数值分析实验报告

姓名： 张童

学号： 201600262022

班级： 16 级泰山学堂计算机取向

指导老师：王丽荣

目录

一、综述

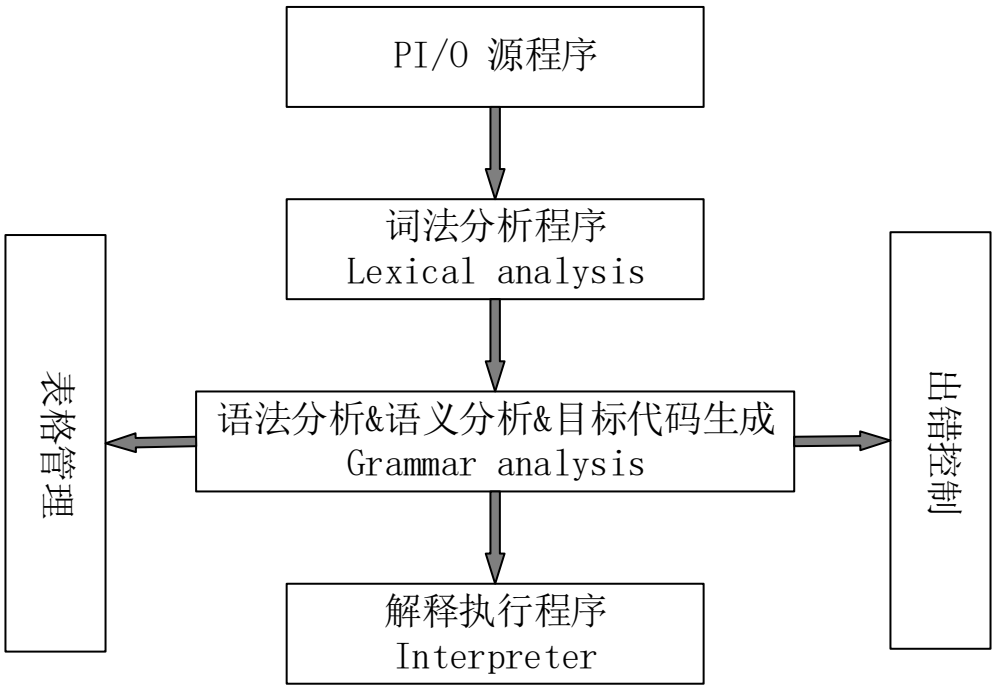
编译器（compiler），是一种计算机程序，它会将用某种编程语言写成的源代码（原始语言），转换成另一种编程语言（目标语言）。

它主要的目的是将便于人编写、阅读、维护的高级计算机语言所写作的源代码程序，翻译为计算机能解读、运行的低阶机器语言的程序，也就是可执行文件。编译器将原始程序（source program）作为输入，翻译产生使用目标语言（target language）的等价程序。

本实验通过对 PI/O 文法实现简单的编译器，加深我们队编译过程的理解。

二、实验部分

基本流程



词法分析

Lexical analysis 类

输入： 源程序

输出： SYM,ID,NUM

SYM：存放每个单词的类别，为内部编码的表示形式。

ID：存放用户所定义的标识符的值，即标识符字符串的机内表示。

NUM：存放用户定义的数。

完成的任务：

1. 滤掉单词间的空格。

2. 识别关键字，用查关键字表的方法识别。当单词是关键字时，将对应的类别放在 SYM 中。

3. 识别标识符，标识符的类别为 0，0 放在 SYM 中，标识符本身的值放在 ID 中。

4. 拼数，将数的类别 30 放在 SYM 中，数本身的值放在 NUM 中。

5. 拼由两个字符组成的运算符，如：>=、<=等等，识别后将类别存放在 SYM 中。

语法分析

grammar analysis 类

输入：lexical analysis 得到的 SYM ID NUM

输出：语法生成树

利用递归下降法

程序自动生成 image.gv 脚本，用 graphviz 生成树，imagefinal.png。

语义分析及目标代码生成

grammar analysis 类

输入：lexical analysis 得到的 SYM ID NUM

输出：目标代码

PL/0 编译程序采用一遍扫描的方法，所以语法分析和代码生成都有在 grammar analysis 中完成。工作分为两步：

a）建表

对每个过程（包括主程序，可以看成是一个主过程）的说明对象造名字表。填写所在层次（主程序是 1 层，在主程序中定义的过程是 2 层，随着嵌套的深度增而层次数增大。PL/0 最多允许 4 层），标识符的属性和分配的相对地址等。标识符的属性不同则填写的信息不同。

所造的表放在全程量一维数组 TABLE 中，数组元素为结构体类型数据。LEV 给出层次，DX 给出每层的局部量的相对地址，每说明完一个变量后 DX 加 1。

例如：一个过程的说明部分为：

```
const a=35,b=49;
 var c,d,e;
 procedure p;
   var g;
```

对它的常量、变量和过程说明处理后，TABLE 表中的信息如下：

NAME: a NAME: b NAME: c NAME: d NAME: e NAME: p	KIND: CONSTANT KIND: CONSTANT KIND: VARIABLE KIND: VARIABLE KIND: VAEIABLE KIND: PROCEDURE	LEVEL: 1 LEVEL: 1 LEVEL: 1 LEVEL: 1 LEVEL: 1 LEVEL: 1	VAL:35 VAL:49 ADR: DX ADR: DX+1 ADR: DX+2 ADR:
NAME: g 。 。 。	KIND: VARIABLE 。 。 。	LEVEL: 2 。 。 。	ADR: DX 。 。 。

对于过程名的 ADR 域，是在过程体的目标代码生成后返填过程体的入口地址。

每个过程的相对起始位置初值 DX=3。

B）语句处理和代码生成

对语句逐句分析，语法正确则生目标代码,当遇到标识符的引用则去查 TABLE 表,看是否有过正确的定义，若有则从表中取出相关的信息,供代码生成用。生成的目标代码放在数组 CODE 中。CODE 是一维数组，数组元素是结构体类型数据。PL/0 语言的目标指令是一种假想的栈式计算机的汇编语言，其格式如下：

f	l	a

其中 f 代表功能码，l 代表层次差，a 代表位移量。

目标指令有 8 条：

① LIT：将常数放到运栈顶，a 域为常数。

② LOD：将变量放到栈顶。a 域为变量在所说明层中的相对位置，l 为调用层与说明层的层差值。

③ STO：将栈顶的内容送到某变量单元中。a,l 域的含义与 LOD 的相同。

④ CAL：调用过程的指令。a 为被调用过程的目标程序的入口地址，l 为层差。

⑤ INT：为被调用的过程（或主程序）在运行栈中开辟数据区。a 域为开辟的个数。

⑥ JMP：无条件转移指令，a 为转向地址。

⑦ JPC：条件转移指令，当栈顶的布尔值为非真时，转向 a 域的地址，否则顺序执行。

⑧ OPR：关系和算术运算。具体操作由 a 域给出。运算对象为栈顶和次顶的内容进行运算，结果存放在次顶。a 域为 0 时是退出数据区。

OPR 在 a 不同时的含义：

1 +

2 -

3 *

4 /

5 =

6 #

7 <

8 >

9 <=

10 >=

11 odd

12 read

13 write

解释执行

编译结束后，记录源程序中标识符的 TABLE 表已退出内存，内存中只剩下用于存放目标程序的 CODE 数组和运行时的数据区 S。S 是由解释程序定义的一维整型数组。解释执行时的数据空间 S 为栈式计算机的存储空间。遵循后进先出的规则，对每个过程（包括主程序）当被调用时，才分配数据空间，退出过程时，则所分配的数据空间被释放。

为解释程序定义四个寄存器:

1．I：指令寄存器，存放当前正在解释的一条目标指令。

2．T：栈顶寄存器，每个过程运行时要为它分配数据区（或称为数据段），该数

据区分为两部分。

静态部分：包括变量存放区和三个联系单元。

动态部分：作为临时工作单元和累加器用。需要时临时分配，用完立即释放。栈顶寄存器 T 指出了当前栈中最新分配的单元（T 也是数组 S 的下标）。

3．B：基地址寄存器，指出每个过程被调用时，在数据区 S 中给出它分配的数据段起始地址，也称为基地址。每个过程被调用时，在栈顶分配三个联系单元。这三个单元的内容分别是：

SL：静态链，它是指向定义该过程的直接外过程运行时数据段的基地址。

DL：动态链，它是指向调用该过程前正在运行过程的数据段的基地址。

RA：返回地址，记录调用该过程时目标程序的断点，即当时的程序地址寄存器 P 的值。

具体的过程调用和结束，对上述寄存器及三个联系单元的填写和恢复由下列目标指令完成。

1．INT　0　a

a:为局部量个数加 3

2．OPR　0　0

恢复调用该过程前正在运行过程（或主程序）的数据段的基地址寄存器的值，恢复栈顶寄存器 T 的值，并将返回地址送到指令寄存器 P 中。

3．CAL　l　a

a 为被调用过程的目标程序的入口，送入指令地址寄存器 P 中。

CAL 指令还完成填写静态链，动态链，返回地址，给出被调用过程的基地址值，送入基址寄存器 B 中。

三、总结

通过实现一个实际编译器（PL/0 语言编译器），我对编译阶段（包括词法分析、语法分析、语义分析、中间代码生成等）和编译系统软件结构有了更深刻的理解。