编译原理报告

2016302362-胡川洋

概述

目前已经完成词法分析器的编码，而后针对群里给出的文法实现了语法分析器的编码、中间代码生成以及最终汇编代码的生成

文法

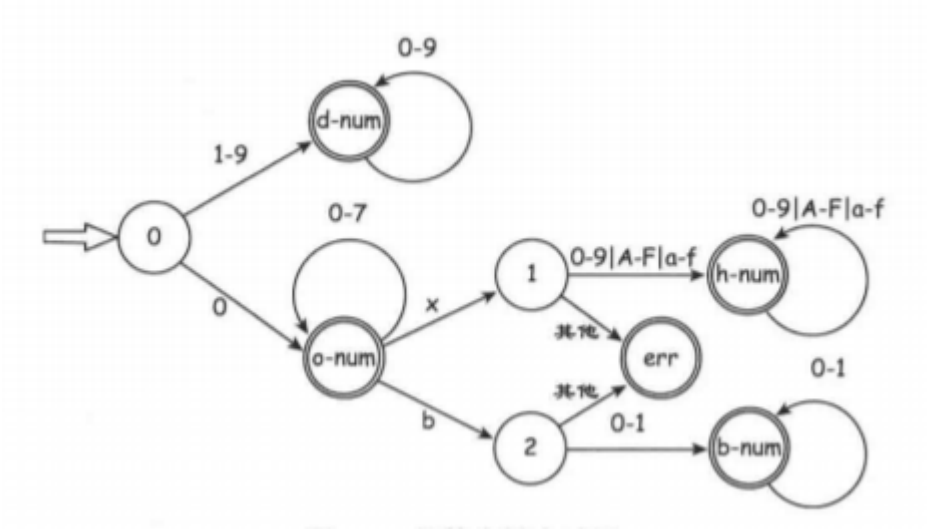
非终结符	推导式
E	TE'
E'	[+] TE'
E'	ϵ
T	FT'
T'	[*] FT'
T'	ϵ
F	i
F	[( ] E [ )]
i	[数字]

说明：原文法中 i 只能推导出0-9中的一个数字，这里改为能够推导出一个数字，以支持简单表达式的编译

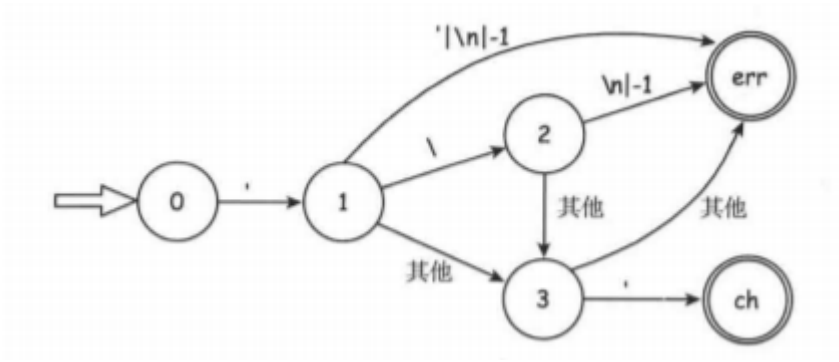
词法分析

使用DFA来进行对单词的识别，主要有数字常量识别DFA，字符常量识别DFA，字符串常量识别DFA，标识符识别DFA以及界符识别DFA

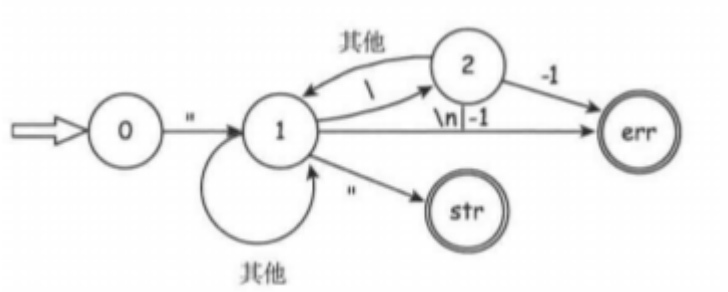
数字常量识别DFA：（支持十六进制、十进制、八进制、二进制）



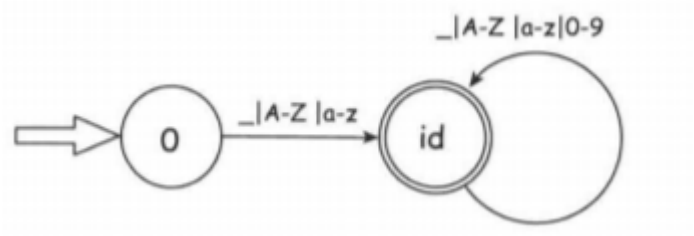
字符常量识别DFA：



字符串常量识别DFA：



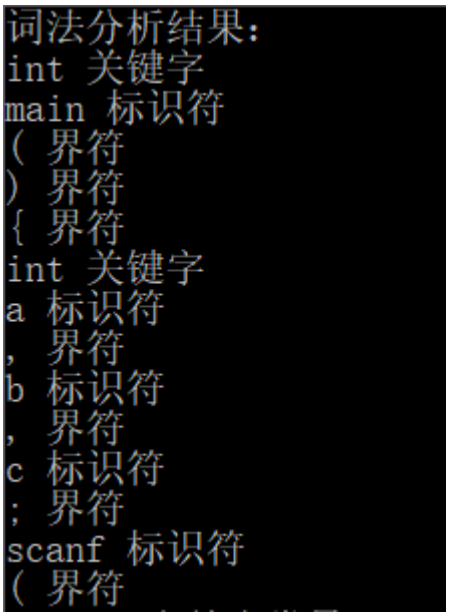
标识符识别DFA：

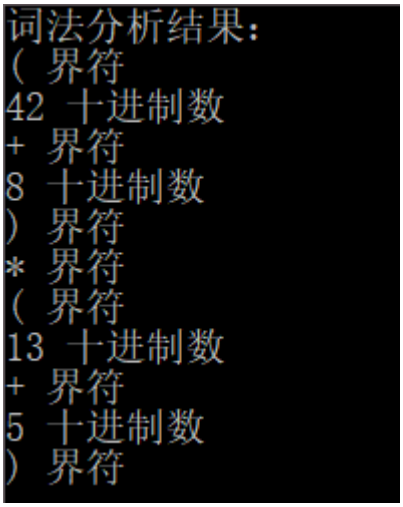


界符识别DFA：

对所有的界符一一进行匹配即可

词法分析结果：



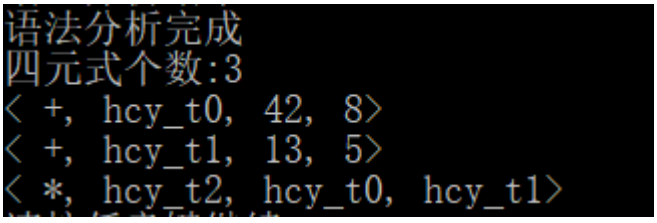


语法分析与语法制导翻译

采用递归下降子程序，在完成子程序的识别后对应生成四元式

待分析表达式：(42+8)*(13+5)

分析结果如下：



生成最终代码

根据中间代码（四元式）再进行翻译生成最终的汇编代码，这里采用MIPS指令集，产生的MIPS汇编代码可以直接在MARS上运行查看最终结果，在翻译时，对于二元运算式使用了$t0与$t1两个寄存器进行运算结果送$t0，同时将结果压栈即($sp+4)，针对上述表达式最终生成的汇编代码如下：

```
li $t0, 1
li $t1, 2
add $t0, $t0, $t1
sw $t0, ($sp)
addi $sp, $sp, 4
li $t0, 3
li $t1, 4
add $t0, $t0, $t1
sw $t0, ($sp)
addi $sp, $sp, 4
subi $sp, $sp, 4
lw $t0, ($sp)
subi $sp, $sp, 4
lw $t1, ($sp)
mul $t0, $t0, $t1
sw $t0, ($sp)
addi $sp, $sp, 4
```

最终在MARS中的执行结果如下：

$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000384
$t1	9	0x00000032
$t2	10	0x00000000
$t3	11	0x00000000

可以看到最终的结果在$t0中为0x384即十进制的900，与实际结果一致

接下来

准备开始针对C0文法编写语法分析器并生成中间代码，涉及到了符号表以及作用域的管理，这次的文法较为简单只是用了简单的栈即完成了，因此接下来需要进一步实现符号表的管理