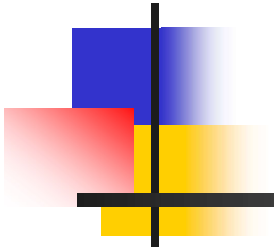
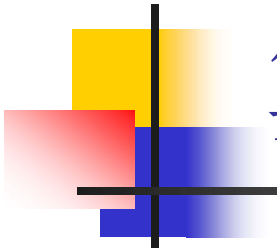


计算机科学导论

张家琳
中国科学院计算技术研究所

zhangjialin@ict.ac.cn

2024-4-19

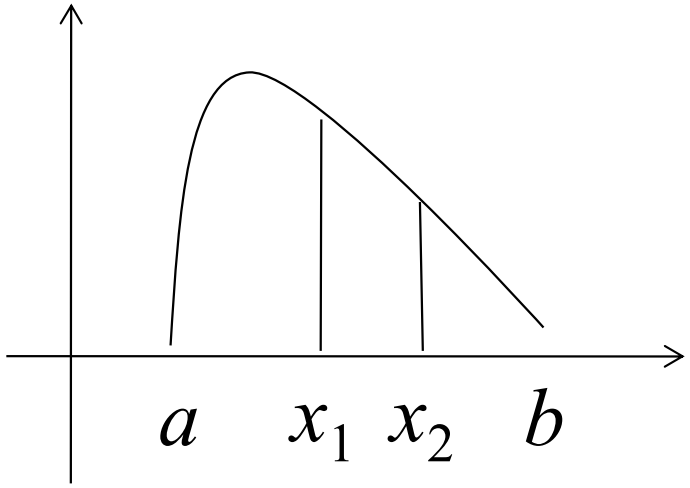


算法思维

- 算法例子：排序
  - 冒泡排序、快速排序
- 大O符号
- 分治思想
- 其他算法实例
- P=NP?问题

分治算法

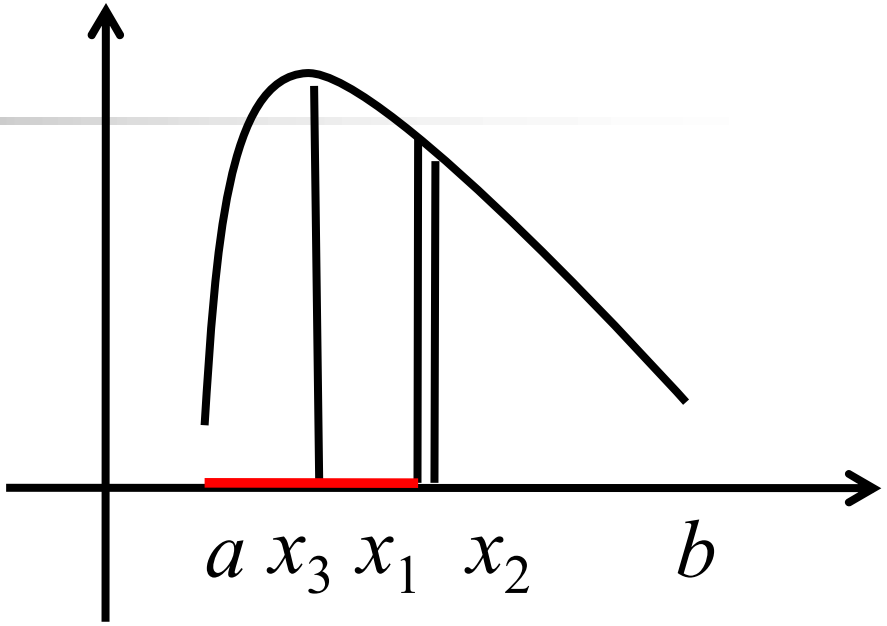
- 单因素优选法: $f(x)$在$[a, b]$上先递增后递减，找出$\max f(x)$.



- idea:
- 选取点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 $f(x_1) = f(x_2)$, 舍去$[a, x_1]$和$[x_2, b]$

- 方案一：

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

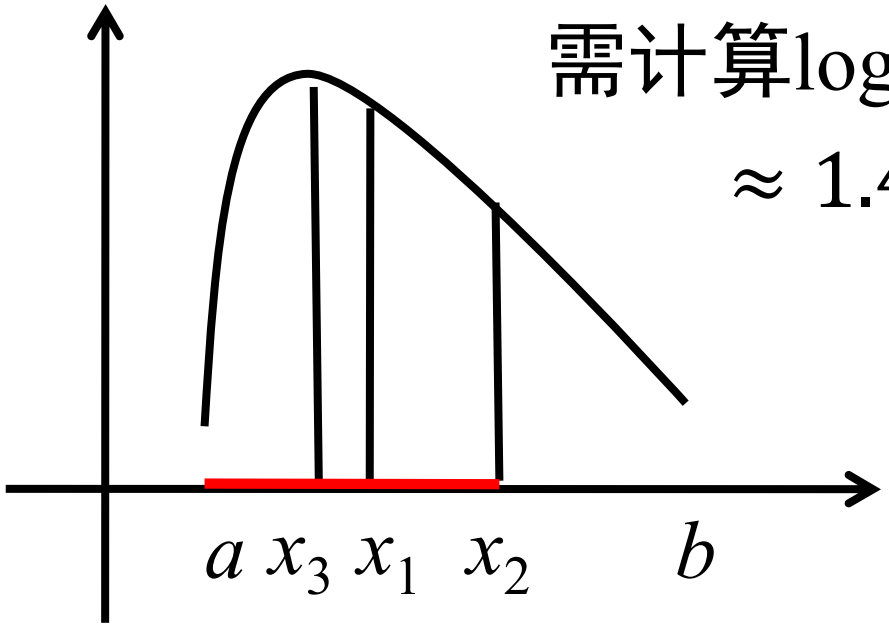


假定将$[a, b]$离散化成n个点

- $\mathrm{T}(n) = \mathrm{T}(n/2) + 2$
- $\mathrm{T}(2) = 2$

需计算 $2\log_2 n$次函数值

- 方案二：$x_1 = ta+(1-t)b$, $x_2 = (1-t)a+tb$，其中 $t = \frac{\sqrt{5}-1}{2} \approx 0.618$



需计算$\log_{1.618} n$次函数值

$\approx 1.44 \log_2 n$

乘法(1)

n^2次乘法运算

- 输入：$X=x_n x_{n-1}\ldots x_1$, $Y=y_n y_{n-1}\cdots y_1$
- 输出：$Z = XY$
- idea:

$$X = X_1 \times 10^{n/2} + X_2,$$

$$Y = Y_1 \times 10^{n/2} + Y_2$$

$$Z = XY =$$

$$X_1Y_1 \times 10^n + (X_1Y_2 + X_2Y_1) \times 10^{n/2} + X_2Y_2$$

$$\begin{array}{r} 123 \\ \times 321 \\ \hline 123 \\ 246 \\ 369 \\ \hline 39483 \end{array}$$

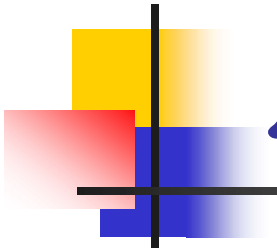
乘法(2)

- 分别计算： $X_1Y_1, X_1Y_2, X_2Y_1, X_2Y_2$
- 乘法次数： $T(n) = 4\,T(n/2), T(1) = 1$
- $T(n) = n^2$...
- $\boxed{X_1Y_1} \times 10^n + \boxed{(X_1Y_2+X_2Y_1)} \times 10^{n/2} + \boxed{X_2Y_2}$
- **计算：** X_1Y_1, X_2Y_2 , $\mathbf{(X_1+X_2)(Y_1+Y_2)}$

$$X_1Y_2+X_2Y_1=\mathbf{(X_1+X_2)(Y_1+Y_2)}-X_1Y_1-X_2Y_2$$

乘法(3)

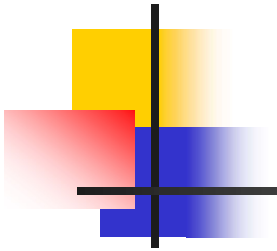
- 乘法次数：$T(n) = 3\ T(n/2),\ T(1) = 1$
- $T(n) = n^{\log_2 3} \approx n^{1.59}$



思考

- 能不能更快？

- 可以！快速傅里叶变换（FFT）
- 通过把X, Y各等分成3段，来演示FFT的思想



$$X = X_2 \times 10^{2n/3} + X_1 \times 10^{n/3} + X_0,$$

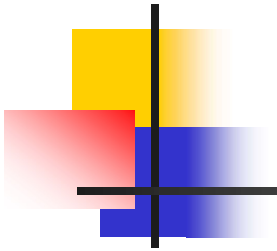
$$Y = Y_2 \times 10^{2n/3} + Y_1 \times 10^{n/3} + Y_0,$$

$$Z = XY =$$

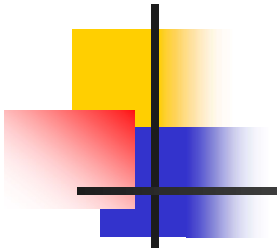
$$X_2Y_2 \times 10^{4n/3} + (X_1Y_2 + X_2Y_1) \times 10^{n}$$

$$+ (X_0Y_2 + X_1Y_1 + X_2Y_0) \times 10^{2n/3}$$

$$+ (X_1Y_0 + X_0Y_1) \times 10^{n/3} + X_0Y_0$$



- 尝试一
  - 分别计算X_0Y_0, X_1Y_1, X_2Y_2, $(X_0+X_1)(Y_0+Y_1)$ $(X_0+X_2)(Y_0+Y_2)$, $(X_1+X_2)(Y_1+Y_2)$
  - $T(n) = 6T(n/3), T(1) = 1$
  - $T(n) = n^{\log_3 6} \approx n^{1.631}$
- 能否用更少的乘法次数实现



$$\omega = e^{i\frac{2\pi}{3}}$$

$$A_0 = X_2 + X_1 + X_0, \qquad B_0 = Y_2 + Y_1 + Y_0,$$

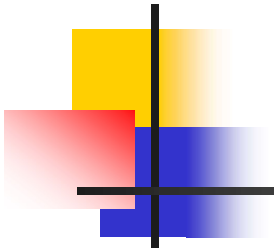
$$A_1 = X_2 + \omega X_1 + \omega^2 X_0, \quad B_1 = Y_2 + \omega Y_1 + \omega^2 Y_0,$$

$$A_2 = X_2 + \omega^2 X_1 + \omega X_0, \quad B_2 = Y_2 + \omega^2 Y_1 + \omega Y_0.$$

$$A_0B_0 + A_1B_1 + A_2B_2 = 3(X_2Y_2 + X_1Y_0 + X_0Y_1)$$

$$A_0B_0 + \omega A_1B_1 + \omega^2 A_2B_2 = 3(X_0Y_2 + X_1Y_1 + X_2Y_0)$$

$$A_0B_0 + \omega^2 A_1B_1 + \omega A_2B_2 = 3(X_2Y_1 + X_1Y_2 + X_0Y_0)$$



- 记两个 n 位数相乘的乘法次数为$\mathrm{T}(n)$

  乘法次数：$\mathrm{T}(n) = 5\ \mathrm{T}(n/3),\ \mathrm{T}(1) = 1$

  $\mathrm{T}(n) = n^{\log_3 5} \approx n^{1.465}$

- 能否更快？
- **快速傅立叶变换(FFT)**

大整数乘法

- 基于快速傅里叶变换
- $O(n \log n \log\log n)$ (Schönhage and Strassen, 1971)
- $O(n \log n \, 2^{O(\log^* n)})$ (Fürer, 2007)
- $O(n \log n \, 8^{\log^* n})$ (Harvey and Hoeven, 2016)
- $O(n \log n \, 4^{\log^* n})$ (Harvey and Hoeven, 2019)
- $O(n \log n)$ (Harvey and Hoeven, 2021)
  - $\log^* n$： https://en.wikipedia.org/wiki/Iterated_logarithm

矩阵乘法(1)

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & \ddots & & \\ \vdots & & \ddots & \\ a_{n1} & & & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & \ddots & & \\ \vdots & & \ddots & \\ b_{n1} & & & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & \ddots & & \\ \vdots & & \ddots & \\ c_{n1} & & & c_{nn} \end{bmatrix}$$

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj}$$

$O(n^3)$次**乘法**， $O(n^3)$次加法

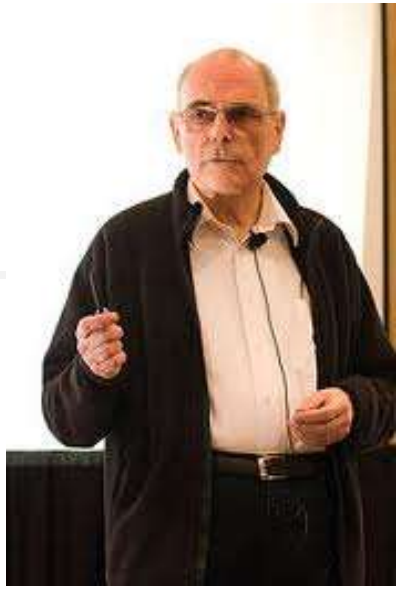
$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \cdot \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

$$m_1 = (a_{12} - a_{22})(b_{21} + b_{22})$$
$$m_2 = (a_{11} + a_{22})(b_{11} + b_{22})$$
$$m_3 = (a_{11} - a_{21})(b_{11} + b_{12})$$
$$m_4 = (a_{11} + a_{12})b_{22}$$
$$m_5 = a_{11}(b_{12} - b_{22})$$
$$m_6 = a_{22}(b_{21} - b_{11})$$
$$m_7 = (a_{21} + a_{22})b_{11}$$

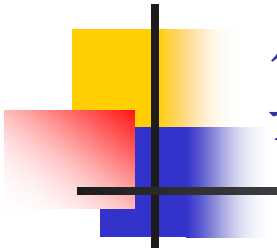
$$c_{11} = m_1 + m_2 - m_4 + m_6$$
$$c_{12} = m_4 + m_5$$
$$c_{21} = m_6 + m_7$$
$$c_{22} = m_2 - m_3 + m_5 - m_7$$

$O(n^{\log 7 \approx 2.81})$次**乘法**

矩阵乘法(3)

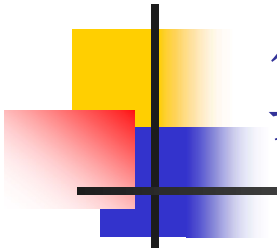


- Strassen algorithm'69 $O(n^{2.81})$
- $O(n^{2.79})$, $O(n^{2.55})$, $O(n^{2.48})$ …
- Coppersmith–Winograd algorithm'89 $O(n^{2.376})$
- Stothers'10 $O(n^{2.374})$
- Williams'11 $O(n^{2.373})$
- Le Gall'14 $O(n^{2.3728639})$
- Alman, Williams'21 $O(n^{2.3728596})$
- Duan, Wu, Zhou'23 $O(n^{2.371866})$
- Williams, Xu, Xu, Zhou'23 $O(n^{2.371552})$



算法思维

- 分治思想
  - 单因素优选法
  - 大整数乘法
  - 矩阵乘法
- 贪心、递归、穷举、回溯、动态规划.....
- 随机算法、近似算法、在线算法、流算法.....
- 后续课程：数据结构、理论计算机基础、算法设计与分析、高等算法.....

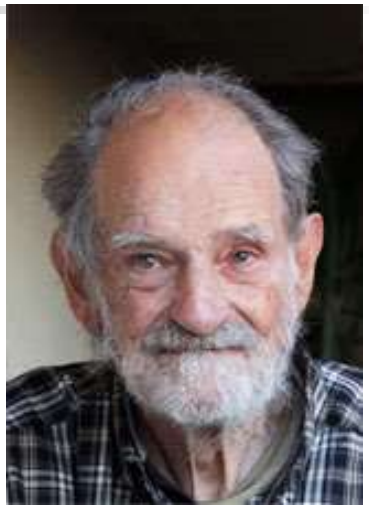


算法思维

- 算法例子：排序
- 大O符号
- 分治思想
- 其他算法实例
- P=NP?问题

2012年经济学诺贝尔奖

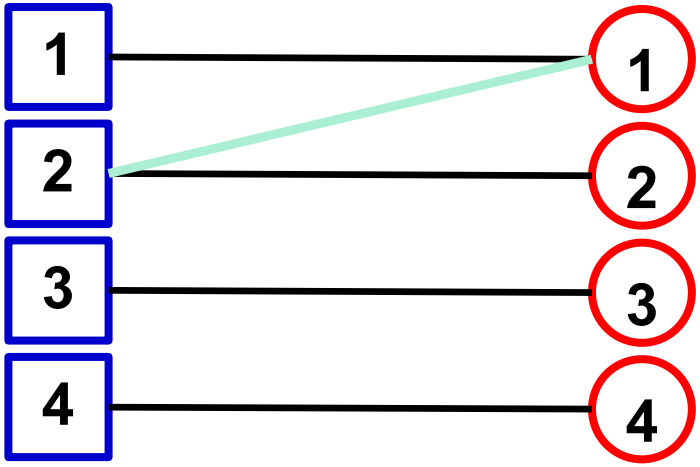
Alvin E. Roth



Lloyd S. Shapley

The Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel 2012 was awarded jointly to Alvin E. Roth and Lloyd S. Shapley "for the theory of stable allocations and the practice of market design"

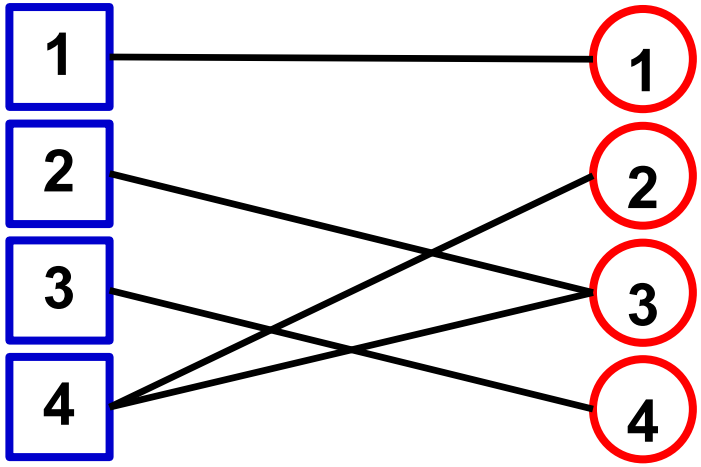
稳定婚姻问题



1	1	3	2	4
2	3	4	1	2
3	4	2	3	1
4	3	2	1	4

1	2	1	3	4
2	4	1	2	3
3	1	3	2	4
4	2	3	1	4

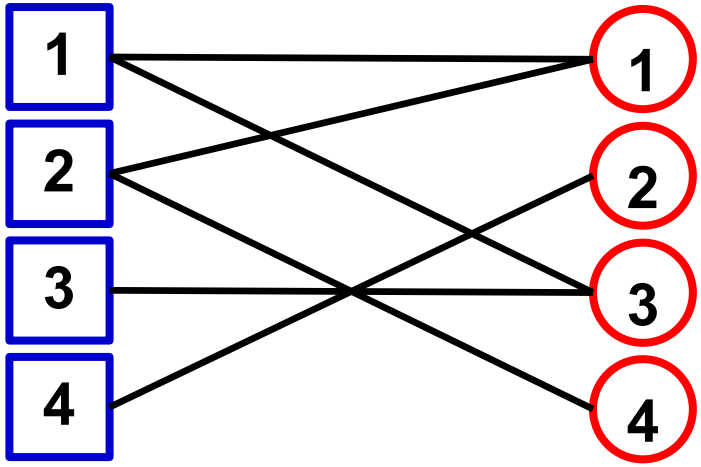
Gale-Shapley Algorithm (1962)



1	1	3	2	4
2	3	4	1	2
3	4	2	3	1
4	3	2	1	4

1	2	1	3	4
2	4	1	2	3
3	1	3	2	4
4	2	3	1	4

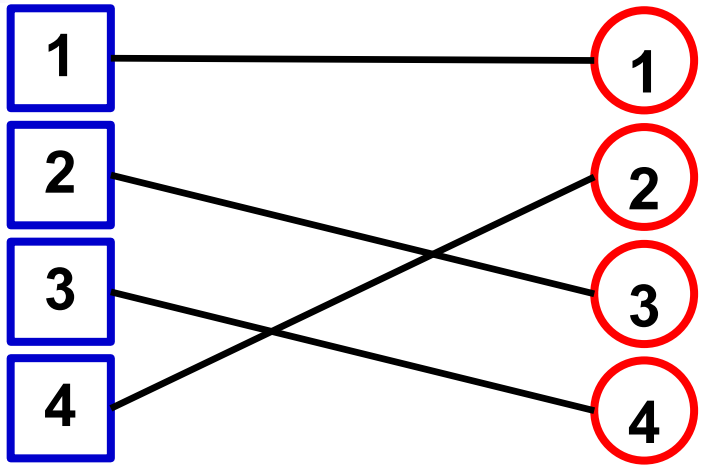
Gale-Shapley Algorithm (1962)

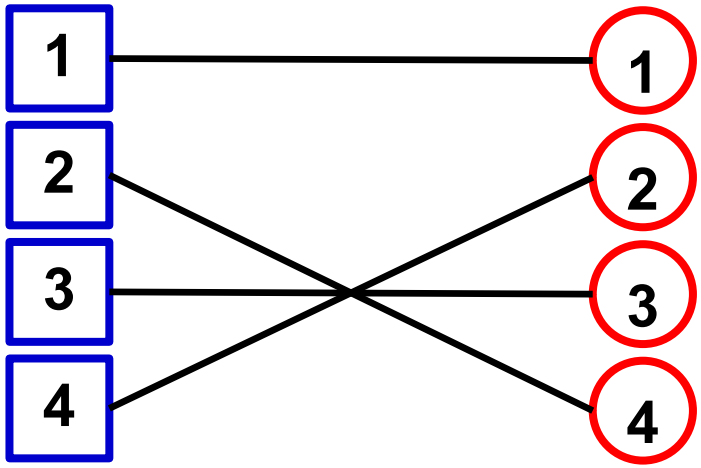


1	1	3	2	4
2	3	4	1	2
3	4	2	3	1
4	3	2	1	4

1	2	1	3	4
2	4	1	2	3
3	1	3	2	4
4	2	3	1	4

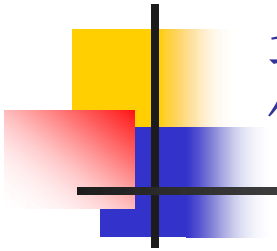
Gale-Shapley Algorithm (1962)





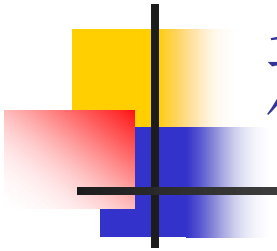
1	1	3	2	4
2	3	4	1	2
3	4	2	3	1
4	3	2	1	4

1	2	1	3	4
2	4	1	2	3
3	1	3	2	4
4	2	3	1	4



稳定婚姻问题

- 稳定婚姻并不唯一
- Gale-Shapley 算法
  - 可以得到一种稳定婚姻匹配
  - 又称 men propose 算法
  - 对求婚者一方有利
    - 思考：为什么？
- 稳定分配的广泛应用



秘书问题

- 又称约会问题、婚姻问题
- 小明想要从n个相亲对象中选择一个结婚
  - 每段时间只能约会一个人（不能骑驴找马）
  - 约会之后分手或者结婚（不能优柔寡断）
  - 分手后不能再回头复合（不能吃回头草）
  - 结婚之后就不能再约了（显然的.....）
- 怎样才能找到最好的对象结婚？

秘书问题-数学假设

- 假设约会顺序是完全随机的，没有任何先验知识
- 每次约会只能知道对方相对之前已经约会过的对象的相对排序
- 目标：找到最好对象的概率最大

A>B　C>A>D>B

C>A>B

E>C>A>D>B

F>E>C>A>D>G>H>B　失败

E>F>C>A>D>G>H>B　成功

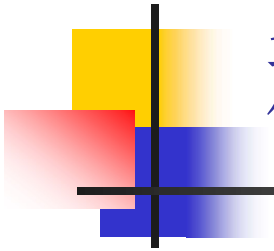
成功概率：1/e

秘书问题-算法

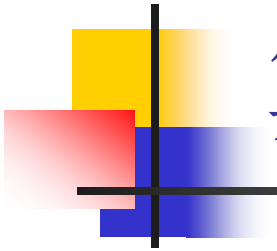
前$\frac{n}{e}$个候选人只看不选择

只要出现当前最好的就选择她

秘书问题-扩展



- 招聘的秘书是前k优秀的概率最大
  - 思考：k=2怎么设计算法？
- 可以招多个秘书
- 招的秘书有组合要求
  - 比如招3个秘书，至少有1个懂财务，1个懂人事
- ……



算法思维

- 算法例子：排序
  - 冒泡排序、快速排序
- 大O符号
- 分治思想
- 其他算法实例
- P=NP?问题
- 思考题：4根柱子的汉诺塔

思考题

- 汉诺塔（Hanoi）
- 如果有4根柱子怎么办？

2^n-1

谢谢！