

## 01-B-03 RAM (Random Access Machine)

#数据结构邓神

寄存器顺序编号： 总数没有限制（在现实时间无法实现）

$R[0], R[1], R[2], R[3] \dots$

每一个基本操作都只需要常数时间

**RAM: Random Access Machine**

❖ 寄存器顺序编号，总数没有限制 //但愿如此  
 $R[0], R[1], R[2], R[3], \dots$

❖ 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

$R[i] \leftarrow c$	$R[i] \leftarrow R[R[j]]$	$R[i] \leftarrow R[j] + R[k]$
$R[i] \leftarrow R[j]$	$R[R[i]] \leftarrow R[j]$	$R[i] \leftarrow R[j] - R[k]$
IF $R[i] = 0$ GOTO 1	IF $R[i] > 0$ GOTO 1	GOTO 1      STOP

$\leftarrow$  赋值运算符

$R[i] \leftarrow c$  常数赋值语句 将  $c$  赋值到  $R[i]$

$R[i] \leftarrow R[j]$  寄存器赋值到寄存器 将  $R[j]$  赋值到  $R[i]$

$R[i] \leftarrow R[R[j]]$  将寄存器  $R[j]$  中的值作为索引 去访问  $R[\text{索引}]$  的值然后赋值给  $R[i]$

$R[R[i]] \leftarrow R[j]$  与上面反一下

$R[i] \leftarrow R[j] + R[k]$  寄存器加

$R[i] \leftarrow R[j] - R[k]$  寄存器减

IF  $R[i] = 0$  GOTO 1 判断是否为 0

IF  $R[i] > 0$  判断是否为正数

GOTO 跳转语句（无条件）

STOP 终止

与 TM 模型一样，RAM 模型也是一般计算工具的简化和抽象，让我们可以独立于具体的平台，对算法和效率做出可信的比较和判断

在这些模型中

算法的运行时间  $\propto$  算法需要执行基本操作的次数

$T(n)$  = 算法求解规模为  $n$  的问题, 所需要执行的基本操作的次数

## RAM 模型算法实例: Floor

功能: 向下取整的除法 :  $0 \leq c$  ,  $0 < d$

$$\lfloor c/d \rfloor = \max \{ x \mid d \cdot x \leq c \}$$

$$= \max \{ x \mid d \cdot x < 1 + c \}$$

### RAM: Floor

❖ 功能: 向下取整的除法,  $0 \leq c$  ,  $0 < d$

$$\lfloor c/d \rfloor = \max \{ x \mid d \cdot x \leq c \}$$

$$= \max \{ x \mid d \cdot x < 1 + c \}$$

❖ 算法: 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  $x$

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