

绪论

计算模型：RAM

当有加减施加拳脚的地万，理性便有了容身之处；而在加减无所适从的地方，理性也就失去了容身之所。

三年之喪，二十五月而畢，哀痛未盡，思慕未忘，然而禮以是斷之者，豈不以送死有已，復生有節也哉！

邓俊辉

deng@tsinghua.edu.cn

Random Access Machine: 组成 + 语言



❖ 寄存器顺序编号，总数没有限制

//但愿如此

❖ 可通过编号**直接访问**任意寄存器

//call-by-rank

❖ 每一**基本操作**仅需常数时间

//循环及子程序本身非基本操作

$R[i] \leftarrow c$ GOTO #

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

$R[i] \leftarrow R[R[j]]$ IF $R[i] = 0$ GOTO #

$R[R[i]] \leftarrow R[j]$ IF $R[i] > 0$ GOTO #

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

$R[i] \leftarrow R[j] - R[k]$ 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** \propto 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：Ceiling Division：思路

- $\spadesuit \quad \forall \quad c \geq 0 \text{ and } d > 0$, define

$$\begin{aligned}\lceil c/d \rceil &= \min\{ x \mid c \leq d \cdot x \} \\ &= \min\{ x \mid c - 1 < d \cdot x \}\end{aligned}$$

❖ 例如: $\lceil 2/7 \rceil = 1$

$$\lceil 35/5 \rceil = 7 \qquad \lceil 2021/43 \rceil = 47$$

$$\lceil 41/7 \rceil = 6 \qquad \lceil 2022/43 \rceil = 48$$

❖ 思路：反复地从 $R[0] = c$ 中，减去 $R[1] = d$

统计在下溢之前，所做减法的次数x

[illegible]

实例：Ceiling Division: 算法

```
[0] R[3] <- 1 //constant increment
[1] GOTO 4 //in case R[0] = c == 0
[2] R[2] <- R[2] + R[3] //x++
[3] R[0] <- R[0] - R[1] //c -= d
[4] IF R[0] > 0 GOTO 2 //if c > 0 goto 2
[5] R[0] <- R[2] //else copy x to R[0] and
[6] STOP //return R[0] = x =  $\lceil c/d \rceil$ 
```

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]	25	12	0	0
1	[1]	^	^	^	1
2	[4]	^	^	^	^
3	[2]	^	^	^	^
4	[3]	^	^	1	^
5	[4]	13	^	^	^
6	[2]	^	^	^	^
7	[3]	^	^	2	^
8	[4]	1	^	^	^
9	[2]	^	^	^	^
10	[3]	^	^	3	^
11	[4]	-11	^	^	^
12	[5]	^	^	^	^
13	[6]	3	^	^	^