->15
0T2/9(4E10)15           #           reduce BY->F->(E)
0T2/9F14            #           reduce BY->T->T/F
0T2         #           reduce BY->E->T
0E1         #           ACC
Program ended with exit code: 0
```

- 错误的输入

```
请输入待识别字符串:(12+13)/(24*13
(N+N)/(N*N0          (N+N)/(N*N#          shift->4
0(4          N+N)/(N*N#          shift->5
0(4N5            +N)/(N*N#            reduce BY->F->N
0(4F3            +N)/(N*N#            reduce BY->T->F
0(4T2            +N)/(N*N#            reduce BY->E->T
0(4E10           +N)/(N*N#            shift->6
0(4E10+6             N)/(N*N#             shift->5
0(4E10+6N5           )/(N*N#          reduce BY->F->N
0(4E10+6F3           )/(N*N#          reduce BY->T->F
0(4E10+6T11          )/(N*N#          reduce BY->E->E+T
0(4E10           )/(N*N#          shift->15
0(4E10)15            /(N*N#           reduce BY->F->(E)
0F3          /(N*N#          reduce BY->T->F
0T2          /(N*N#          shift->9
0T2/9            (N*N#            shift->4
0T2/9(4          N*N#             shift->5
0T2/9(4N5            *N#          reduce BY->F->N
0T2/9(4F3            *N#          reduce BY->T->F
0T2/9(4T2            *N#          shift->8
0T2/9(4T2*8          N#           shift->5
0T2/9(4T2*8N5            #            error!
Program ended with exit code: 0
```

- 无效字符

```
请输入待识别字符串:(12+13)/(24*13@)
0           (N+N)/(N*N@)#           shift->4
0(4         N+N)/(N*N@)#           shift->5
0(4N5           +N)/(N*N@)#        reduce BY->F->N
0(4F3           +N)/(N*N@)#        reduce BY->T->F
0(4T2           +N)/(N*N@)#        reduce BY->E->T
0(4E10          +N)/(N*N@)#        shift->6
0(4E10+6            N)/(N*N@)#         shift->5
0(4E10+6N5          )/(N*N@)#          reduce BY->F->N
0(4E10+6F3          )/(N*N@)#          reduce BY->T->F
0(4E10+6T11         )/(N*N@)#          reduce BY->E->E+T
0(4E10          )/(N*N@)#          shift->15
0(4E10)15           /(N*N@)#           reduce BY->F->(E)
0F3         /(N*N@)#            reduce BY->T->F
0T2         /(N*N@)#            shift->9
0T2/9           (N*N@)#         shift->4
0T2/9(4         N*N@)#          shift->5
0T2/9(4N5           *N@)#           reduce BY->F->N
0T2/9(4F3           *N@)#           reduce BY->T->F
0T2/9(4T2           *N@)#           shift->8
0T2/9(4T2*8         N@)#            shift->5
0T2/9(4T2*8N5           @)#         error!
Program ended with exit code: 0
```

实验收获

- 经过这次LR语法分析程序,首先完整的整理了一遍Lr语法分析的过程,并且把这个步骤准备为一个程序,可以顺序执行
- 在构造语法分析表的过程中,使用了许多以前不在意的细节
- 在字符串处理方面取得了收获
- 增强了我学习相关知识的信心

备注

- 因为是在xcode写的所以输入尽量为utf-8字符,否则可能出现字符,最简单就是直接复制我在代码中的字符串即可
- 有任何问题都可以随时联系我:13051957575
- 邮箱:wxscs@qq.com