如何将文法改造为LL（1）的？

典型例题：第四章习题5

大概要做的几个步骤：

1. 消除左递归：P70算法，不难----------------------------------------薛方岗
2. 提取左公因子：P71的规则，不难------------------------------------薛方岗
3. 求FIRST与FOLLOW集合------------------------------------冼健邦，李珍岩

P79上半页算法，这里是难点

1. 确定改造后的文法是否为LL（1）？（这里暂时不需要实现，有时间再做）

P73的三个条件，也是难点

1. 构造LL(1)预测表：P79下半页算法,难点-------------------------------刁浩
2. 有时间就做一个句子的识别过程，打印栈的变化过程----------------------随意

一、数据结构

产生式

Production

{

String left;

String right;

}

例如E->Aa|Bb则分为2个production来存放

p1.left = “E” p1.right = “Aa”

p2.left = “E” p2.right = “Bb”

First集合

如下类的每一个对象代表一个first集合

**public** **class** First {

**public** String vt;

**public** HashSet<Pair<String,Production>> firstSet;

**public** First(String VT)

{

**this**.vt=VT;

}

**public** **void** addElement(String element, Production production)

{

Pair<String,Production> pair = **new** Pair<String, Production>(element,production);

firstSet.add(pair);

//firstSet(A) = { [element1,"A->???"] , [element2,"A->???"] , [element3,"A->???"]}

}

}

Follow集合

HashMap<String,String> map;

Follow(A)={a,b,c}

map[“A”] = “abc”

LL1分析构造表

String[][] table; //表格

HashMap<String, Integer> mapVt,mapVn; //vt表示终结符，vn代表非终结符

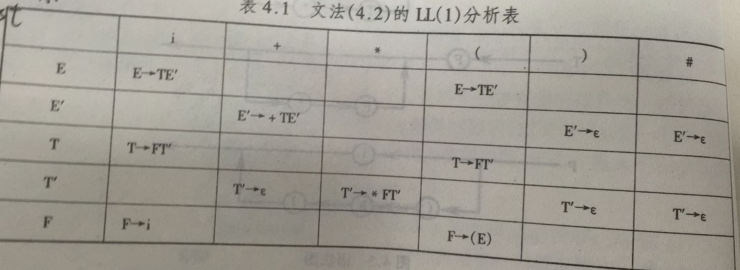
拿P76的表格举例：

mapVt[“i”]=0, mapvt[“+”]=1

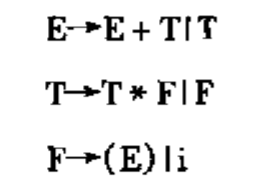
mapVn[“E”]=0, mapvn[E’]=2

其他以此类推

table[mapVn[“E”]][mapVt[“i”]] = “E->TE’”



二、测试样例

输入产生式，默认第一个产生式左部（即E）是开始符号：  


在数据结构中：

Production[] p;

p[0].left = E

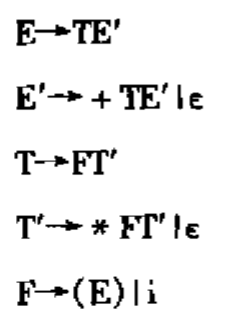
p[0].right = E+T

p[1].left = E

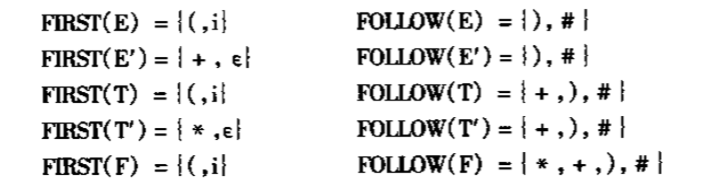
p[1].right = T

……

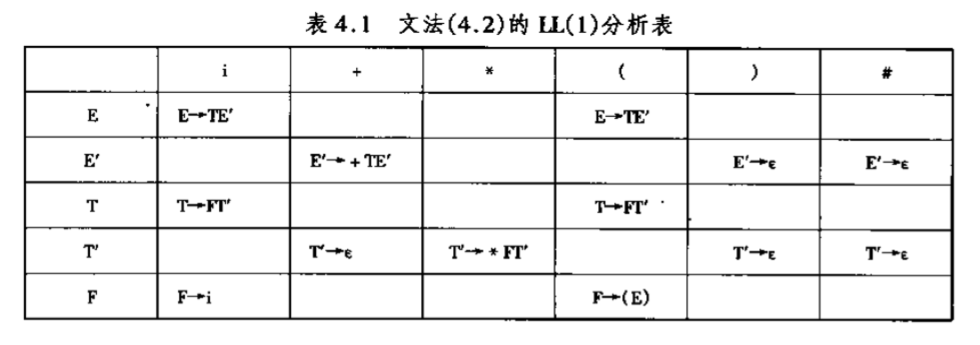
消除左递归后



first与follow

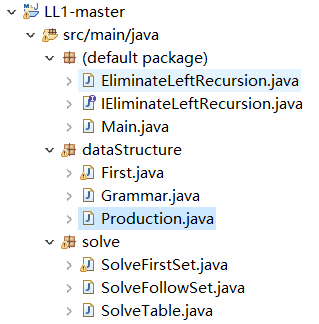


表格



三、代码框架

代码框架中很多类的变量直接定义为public，嫌麻烦的话，就直接通过 xxx.var 去修改就行了。



1. 构造LL1文法的总体框架

看MAIN.java

2. dataStructure包是数据结构

包括First集合，文法Grammar，产生式Production

若有其他需要，自己添加即可。

3. solve包

把数据处理的函数都写在这里, 有需要自己添加

四、变量名/类名的解释

产生式 Production

集合 Set

表格 Table

递归 Recursion