

1. 绪论

(d) 算法分析

He calculated just as men breathe,
as eagles sustain themselves in the air.

- Francois Arago

邓俊辉

deng@tsinghua.edu.cn

算法分析

- ❖ 两个主要任务 = 正确性（不变性 × 单调性） + 复杂度
- ❖ 为确定后者，真地需要将算法描述为RAM的基本指令，再统计累计的执行次数？
不必！
- ❖ C++等高级语言的基本指令，均等效于常数条RAM的基本指令；在渐进意义下，二者大体相当
 - 分支转向：`goto` //算法的灵魂；出于结构化考虑，被隐藏了
 - 迭代循环：`for()`、`while()`、... //本质上就是“if + goto”
 - 调用 + 递归（自我调用） //本质上也是goto
- ❖ 复杂度分析的主要方法
 - 迭代：级数求和
 - 递归：递归跟踪 + 递推方程
 - 猜测 + 验证

级数

❖ 算数级数：与末项平方同阶

$$T(n) = 1 + 2 + \dots + n = n(n+1)/2 = O(n^2)$$

❖ 幂方级数：比幂次高出一阶：

$$\sum_{k=0}^n k^d \approx \int_0^n x^d dx = \frac{1}{d+1} x^{d+1} \Big|_0^n = \frac{1}{d+1} n^{d+1} = O(n^{d+1})$$

$$T_2(n) = 1^2 + 2^2 + 3^2 + \dots + n^2 = n(n+1)(2n+1)/6 = O(n^3)$$

$$T_3(n) = 1^3 + 2^3 + 3^3 + \dots + n^3 = n^2(n+1)^2/4 = O(n^4)$$

$$T_4(n) = 1^4 + 2^4 + 3^4 + \dots + n^4 = n(n+1)(2n+1)(3n^2+3n-1)/30 = O(n^5)$$

...

❖ 几何级数 ($a > 1$)：与末项同阶

$$T_a(n) = a^0 + a^1 + \dots + a^n = (a^{n+1} - 1)/(a - 1) = O(a^n)$$

$$1 + 2 + 4 + \dots + 2^n = 2^{n+1} - 1 = O(2^{n+1}) = O(2^n)$$

级数

❖ 收敛级数

$$1/1/2 + 1/2/3 + 1/3/4 + \dots + 1/(n-1)/n = 1 - 1/n = \mathcal{O}(1)$$

$$1 + 1/2^2 + \dots + 1/n^2 < 1 + 1/2^2 + \dots = \pi^2/6 = \mathcal{O}(1)$$

$$1/3 + 1/7 + 1/8 + 1/15 + 1/24 + 1/26 + 1/31 + 1/35 + \dots = 1 = \mathcal{O}(1)$$

❖ 有必要讨论这类级数吗？

难道，基本操作次数、存储单元数可能是**分数**？某种意义上！

$$(1-\lambda) \cdot [1 + 2\lambda + 3\lambda^2 + 4\lambda^3 + \dots] = 1/(1-\lambda) = \mathcal{O}(1), \quad 0 < \lambda < 1 \quad // \text{几何分布}$$

❖ 可能未必收敛，然而长度有限

$$h(n) = 1 + 1/2 + 1/3 + \dots + 1/n = \Theta(\log n) \quad // \text{调和级数}$$

$$\log 1 + \log 2 + \log 3 + \dots + \log n = \log(n!) = \Theta(n \log n) \quad // \text{对数级数}$$

❖ 如有兴趣，不妨读读：[Concrete Mathematics](#)

// ex-2.35, Goldbach Theorem

循环 vs. 级数

```
❖ for (int i = 0; i < n; i++)  
    for (int j = 0; j < n; j++)  
        O1Operation(i, j);
```

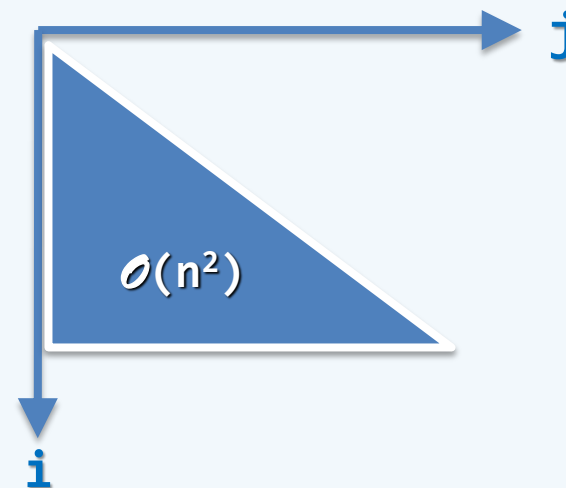
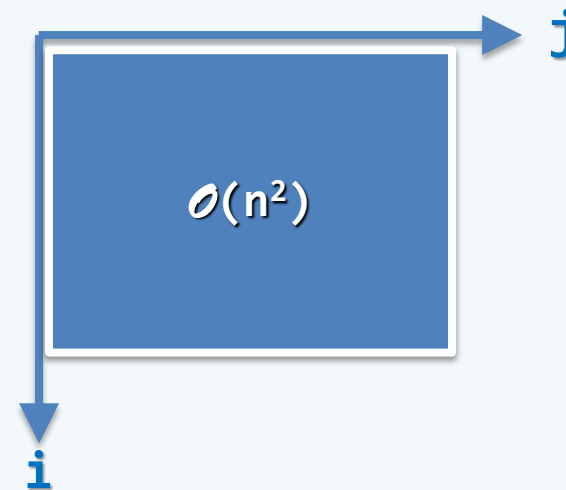
算术级数：

$$\sum_{i=0}^{n-1} n = n + n + \dots + n = n * n = O(n^2)$$

```
❖ for (int i = 0; i < n; i++)  
    for (int j = 0; j < i; j++)  
        O1Operation(i, j);
```

算术级数：

$$\sum_{i=0}^{n-1} i = 0 + 1 + \dots + (n-1) = \frac{n(n-1)}{2} = O(n^2)$$



循环 vs. 级数

```
❖ for (int i = 0; i < n; i++)  
    for (int j = 0; j < i; j += 2013)  
        O1operation(i, j);
```

算术级数： ...

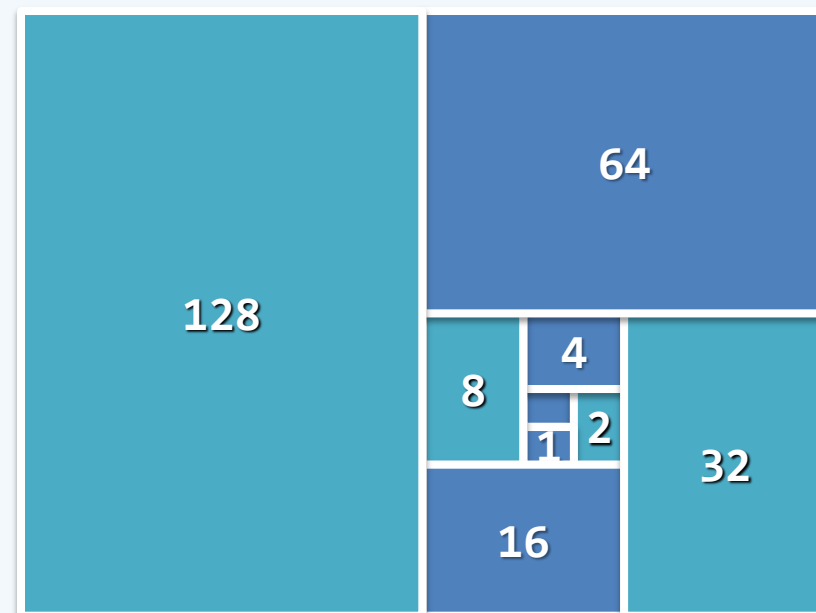
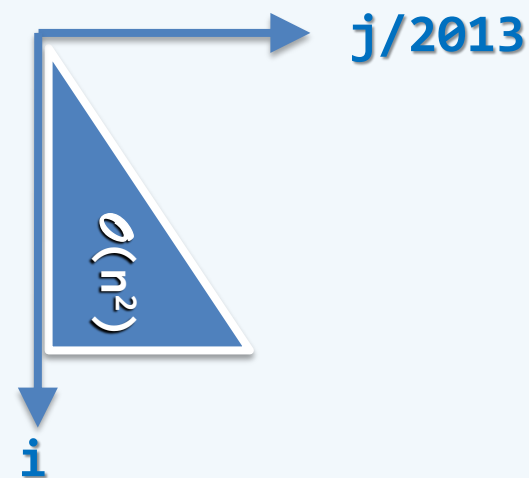
```
❖ for (int i = 1; i < n; i <<= 1)  
    for (int j = 0; j < i; j++)  
        O1operation(i, j);
```

几何级数：

$$1 + 2 + 4 + \dots + 2^{\lfloor \log_2(n-1) \rfloor}$$

$$= \sum_{k=0}^{\lfloor \log_2(n-1) \rfloor} 2^k \quad (\text{let } k = \log_2 i)$$

$$= 2^{\lceil \log_2 n \rceil} - 1 = \mathcal{O}(n)$$



循环 vs. 级数

```
❖ for (int i = 0; i <= n; i++)  
    for (int j = 1; j < i; j += j)  
        operation(i, j);
```

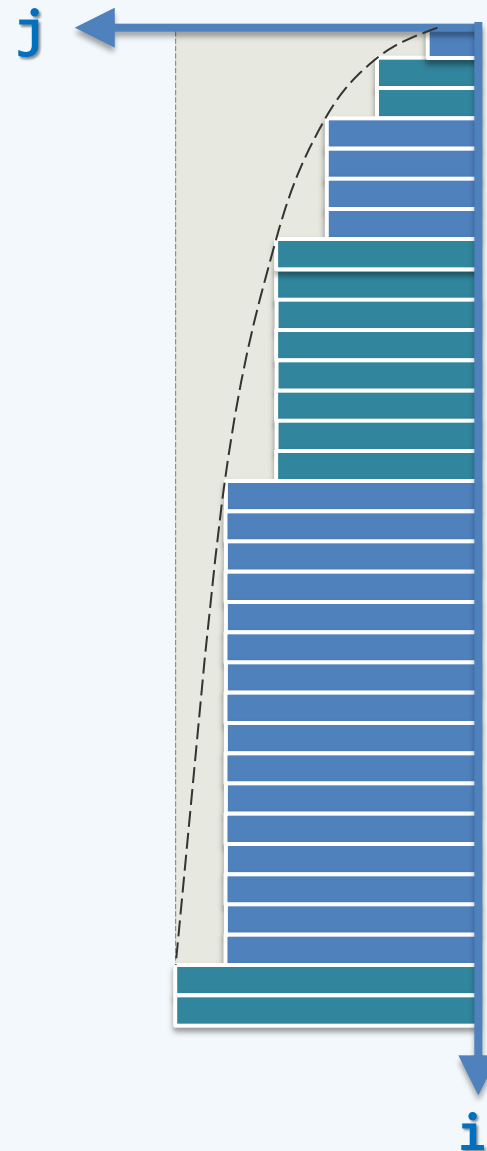
几何级数： $\sum_{k=0}^n \lceil \log_2 i \rceil = \mathcal{O}(n \log n)$

($i = 0, 1, 2, 3 \sim 4, 5 \sim 8, 9 \sim 16, \dots$)

$= 0 + 0 + 1 + 2*2 + 3*4 + 4*8 + \dots$

$= \sum_{k=0.. \log n} (k * 2^{k-1})$

$= \mathcal{O}(\log n * 2^{\log n})$ (CM page#33)



取非极端元素

❖ 问题：给定整数子集 S ， $|S| = n \geq 3$

找出元素 $a \in S$ ， $a \neq \max(S)$ 且 $a \neq \min(S)$

❖ 算法：从 S 中任取三个元素 $\{x, y, z\}$

//若 S 以数组形式给出，不妨取前三个

//由于 S 是集合，这三个元素必互异

确定并排除其中的最小、最大者

//不妨设 $x = \max\{x, y, z\}$ ， $y = \min\{x, y, z\}$

输出剩下的元素 z

❖ 无论输入规模 n 多大，上述算法需要的执行时间都不变

$$T(n) = \text{常数} = O(1) = \Omega(1) = \Theta(1)$$

起泡排序

❖ 问题：给定n个整数，将它们按（非降）序排列

❖ 观察：有序/无序序列中，任意/总有一对相邻元素顺序/逆序

❖ 扫描交换：依次比较每一对相邻元素，如有必要，交换之
若整趟扫描都没有进行交换，则排序完成；否则，再做一趟扫描交换

❖ void bubblesort(int A[], int n) { //第二章将进一步改进

for (bool sorted = false; sorted = !sorted; n--) //逐趟扫描交换，直至完全有序

for (int i = 1; i < n; i++) //自左向右，逐对检查A[0, n)内各相邻元素

if (A[i-1] > A[i]) { //若逆序，则

swap(A[i-1], A[i]); //令其互换，同时

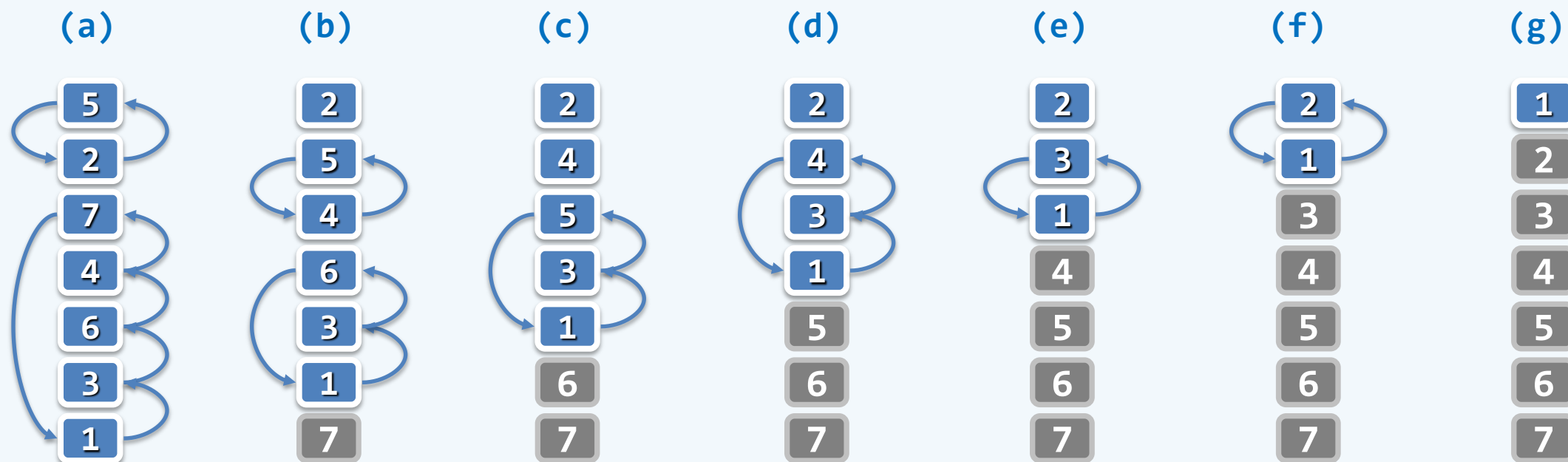
sorted = false; //清除（全局）有序标志

}



起泡排序

- ❖ 问题：该算法必然会结束？至多需迭代多少趟？
- ❖ 不变性：经 k 轮扫描交换后，最大的 k 个元素必然就位
- ❖ 单调性：经 k 轮扫描交换后，问题规模缩减至 $n-k$
- ❖ 正确性：经至多 n 趟扫描后，算法必然终止，且能给出正确解答



起泡排序

❖ 最坏情况：输入数据反序排列

共 $n-1$ 趟扫描交换

每趟的效果，都等同于当前有效区间循环左移一位

第 k 趟中，需做 $n-k$ 次比较和 $3(n-k)$ 次移动， $0 < k < n$

累计：#KMP = $(n-1) + (n-2) + \dots + 1 = n(n-1)/2$

#MOV = $3 \times n(n-1)/2$

$T(n) = 4 \times n(n-1)/2 = O(n^2)$

❖ 最好情况：所有输入元素已经完全（或接近）有序

外循环仅 1 次，做 $n-1$ 次比较和 0 次元素交换

累计： $T(n) = n-1 = \Omega(n)$

Back-Of-The-Envelope Calculation

$$\begin{aligned}\text{❖ 地球 (赤道) 周长} &\approx 787 \times 360 / 7.2 \\ &= 787 \times 50 \\ &= 39,350 \text{ km}\end{aligned}$$

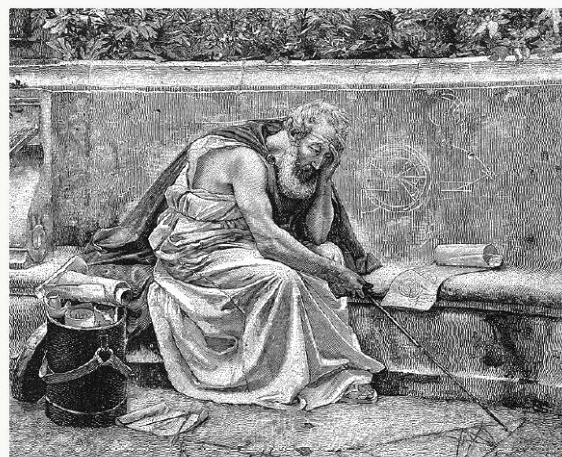
$$\begin{aligned}\text{❖ 1天} &= 24\text{hr} \times 60\text{min} \times 60\text{sec} \\ &\approx 25 \times 4000 = 10^5 \text{ sec}\end{aligned}$$

$$\begin{aligned}\text{❖ 1生} &\approx 1\text{世纪} \\ &= 100\text{yr} \times 365 = 3 \times 10^4 \text{ day} = 3 \times 10^9 \text{ sec}\end{aligned}$$

$$\text{❖ “为祖国健康工作五十年”} \approx 1.6 \times 10^9 \text{ sec}$$

$$\text{❖ “三生三世”} \approx 300 \text{ yr} = 10^{10} = (1 \text{ googol})^{(1/10)} \text{ sec}$$

$$\text{❖ 宇宙大爆炸至今} = 10^{21} = 10 \times (10^{10})^2 \text{ sec}$$



Eratosthenes
(276 ~ 194 B.C.)



Enrico Fermi
(1901 - 1954)

Back-Of-The-Envelope Calculation

❖ 考察对全国人口
普查数据的排序

$n = 10^9 \dots$

普通PC
1GHz
 10^9 flops

天河1A
千万亿次 = 1P
 10^{15} flops

Bubblesort
 $(10^9)^2$
 10^{18}

Mergesort
 $(10^9) \times \log(10^9)$
 30×10^9

10^9 sec
30 yr

10^3 sec
20 min

30 sec

0.03 ms

硬件

算法

课后

- ❖ 试按照“不变性+单调性”的模式，归纳证明本章各算法的正确性
- ❖ 试举例说明，`O1Operation()`对循环体的复杂度也可能有实质影响
- ❖ 学习不同开发环境提供的Profiler工具，并藉此优化你的程序性能
- ❖ 习题[1-32]