自底向上语法分析程序

谢蔚钦 2020211451

2022 年 11 月 20 日

1 实验要求

给定上下文无关文法如下所示:
E → E+T | E–T | T
T → T*F | T/F | F
F → (E) | num
编写语法分析程序实现自底向上的分析，要求如下:

1. 构造识别该文法所有活前缀的 DFA.
2. 构造该文法的 LR 分析表.
3. 编程实现算法 4.3, 构造 LR 分析程序.

2 实验过程

首先要构造能够识别该文法的 LR(0) 项目集规范族及识别其所有活前缀的 DFA, 构造算法如下所示:

输入： 文法G
输出： G的LR(0)项目集规范族C
方法：

C=closure(S′→ • S);

do
for (对C中的每一个项目集I和每一个文法符号X)
if (go(I, X)不为空，且不在C中)
把go(I, X)加入C中;
while (没有新项目集加入C中);

这里 closure(S′ • S) ϵ 的有效项目集

根据该算法, 得到 DFA 如下所示:

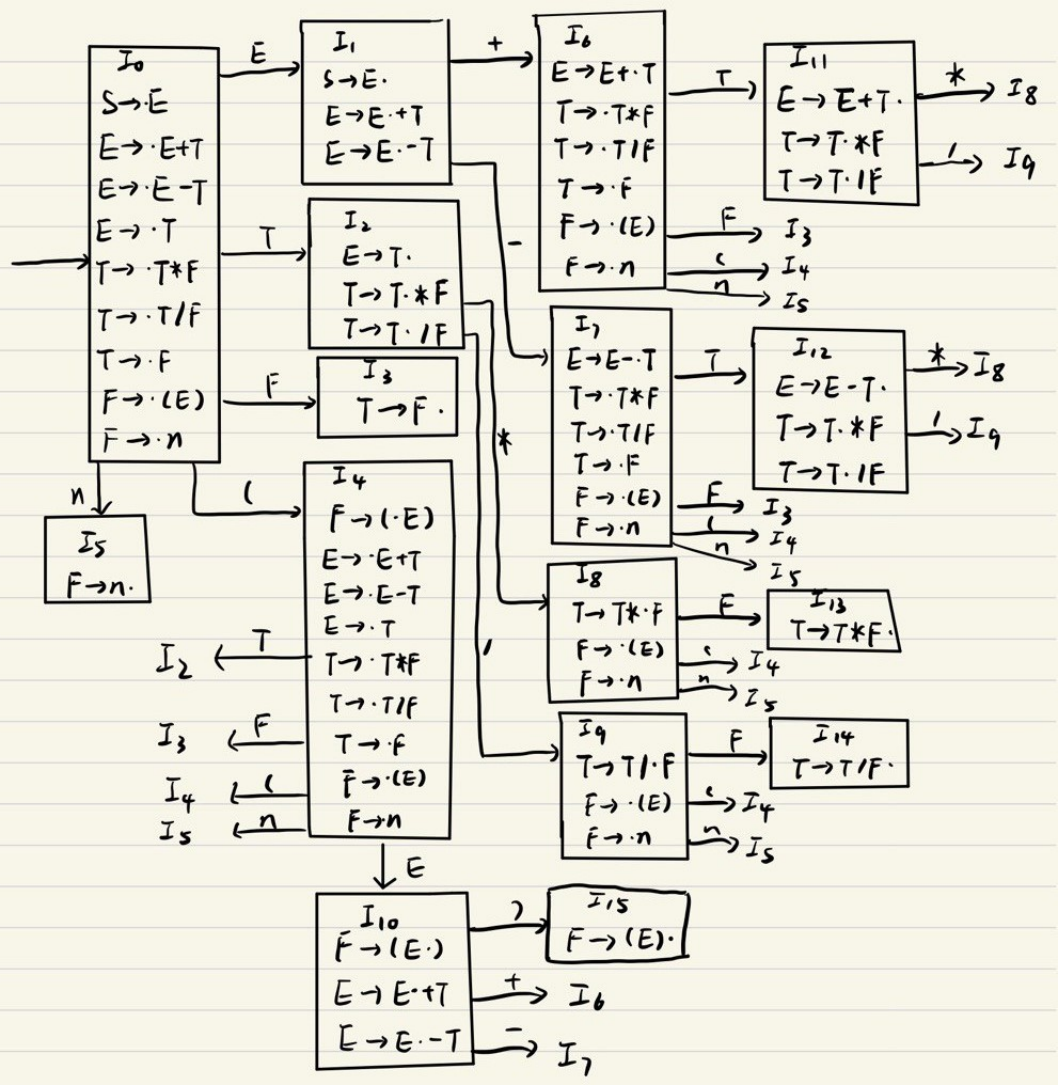

图 1: LR(0) 项目集规范族及识别其所有活前缀的 DFA

对其进行观察, 发现其是 SLR(1) 文法, 因为其中不存在移进-规约冲突和规约-规约冲突, 然后再构造该文法的 LR 分析表, 构造使用算法如下所示:

输入：拓广文法 G'

输出：G' 的 SLR 分析表

1. 构造 G' 的 LR(0) 项目集规范族 C={I0, I1, ⋯, In}.

2. 对于状态 i(对应于项目集 I_i 的状态) 的分析动作如下:

   (a) 若 A→α • aβ∈I_i，且 go(I_i, a)=I_j，则置 action[i, a]=S_j

(b) 若 A→$\alpha \bullet$ a∈I_i, 则对所有 a∈FOLLOW(A), 置 action[i, a]=R A→α

(c) 若 S'→S $\bullet \in I_i$，则置 action[i, $]=ACC，表示分析成功

3. 分析表中凡不能用规则 (2)、(3) 填入信息的空白表项，均置为出错标志 error.

4. 分析程序的初态是包含项目 $S' \bullet S$ 的有效项目集所对应的状态。

所构造的分析表如下所示:

状态	action								goto		
	+	-	*	/	(	)	n	$	E	T	F
0					S4		S5		1	2	3
1	S6	S7						ACC			
2	R3	R3	S8	S9		R3		R3			
3	R6	R6	R6	R6		R6		R6			
4					S4		S5		10	2	3
5	R8	R8	R8	R8		R8		R8			
6					S4		S5			11	3
7					S4		S5			12	3
8					S4		S5				13
9					S4		S5				14
10	S6	S7				S15					
11	R1	R1	S8	S9		R1		R1			
12	R2	R2	S8	S9		R2		R2			
13	R4	R4	R4	R4		R4		R4			
14	R5	R5	R5	R5		R5		R5			
15	R7	R7	R7	R7		R7		R7			

3 代码编写以及结果分析

最后构造 LR 分析程序, 本次实验使用 Python 编程语言进行构造 (代码在附件中给出), 通过编写一个 mapping 函数:def mapping(x, y), 其中 x 为当前状态, y 为输入字符, 来确定规约过程, 然后通过使用算法 4.3 来构造分析程序

直接给出 LR 分析程序的分析过程, 如下所示:

输入字符串: (n*n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n

输出结果 (部分, 输出过长完整版在附件中):

```
please enter string you want to analyze:
(n*n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n
栈                输入                  输出
0/_               (n*n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n$                shift 4
0/_ 4/(               n*n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n$               shift 5
0/_ 4/( 5/n               *n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n$            reduced by F -> n
0/_ 4/( 3/F               *n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n$            reduced by T -> F
0/_ 4/( 2/T               *n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n$              shift 8
0/_ 4/( 2/T 8/*               n-(n/n-n)+n)/((n*n)+(n*(n-n)))*n$           shift 5
0/_ 4/( 2/T 8/* 5/n               -(n/n-n)+n)/((n*n)+(n*(n-n)))*n$        reduced by F -> n
0/_ 4/( 2/T 8/* 13/F               -(n/n-n)+n)/((n*n)+(n*(n-n)))*n$       reduced by T -> T*F
0/_ 4/( 2/T               -(n/n-n)+n)/((n*n)+(n*(n-n)))*n$            reduced by E -> T
0/_ 4/( 10/E               -(n/n-n)+n)/((n*n)+(n*(n-n)))*n$               shift 7
0/_ 4/( 10/E 7/-               (n/n-n)+n)/((n*n)+(n*(n-n)))*n$            shift 4
0/_ 4/( 10/E 7/- 4/(               n/n-n)+n)/((n*n)+(n*(n-n)))*n$         shift 5
0/_ 4/( 10/E 7/- 4/( 5/n               /n-n)+n)/((n*n)+(n*(n-n)))*n$      reduced by F -> n
0/_ 4/( 10/E 7/- 4/( 3/F               /n-n)+n)/((n*n)+(n*(n-n)))*n$      reduced by T -> F
0/_ 4/( 10/E 7/- 4/( 2/T               /n-n)+n)/((n*n)+(n*(n-n)))*n$         shift 9
0/_ 4/( 10/E 7/- 4/( 2/T 9//               n-n)+n)/((n*n)+(n*(n-n)))*n$      shift 5
0/_ 4/( 10/E 7/- 4/( 2/T 9// 5/n               -n)+n)/((n*n)+(n*(n-n)))*n$   reduced by F -> n
0/_ 4/( 10/E 7/- 4/( 2/T 9// 14/F               -n)+n)/((n*n)+(n*(n-n)))*n$  reduced by T -> T/F
0/_ 4/( 10/E 7/- 4/( 2/T               -n)+n)/((n*n)+(n*(n-n)))*n$         reduced by E -> T
0/_ 4/( 10/E 7/- 4/( 10/E               -n)+n)/((n*n)+(n*(n-n)))*n$          shift 7
0/_ 4/( 10/E 7/- 4/( 10/E 7/-               n)+n)/((n*n)+(n*(n-n)))*n$       shift 5
0/_ 4/( 10/E 7/- 4/( 10/E 7/- 5/n               )+n)/((n*n)+(n*(n-n)))*n$    reduced by F -> n
0/_ 4/( 10/E 7/- 4/( 10/E 7/- 3/F               )+n)/((n*n)+(n*(n-n)))*n$    reduced by T -> F
0/_ 4/( 10/E 7/- 4/( 10/E 7/- 12/T               )+n)/((n*n)+(n*(n-n)))*n$   reduced by E -> E-T
0/_ 4/( 10/E 7/- 4/( 10/E               )+n)/((n*n)+(n*(n-n)))*n$          shift 15
0/_ 4/( 10/E 7/- 4/( 10/E 15/)               +n)/((n*n)+(n*(n-n)))*n$      reduced by F -> (E)
0/_ 4/( 10/E 7/- 3/F               +n)/((n*n)+(n*(n-n)))*n$              reduced by T -> F
0/_ 4/( 10/E 7/- 12/T               +n)/((n*n)+(n*(n-n)))*n$             reduced by E -> E-T
```

图 2: 输出结果

4 实验总结

本次实验较为容易, 由于前两个步骤未要求通过编程实现, 且文法较为简单, 所以实现难度较小, 算法 4.3 个人感觉相比构造非递归预测分析程序的算法要更简单, 且本次实验未对编程语言有要求, 所以选择了 Python 来实现 LR 分析程序, 通过使用字典等数据结构实现栈的功能, 同时也实现了对状态的保存功能以及缓冲区的功能。

总的来说, 本次实验收获还是比较大的, 让我对 SLR(1) 文法的预测分析表构造方法有了更深的认识, 同时让我对编译原理这门课有了更深入的了解。