

编译原理作业 (1)

姓名: 陈彦泽 学号: 181250015

评分: _____ 评阅: _____

2020 年 11 月 19 日

请独立完成作业, 不得抄袭。
若得到他人帮助, 请致谢。
若参考了其它资料, 请给出引用。
鼓励讨论, 但需独立书写解题过程。

1 作业 (必做部分)

题目 1 (编译器, 然后呢?)

观看系列视频 [Crash Course Computer Science@bilibili](#):

- $P5 \sim P8$; 总时长约 45 分钟
- 目的: 了解机器语言是如何跑起来的
- 作业: 随便写点什么吧 (要能表明你确实学习了这些视频)

解答:

“根本上, 这些技术都是矩阵层层嵌套, 来存储大量信息, 就像计算机中的很多事情, 底层其实很简单, 让人难以理解的是一层层精妙的抽象, 像一个越来越小的俄罗斯套娃”

这组视频用了不到一个小时的时间, 用很动画的方式复习了一遍计组的内容。实际上里面的思想用上述的一句话就可以概括

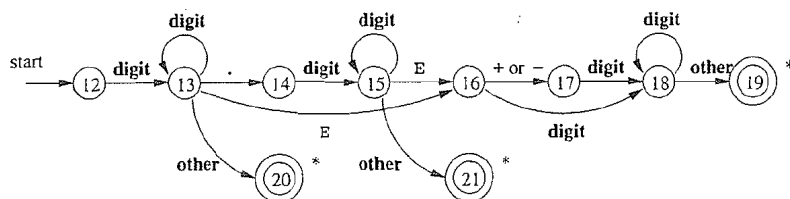
从 ALU 说起, ALU 本质上就是逻辑门的抽象, 用最基本的 AND、OR、NOT、XOR 电路组成了一个一位的加法器, 再用 8 个一位的加法器 (举个例子) 组成了八位的全加器, 最后把各组运算单元进行总体抽象, 抽象成一个 ALU, 接受两个输入 InputA、InputB 和操作码 OpCode, 输出 Ouput 并设置 FLAG。

再到存储, 寄存器就是一组锁存器的矩阵抽象, 而锁存器则是利用逻辑门组成门锁的一个抽象。

最后是 CPU, CPU 根据时钟周期自动地从内存中取指令、解码、执行的过程利用到了先前抽象的 ALU, 与存储的交互中直接与存储抽象的地址线和使能线进行交互, 可以看作是对 ALU 和寄存器、时钟等的一个抽象。

题目 2 (手写词法分析器)

根据下面的状态转移图以及课上介绍的识别方法, 给出识别数字 (正整数、不带科学计数法的浮点数以及带科学计数法的浮点数) 的伪代码。(推荐使用 \LaTeX `algorithmicx` 包书写伪代码)



例如, 对于输入串 `1.23E+a4.5E6b78.c`, 应该识别出 `1.23`, `4.5E6`, `78`, 并且其余字符均应被视为神秘的未知字符。

解答:

Algorithm 1 识别数字算法

1: function READINT	▷ 定义重复读 digit 转为整数的函数, 供复用
2: $v \leftarrow 0$	
3: while isDight(peek) do	
4: $v \leftarrow 10 * v + toInt(peek)$	
5: $peek \leftarrow read()$	
6: end while	
7: return v	
8: end function	
9: procedure START	▷ start
10: $left \leftarrow readInt()$	▷ 循环读取 digit 到状态 13
11: if peek='.' then	▷ 如果有., 读小数部分
12: $peek \leftarrow read()$	
13: $r \leftarrow readInt()$	
14: else if peek ≠ 'E' then	
15: return INT(l)	▷ other 20
16: end if	
17: if peek='E' then	▷ 读到 E, 接下来读指数部分, 状态 16
18: $peek \leftarrow read()$	
19: if peek='-' then	▷ 如果读到了-, 将符号位 neg 记为 1
20: $peek \leftarrow read()$	
21: $neg \leftarrow 1$	
22: end if	
23: $e \leftarrow readInt()$	▷ 循环读取 digit 到状态 13
24: if neg='1' then	
25: $e \leftarrow -e$	▷ 如果符号位为 1, 取负
26: end if	
27: return FLOAT($l + r/len(r) * pow(10, e)$)	▷ other 19
28: else if peek ≠ 'E' then	
29: return FLOAT($l + r/len(r)$)	▷ other 21
30: end if	
31: end procedure	

题目 3 (正则表达式)

课堂上, 我们提到了下面的正则表达式可以用于识别所有 (二进制表示的)3 的倍数。请证明该结论 (或给出直观的解释)。参考: <https://regex101.com/r/ED4qgC/1>

$$\left(0|1(01^*0)^*1\right)^*$$

解答:

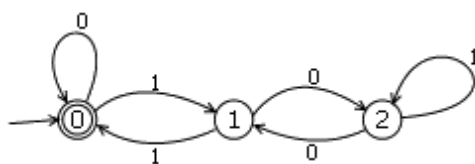
该正则表达式可以通过有限状态自动机进行解释。

因为 mod3 的情况下, 余数的状态有 0、1、2 三种。设定三个状态, 分别叫做 0、1 和 2, 它们表示当前的数除以 3 所得的余数。

因为二进制数左移一位表示十进制中的乘 2, 我们从左往右读二进制串, 每读一位 0 或者 1 进行一次状态转移, 即如果对于某个 i 和 j, 有 $i \cdot 2 \equiv j \pmod{3}$, 就加一条路径 $i \rightarrow j$, 路径上标一个字符 “0” 或者 “1”。

例如, 二进制串 1011, 我们左往右读, 初始状态为 0, 则读入 1 后状态为 1, 表示余数为 1; 读入 0 后状态为 2, 表示余数为 2; 读入 1 后状态为 1, 表示余数为 1; 读入 1 后状态为 1, 表示余数为 1。

构建出的有限状态自动机如图所示:



我们的目标是识别 3 的倍数, 即 mod3 余 0, 也就是表示状态为 0 的所有可能, 既 0 为接受状态。

我们简单枚举从 0 出发, 回到 0 的路径可能:

- 从 0 到 0: 0
- 从 0 到 1 到 0: 11
- 从 0 到 1 到 2 到 1 到 0: 1001
- 因为在状态 2 下可以循环, 改为: $10(1^*)01$
- 因为在状态 12 间可以循环, 改为 $1(0(1^*)0)^*1$

根据 Kleene 构造法, 表示为正则表达式, 即为:

$$\left(0|1(01^*0)^*1\right)^*$$

2 反馈

- 老师真的很用心上课, 爱了爱了!