绪论

计算模型:RAM

There is an infinite set A that is not too big.

- J. von Neumann



deng@tsinghua.edu.cn

Random Access Machine:组成 + 语言

R[0] R[1] R[2] R[3]

- ❖ 寄存器顺序编号,总数没有限制
- 可通过编号直接访问任意寄存器
- 每一基本操作仅需常数时间

$$R[i] \leftarrow c$$
 $R[i] \leftarrow R[R[j]$

$$R[i] \leftarrow R[j]$$

IF
$$R[i] = O GOTO #$$

$$R[i] \leftarrow R[R[j]]$$

$$R[R[i]] \leftarrow R[j]$$

IF
$$R[i] = 0$$
 GOTO # IF $R[i] > 0$ GOTO #

//循环及子程序本身非基本操作

$$R[i] \leftarrow R[j] + R[k]$$

$$R[i] \leftarrow R[j] - R[k]$$

STOP

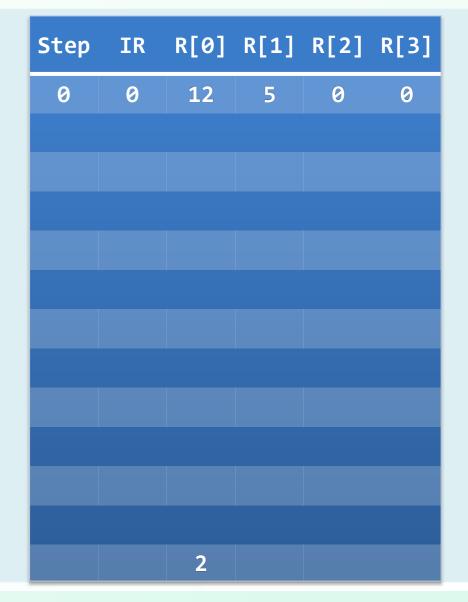
Random Access Machine:效率

- ❖ 与TM模型一样,RAM模型也是一般计算工具的简化与抽象 使我们可以独立于具体的平台,对算法的效率做出可信的比较与评判
- ❖ 在这些模型中
 - 算法的运行时间 ∞ 算法需要执行的基本操作次数
 - T(n) = 算法为求解规模为n的问题,所需执行的基本操作次数
- ❖ 思考:在TM、RAM等模型中衡量算法效率,为何通常只需考查运行时间?空间呢?

实例:Floor Division:思路

$$\lfloor 12/5 \rfloor = 2$$

❖ 思路:反复地从R[0] = 1 + c中,减去R[1] = d
统计在下溢之前,所做减法的次数×



实例:Floor Division:算法

```
0 R[3] <- 1 //increment</pre>
   R[0] \leftarrow R[0] + R[3] //c++
2 R[0] \leftarrow R[0] - R[1] //c -= d
R[2] \leftarrow R[2] + R[3] //x++
4 IF R[0] > 0 GOTO 2 //if c > 0 goto 2
5 R[0] \leftarrow R[2] - R[3] //else x-- and
6 STOP //return R[0] = x = \lfloor c/d \rfloor
```

时间成本 ~ 各条指令执行次数之总和

Step	IR	R[0]	R[1]	R[2]	R[3]
0	0	12	5	0	0
1	1	۸	^	^	1
2	2	13	^	^	^
3	3	8	٨	٨	^
4	4	۸	^	1	^
5	2	۸	^	^	^
6	3	3	^	^	^
7	4	۸	^	2	^
8	2	٨	^	^	^
9	3	0	^	^	^
10	4	٨	^	3	۸
11	5	۸	^	^	۸
12	6	2	^	^	۸