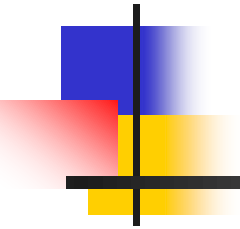


计算机科学导论

孙晓明

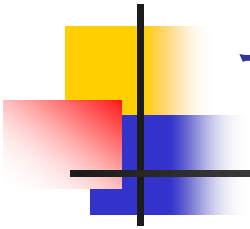
中国科学院计算技术研究所

2022-4-8

问题

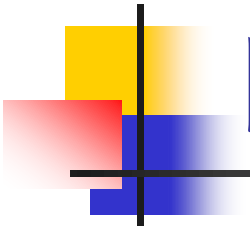
- 下列命题中与$\exists x\ (\forall y\ A(x) \vee B(y))$等价的是____

- 1) $\forall x\ (\exists y\ A(x) \vee B(y))$
- 2) $\forall y\ (\exists x\ A(x) \vee B(y))$
- 3) $(\exists x\ A(x)) \wedge (\forall y\ B(y))$
- 4) $(\exists x\ A(x)) \vee (\forall y\ B(y))$



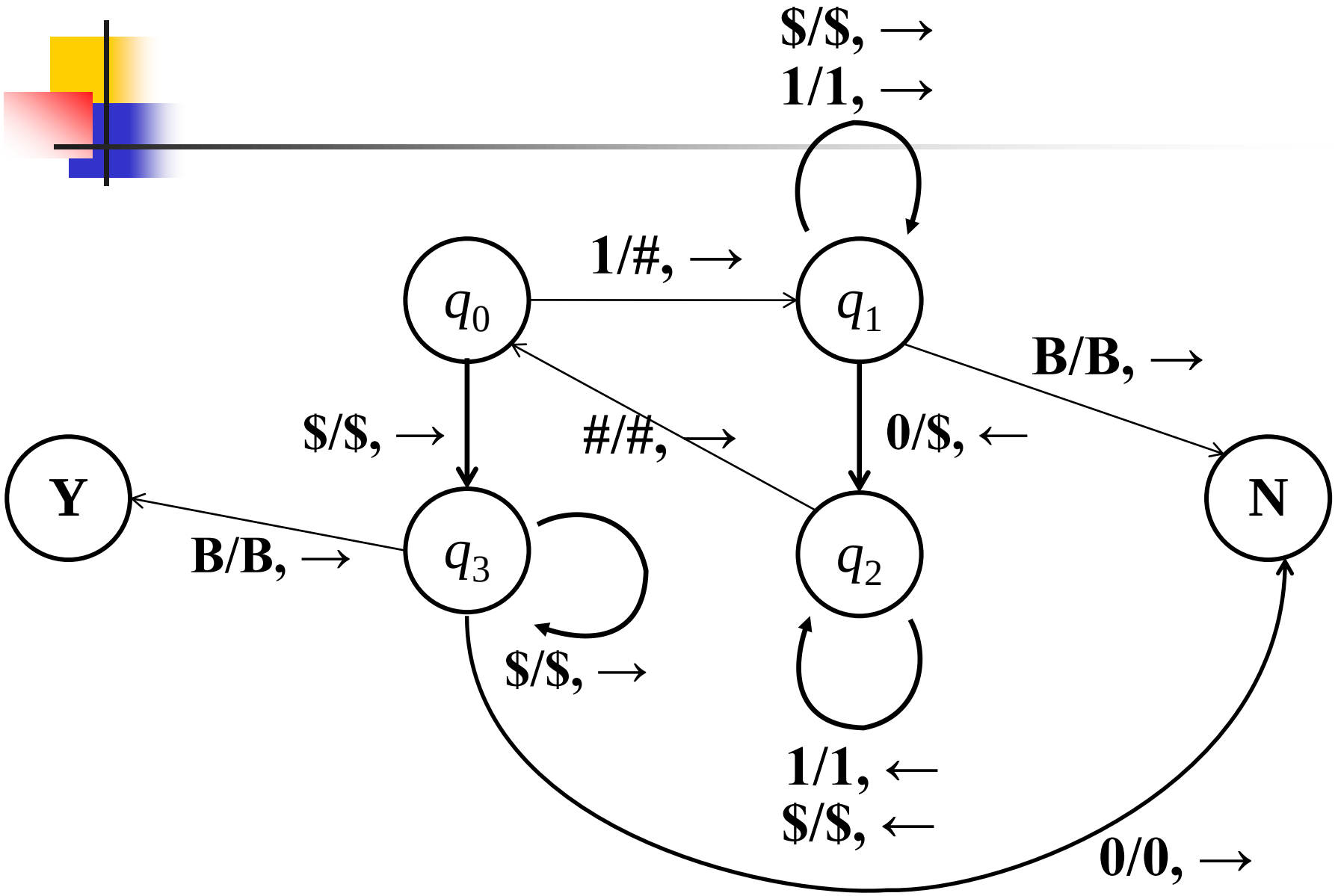
前情回顾

- 命题逻辑
- 谓词逻辑
- 图灵机模型
- 可计算性
  - 停机问题
  - 罗素悖论



图灵机判定相等

- 输入：111…11000…00，判断1和0的个数是否相等？


$/$, →
1/1, →
1/#, →
q_0
q_1
B/B, →
$/$, →
#/#, →
0/$, ←
Y
N
B/B, →
q_3
q_2
$/$, →
1/1, ←
$/$, ←
0/0, →

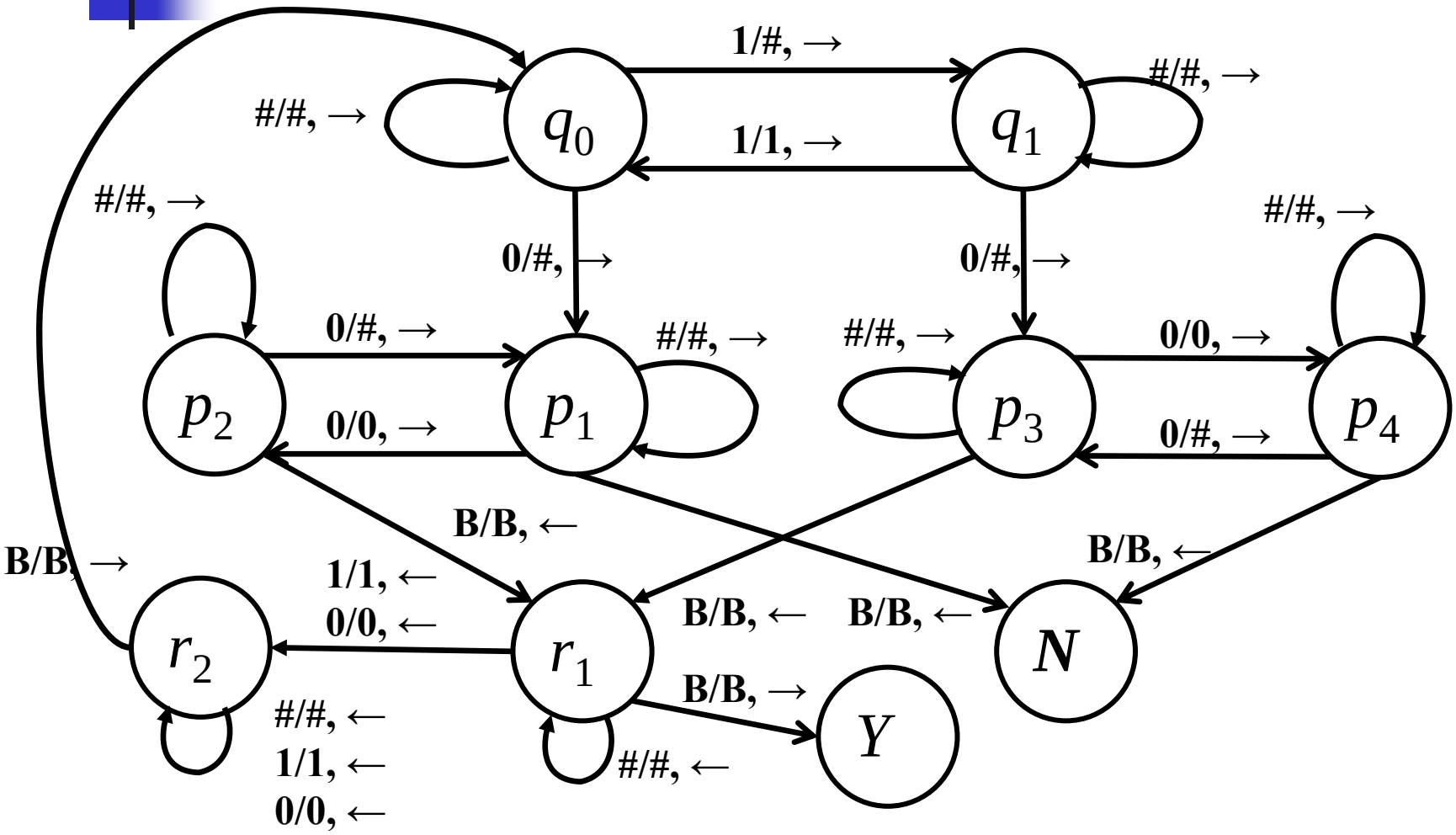
如何改进?

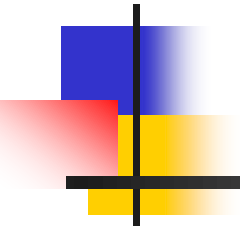
- 上面的图灵机判断 1^n0^n 需要花费~$2n^2$ 的时间，是否能够做的更快？

- Yes.
- Idea: 对于形如输入1^m0^n的输入，将m和n分别写成二进制数

  $m = m_s m_{s-1} \cdots m_1, n = n_t n_{t-1} \cdots n_1$

  然后逐位判定$m_i = n_i$，以及 $s = t$

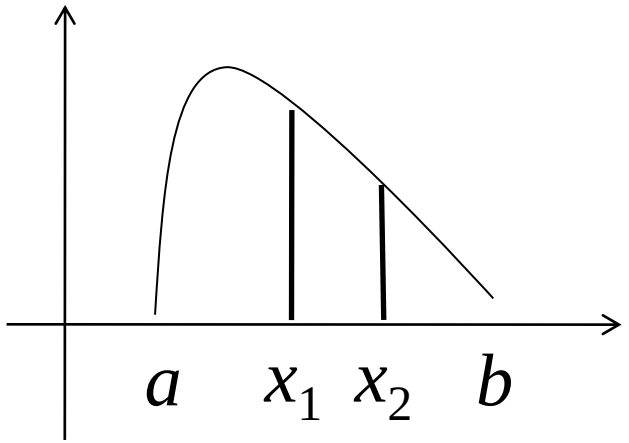

q_0
q_1
p_2
p_1
p_3
p_4
r_2
r_1
N
Y
1/#, →
#/#, →
1/1, →
#/#, →
#/#, →
0/#, →
0/#, →
#/#, →
0/#, →
#/#, →
0/0, →
0/0, →
0/#, →
B/B, ←
B/B, ←
B/B, →
1/1, ←
0/0, ←
B/B, ←
B/B, ←
B/B, →
#/#, ←
1/1, ←
0/0, ←
#/#, ←



算法思维

引子

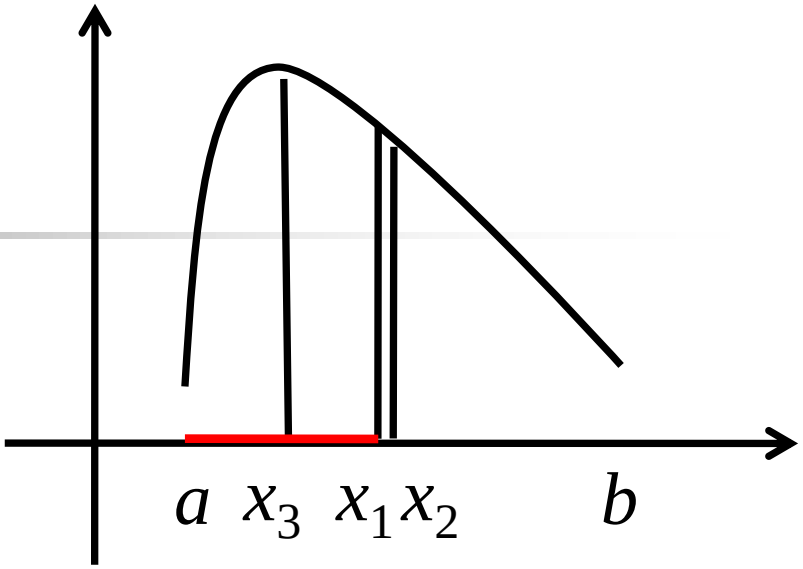
- 单因素优选法: 函数$f(x)$在$[a,b]$上先递增后递减，找出$\max f(x)$.



- idea:
- 选取两个点$x_1 < x_2$
- If $f(x_1) > f(x_2)$, 舍去$[x_2, b]$
- If $f(x_1) < f(x_2)$, 舍去$[a, x_1]$
- If $f(x_1) = f(x_2)$, 同时舍去$[a, x_1]$和$[x_2, b]$

- 方案一：

$x_1 \approx x_2 \approx (a+b)/2$



假定将$[a,b]$离散化成n个点

- $T(n) = T(n/2) + 2$
- $T(2) = 2$

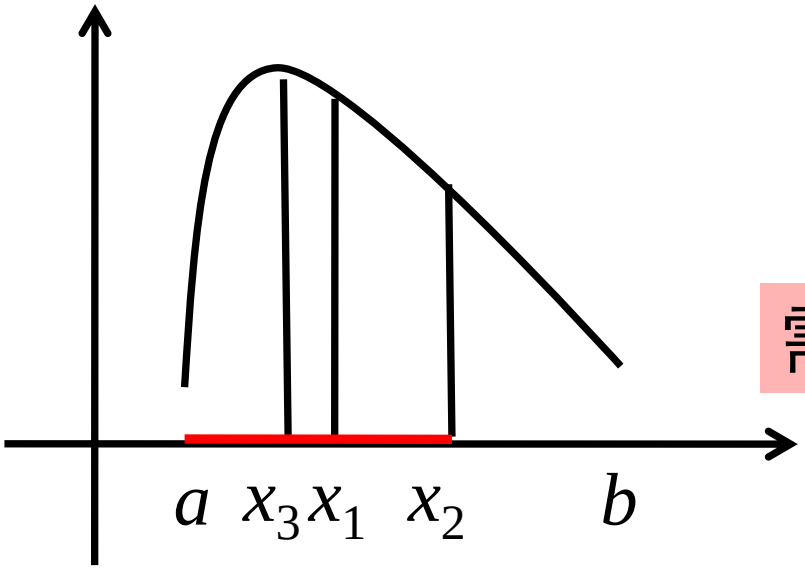
需计算 $2\log_2 n$次函数值

$$T(2^k) = T(2^{k-1}) + 2 = T(2^{k-2}) + 4 = \cdots$$
$$= T(2) + 2(k-1) = 2k$$

- 方案二：

$x_1 = \lambda a + (1-\lambda)b, x_2 = (1-\lambda)a + \lambda b$，其中

$\lambda = \frac{\sqrt{5}-1}{2} \approx 0.618$

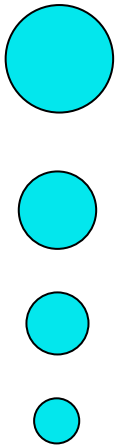


$$T(n) = T(\lambda n) + 1$$

需计算$\log_{1.618} n$次函数值

排序

- 输入：n个正整数，把他们从小到大排成一列

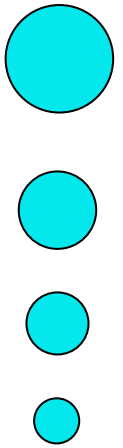


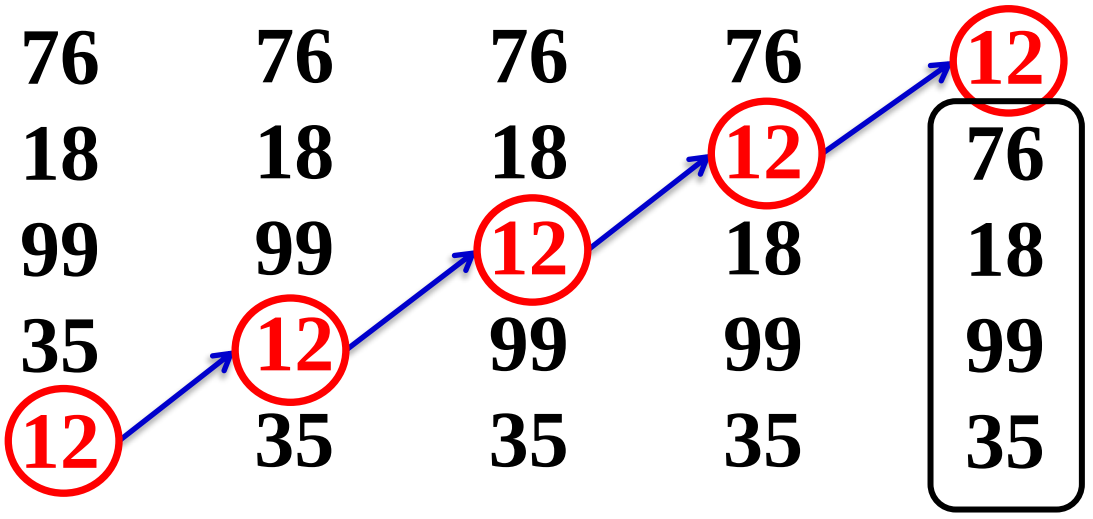
76
18
99
35
12

冒泡排序
（Bubble Sort）

排序

- 输入：n个正整数，把他们从小到大排成一列

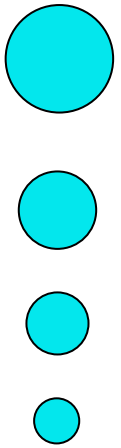


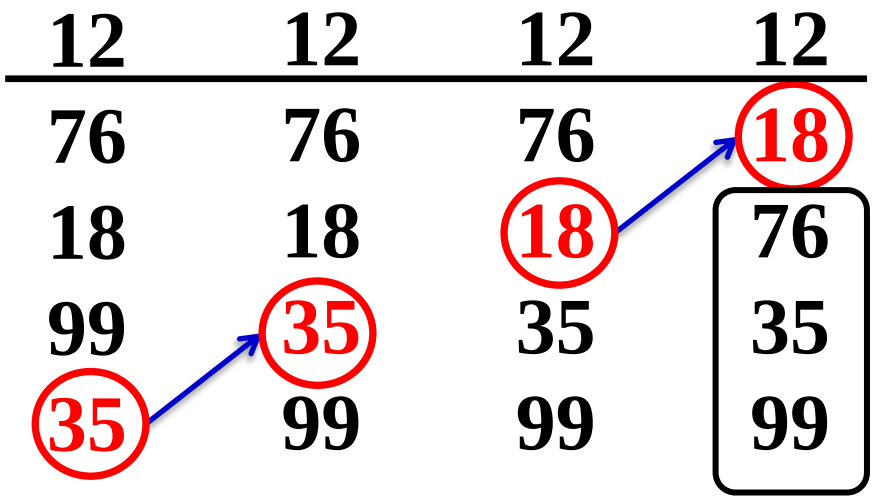


冒泡排序
（Bubble Sort）

排序

- 输入：n个正整数，把他们从小到大排成一列





冒泡排序
（Bubble Sort）

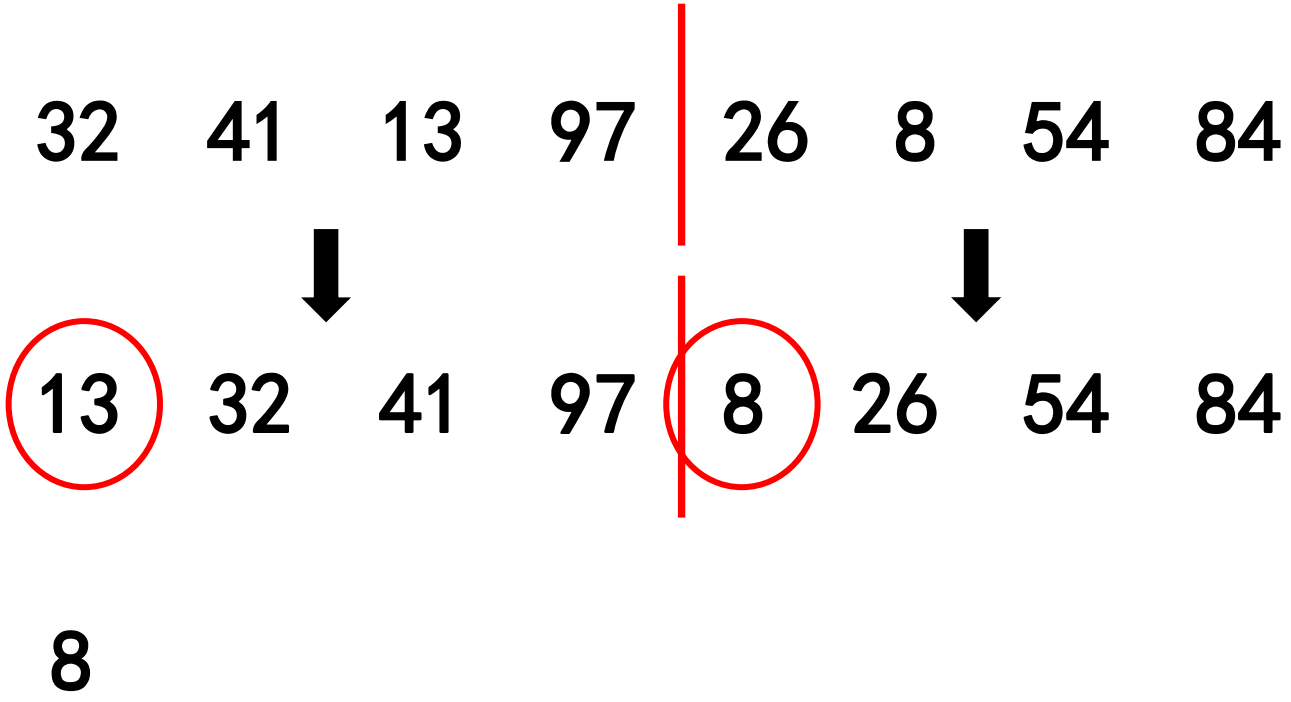
冒泡排序的运行时间

- 总共需要$T(n) = (n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$次比较操作
- 最坏情况下需要$n(n-1)/2$次交换操作
  - $n, n-1, \ldots, 2, 1$
- 能不能更快？

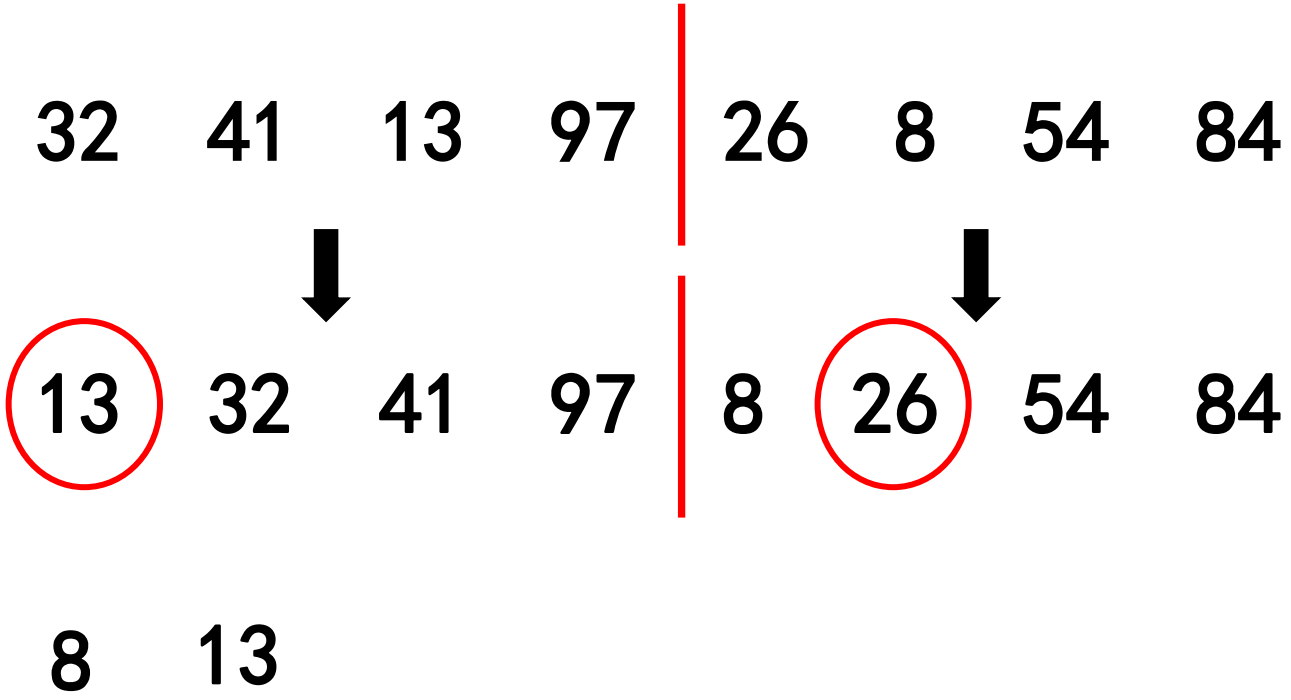
归并排序（Merge Sort）

- 把输入序列分成两个长度为$n/2$的子序列
- 递归调用归并排序分别对两个子序列排序
- 将两个排好序的序列进行归并

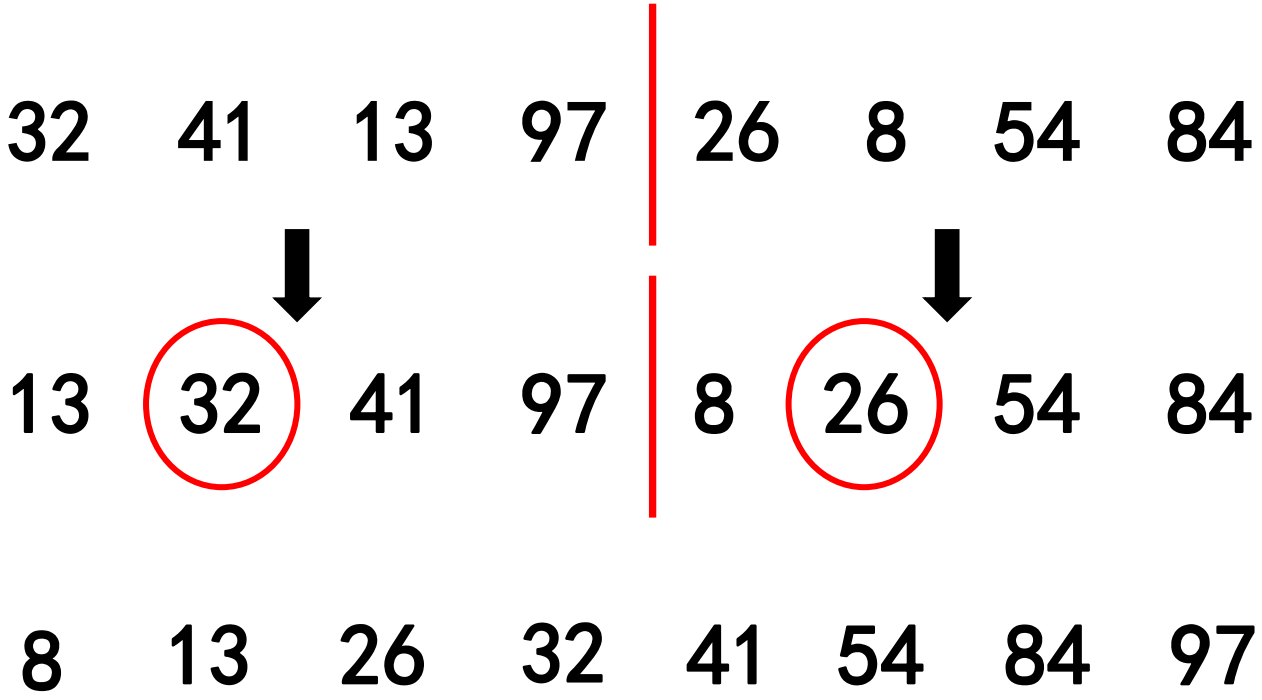
归并排序（Merge Sort）



归并排序（Merge Sort）

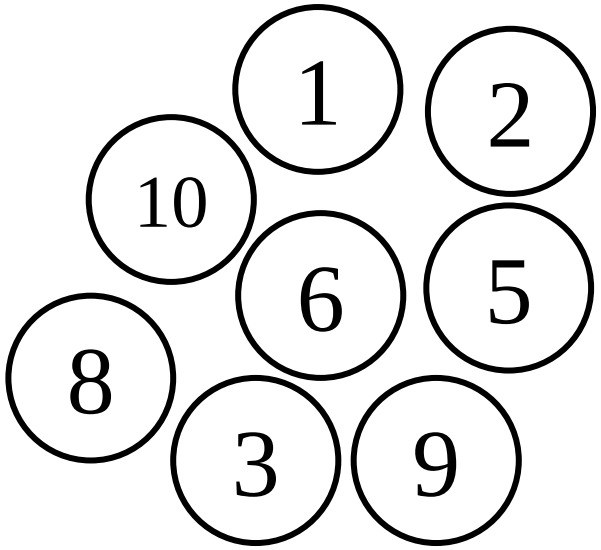


归并排序（Merge Sort）



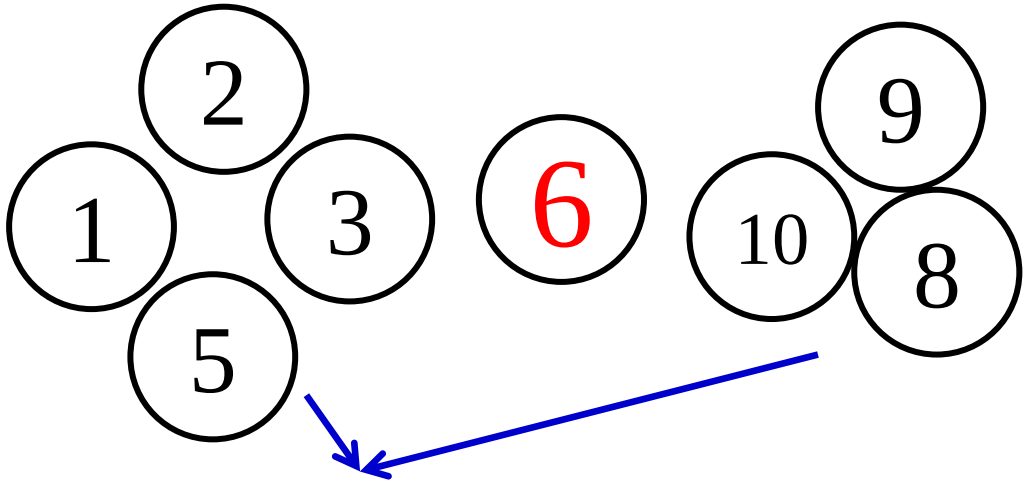
$$T(n) = 2T\left(\frac{n}{2}\right) + n - 1$$

快速排序（Quick Sort）



Step1：随机选其中一个数字

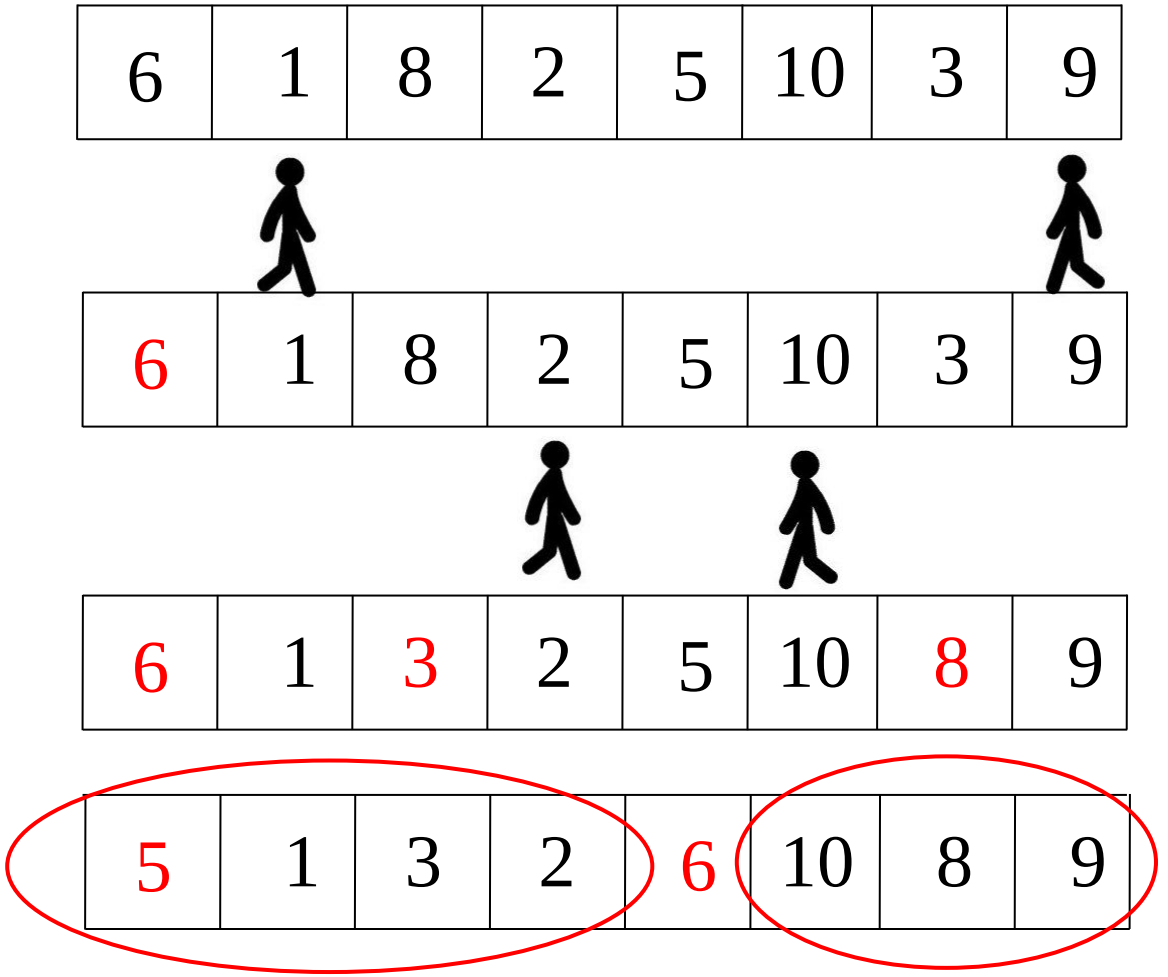
Step2：其他数字按照和基准数的大小关系分成两部分

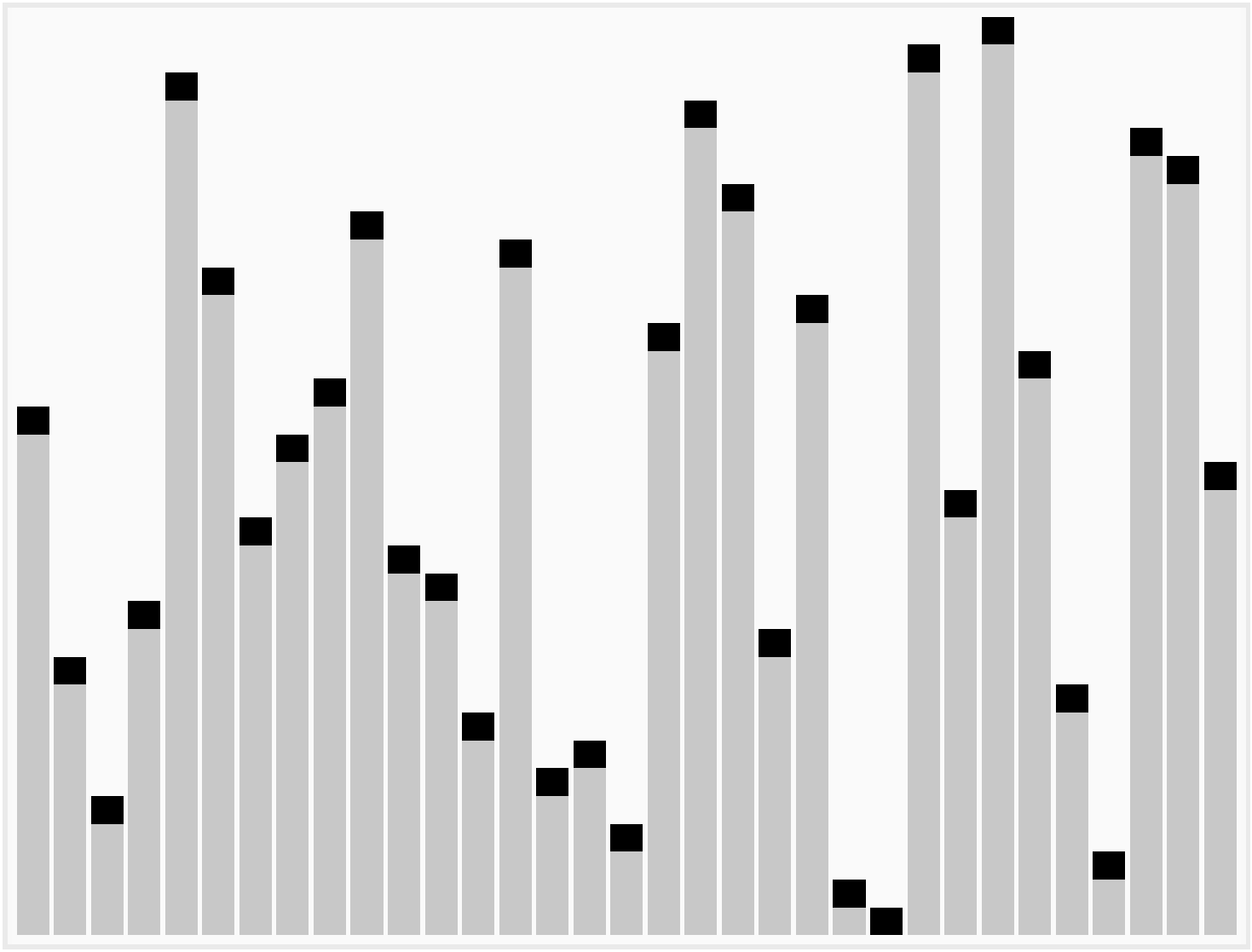


Step3：分别递归

快速排序（Quick Sort）

基准数：6





https://en.wikipedia.org/wiki/Quicksort

快速排序（Quick Sort）

- 需要多少次比较？
  - 最坏情况： $n \times (n-1)/2$
  - 平均情况：？

$$T(n) = \frac{2}{n}\left(T(n-1) + T(n-2) + \cdots + T(1)\right) + n - 1$$

快速排序（Quick Sort）

$$T(n) = \frac{2}{n}\left(T(n-1) + T(n-2) + \cdots + T(1)\right) + n - 1$$

$$T(n+1) = \frac{2}{n+1}\left(T(n) + T(n-1) + \cdots + T(1)\right) + n$$

$$(n+1)T(n+1) - nT(n) = 2T(n) + 2n$$

$$T(n+1) = \frac{n+2}{n+1}T(n) + \frac{2n}{n+1}$$

$$\frac{T(n+1)}{n+2} = \frac{T(n)}{n+1} + \frac{2n}{(n+1)(n+2)}$$

快速排序（Quick Sort）

$$T(n) = \frac{2}{n}\left(T(n-1) + T(n-2) + \cdots + T(1)\right) + n - 1$$

$$\frac{T(n+1)}{n+2} = \frac{T(n)}{n+1} + \frac{2n}{(n+1)(n+2)}$$

$$\frac{T(n)}{n+1} = \frac{T(n-1)}{n} + \frac{2(n-1)}{n(n+1)}$$

$$\frac{T(n)}{n+1} = \frac{T(n-2)}{n-1} + \frac{2(n-2)}{(n-1)n} + \frac{2(n-1)}{n(n+1)}$$

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

小o，大O记号

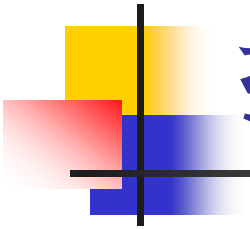
- 冒泡排序
  - 运行时间 $O(n^2)$
- 快速排序
  - 平均运行时间 $O(n \log n)$
  - 是$o(n^2)$

小o大O记号 $\{f(n)\}_{n\geq 1}, \{g(n)\}_{n\geq 1}$

- $f(n) = o(g(n))$ if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$
  - $1000n \log n = o(n^2)$
  - $n^{1000} = o(2^n)$
  - $\log^{200} n = o(n^{0.001})$
- $f(n) = O(g(n))$ if ∃常数$c > 0$,使得 $f(n) \leq cg(n)$ 对充分大的n都成立
  - $1000n \log n = O(n^2)$
  - $\mathbf{10^{1000}\, n = O(n)}$

$\Omega(\cdot), \Theta(\cdot)$记号

- $f(n) = \Omega\big(g(n)\big)$ if ∃常数$c > 0$,使得$f(n) \geq cg(n)$ 对充分大的n都成立
  - $n^2 = \Omega(n \log n)$
  - $0.0001n^3 = \Omega(n^3)$
- $f(n) = \Theta(g(n))$ iff $f(n) = O\big(g(n)\big) \wedge f(n) = \Omega\big(g(n)\big)$
  - $10n^2 - 20n + 45 = \Theta(n^2)$
- 思考： $2^{\Theta(n)}$ 和 $\Theta(2^n)$ 一样吗？



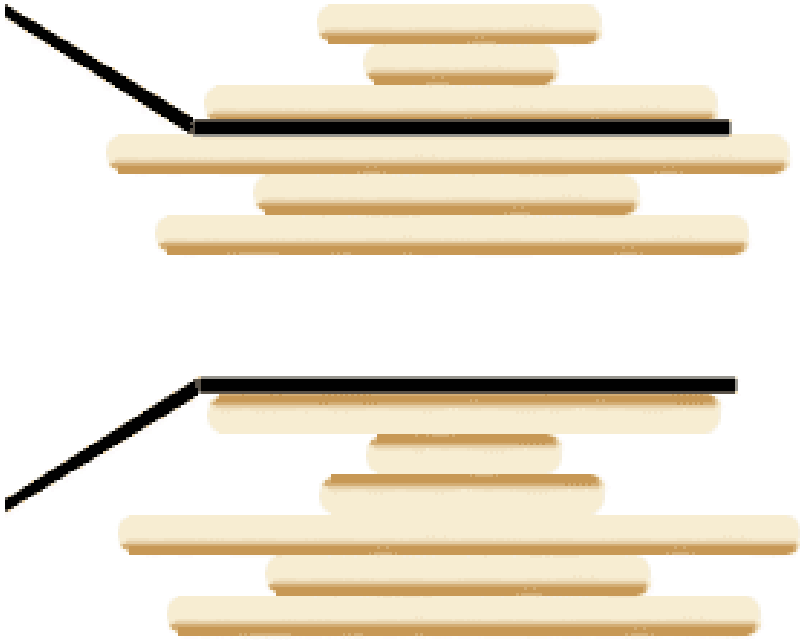
排序算法

- 冒泡排序
- 归并排序
- 快速排序
- 还有更多的排序算法
  - 选择排序、插入排序、堆排序、二叉搜索树、基数排序……

思考题

- 翻煎饼问题(Pancake Sorting)

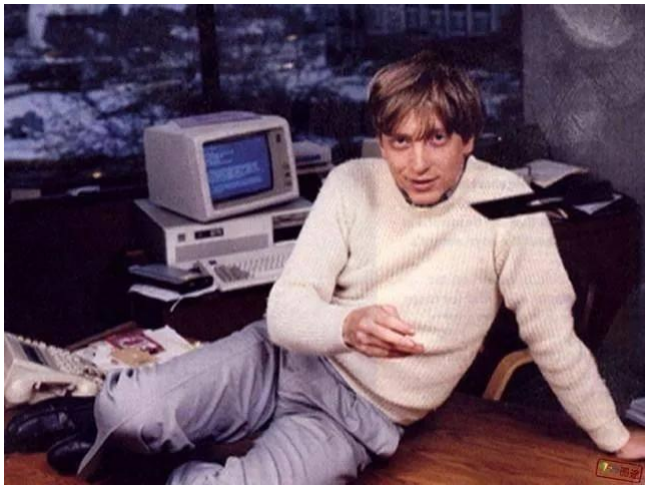
一名厨师做了一叠大小不同的煎饼，他要不断从上面取几个煎饼进行翻面。假设一共有n张煎饼，厨师需要翻动多少次才能把煎饼按从小到大排好？



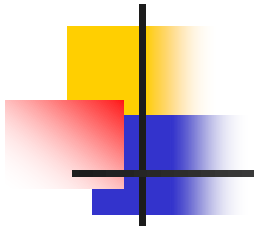
翻煎饼问题

- 2, 1, 4, 5, 3
- 5, 4, 1, 2, 3
- 3, 2, 1, 4, 5
- 1, 2, 3, 4, 5

christos papadimitriou



Bill Gates



谢谢！