



《人工智能》

第4讲：不确定性搜索



第4章 不确定性搜索

- 主要内容

- 4.1 基本概念
- 4.2 局部搜索方法
- 4.3 模拟退火算法
- 4.4 本章小结
- 4.5 演化算法

- 参考书目:

- 《人工智能》 朱福喜, 清华大学出版社, 2016. Ch3: pp.30-41, Ch6:pp.77-89.
- 《人工智能: 一种现代的方法(第3版)》 拉塞尔、诺维格, 清华大学出版社, 2011. Ch4: pp. 120-160.





4.1 基本概念

- 旅行商问题

- TSP问题 (*Traveling Salesman Problem*) 又叫推销员问题、货郎担问题。假设有一个旅行商人要拜访 n 个城市，他必须选择所要走的路径，路径的限制是每个城市只能拜访一次，而且最后要回到原来出发的城市。路径的选择目标是要求得的路径路程为所有路径之中的最小值。



典型组合优化问题
重点考虑：时间复杂度



- 搜索（优化）问题的形式化描述

- 设 x 是决策变量， D 是 x 的定义域， $f(x)$ 是指标函数， $g(x)$ 是约束条件集合。

- 则**优化问题**可以表示为，求解满足 $g(x)$ 的 $f(x)$ 最小值问题。

- 如果在定义域 D 上，满足条件 $g(x)$ 的解是有限的，则优化问题称为**组合（离散）优化问题**。

- 符号表示：

$$\min_{x \in D} (f(x) \mid g(x))$$



算法的时间复杂度

- 对于组合优化问题，由于其可能的解是有限的，当问题的规模比较小时，总可以通过**枚举法**获得问题的最优解，但当问题的规模比较大时，就难于求解了。
- 常用的算法复杂度函数

$$O(\log n), O(n), O(n \log n), O(n^2), O(2^n), O(n^{\log n}), O(n!), O(n^n)$$

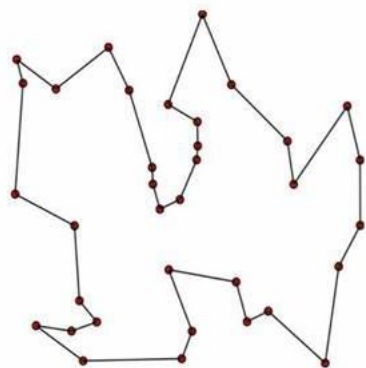
- 时间复杂性函数比较（10亿次/秒）

输入量 n 复杂性函数	10	20	30	40	100
n	10ns	20ns	30ns	40ns	100ns
$n \log n$	10ns	26.0ns	44.3ns	64.1ns	200ns
n^2	100ns	400ns	900ns	1.6us	10us
2^n	1.0us	1.0ms	1.1s	18.3min	4.0世纪
$n!$	3.6ms	77.1年	8.4×10^{13} 世纪	2.6×10^{29} 世纪	3.0×10^{139} 世纪

TSP



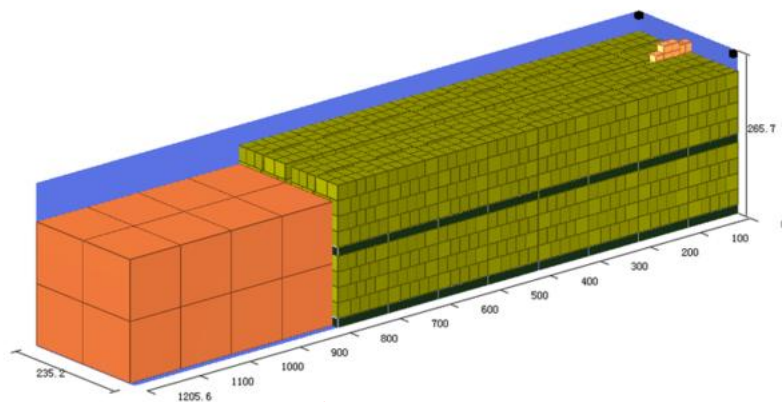
典型组合优化问题



TSP问题



背包问题



装箱问题

七月份排班表

姓名	一	二	三	四	五	六	日	一	二	三	四	五	六	日	一	二	三	四	五	六	日
徐雪松	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休
徐雪松	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休

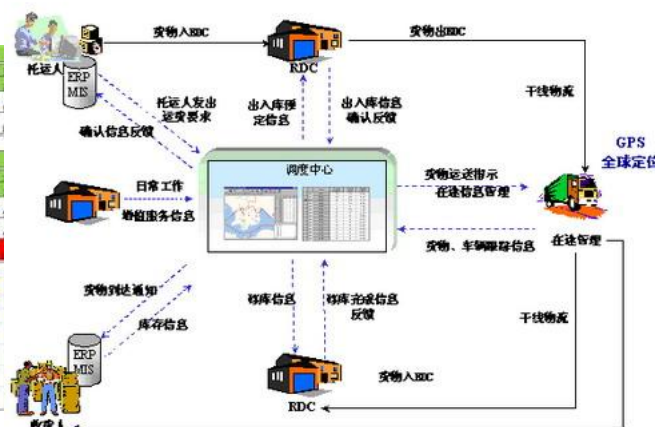
八月份排班表

姓名	一	二	三	四	五	六	日	一	二	三	四	五	六	日	一	二	三	四	五	六	日
徐雪松	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休
徐雪松	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休

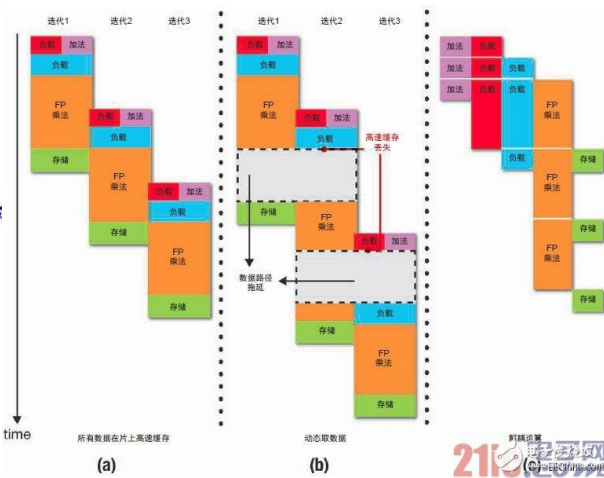
九月份排班表

姓名	一	二	三	四	五	六	日	一	二	三	四	五	六	日	一	二	三	四	五	六	日
徐雪松	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休
徐雪松	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休	休

排班排课问题



物流运输问题



CPU车间调度问题



• 搜索算法

- 搜索算法是利用计算机的高性能来有目的的穷举一个问题解空间的部分或者所有的可能情况，从而求出问题的解的一种计算机算法。

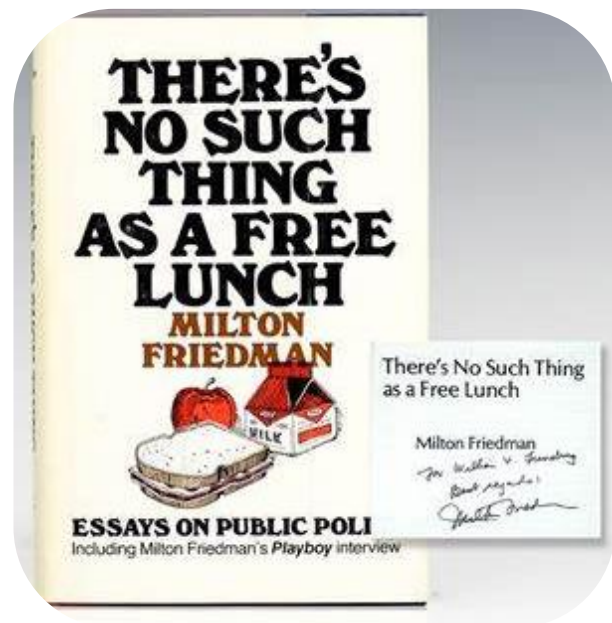
• 算法分类

本章重点

- **确定性算法**：给定初始状态，执行过程唯一确定
 - **不确定性算法**：给定初始状态，每次执行过程不唯一
-
- **构造算法**：算法从0逐渐构造一个完整的（最优）解（无中生有）
 - **逼近算法**：算法从完整的初始解开始，逐渐改进解，最终逼近最优解（优中选优）
-
- **全局最优算法**：算法能保证得到一个全局最优的解
 - **局部最优算法**：算法只能保证得到一个局部最优的解



- 问题脱离算法存在
- 算法依赖于问题
 - 脱离问题存在的算法 -- 无信息搜索算法 -- 算法效率低
 - 利用问题的信息能有效提高算法搜索效率（如A*算法）
- 没有免费的午餐定理(No Free Lunch, 简称NFL)
 - 由于对所有可能问题的相互补偿，最优化算法的**性能是等价**的。（Wolpert、Macerday）
 - 该定理暗指，没有其它任何算法能够比搜索空间的线性列举或者纯随机搜索算法更优。
 - 该定理只是定义在有限的搜索空间，对无限搜索空间结论是否成立尚不清楚。

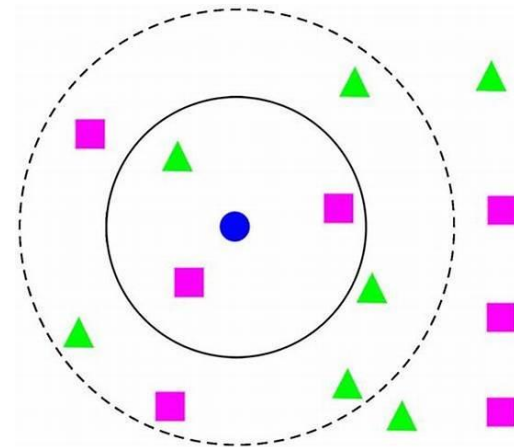


• 邻域 (Neighborhood)

- 简单的说就是一个点 S 附近的其他点的集合
- 在距离空间，邻域就是以点 S 为中心， r 为半径的圆

$$N(S) = \{S' \mid \text{dis}(S, S') \leq r\}$$

- 距离 $\text{dis}(S, S')$ 有多种可能的定义，它是AI的一个专门研究方向



• 组合优化下的邻域

- 设 D 是问题的定义域，若存在一个映射 N ，使得：

$$N : S \in D \rightarrow N(S) \subseteq D$$

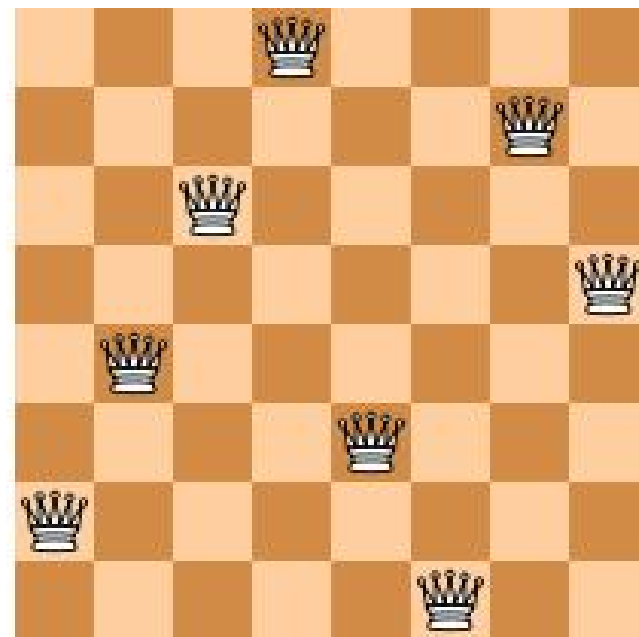
则称 $N(S)$ 为 S 的邻域，表示 D 中部分元素构成的子集。



例：皇后问题

- **解的表示：** $S=\{S_i\}$ 表示一个可能解，其中 S_i 表示在第 i 行，第 S_i 列有一个皇后。
- 如四皇后问题的一个解： $S=(2, 4, 1, 3)$ 。

	Q		
			Q
Q			
		Q	



- **映射N：** 棋盘上任意两个皇后的所在行或列进行交换，即 S 中任意两个元素交换位置。
- 当 $S = (2, 4, 1, 3)$ 时，其邻域为：

$$N(S) = \{(4, 2, 1, 3), (1, 4, 2, 3), (3, 4, 1, 2), (2, 1, 4, 3), (2, 3, 1, 4), (2, 4, 3, 1)\}$$

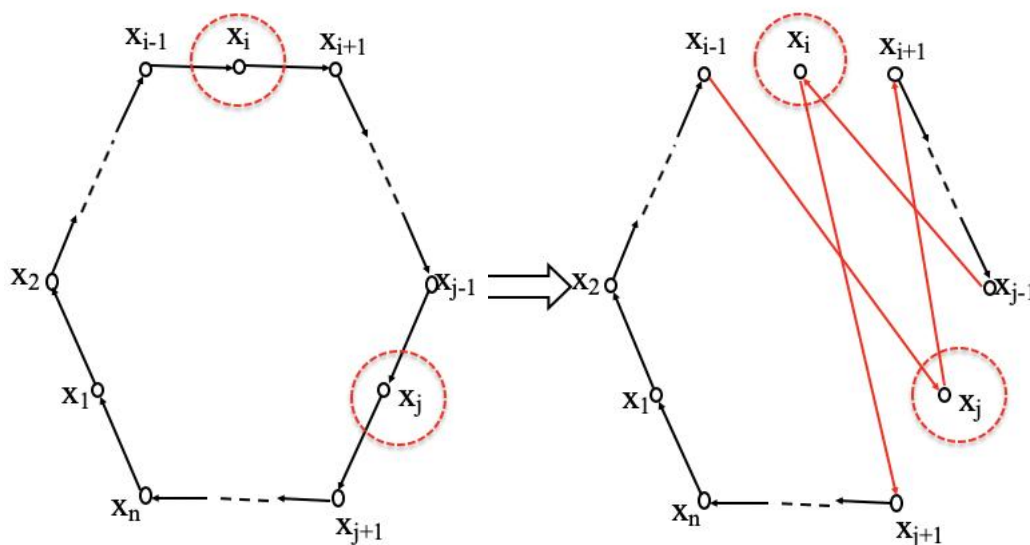


例：旅行商问题

- **解的表示：** 用一个城市的序列表示一个可能的解S。
- **映射N：** 通过交换两个城市的位置获取S的邻居
- **例：简单交换方法**

设 $S = (x_1, x_2, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_{j-1}, x_j, x_{j+1}, \dots, x_n)$ ，交换 x_i 和 x_j 得到：

$$S' = (x_1, x_2, \dots, x_{i-1}, x_j, x_{i+1}, \dots, x_{j-1}, x_i, x_{j+1}, \dots, x_n)$$



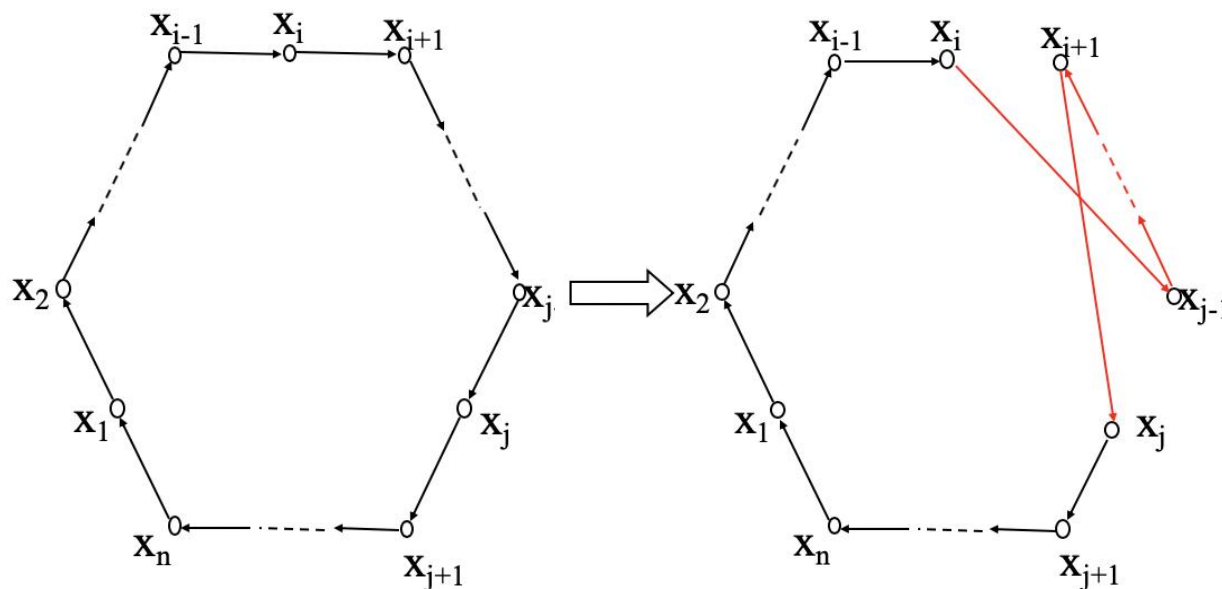


例：旅行商问题

- **映射N：** 逆序交换方法，设 x_i 、 x_j 是选取的两个城市，所谓的逆序交换方式是指，通过逆转 x_i 、 x_j 两个城市之间的城市次序来得到S的邻居。

设 $S = (x_1, x_2, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_{j-1}, x_j, x_{j+1}, \dots, x_n)$

则 $S' = (x_1, x_2, \dots, x_{i-1}, x_i, x_{j-1}, x_{j-2}, \dots, x_{i+1}, x_j, x_{j+1}, \dots, x_n)$





- 邻域与问题解的表示及邻域映射关系定义有关
- 同一个问题，可以有多种解的定义
- 同一种解的定义，可以有多种邻域映射关系
- 算法性能与邻域定义直接相关，最优的解表示及邻域定义能有

效提升算法性能

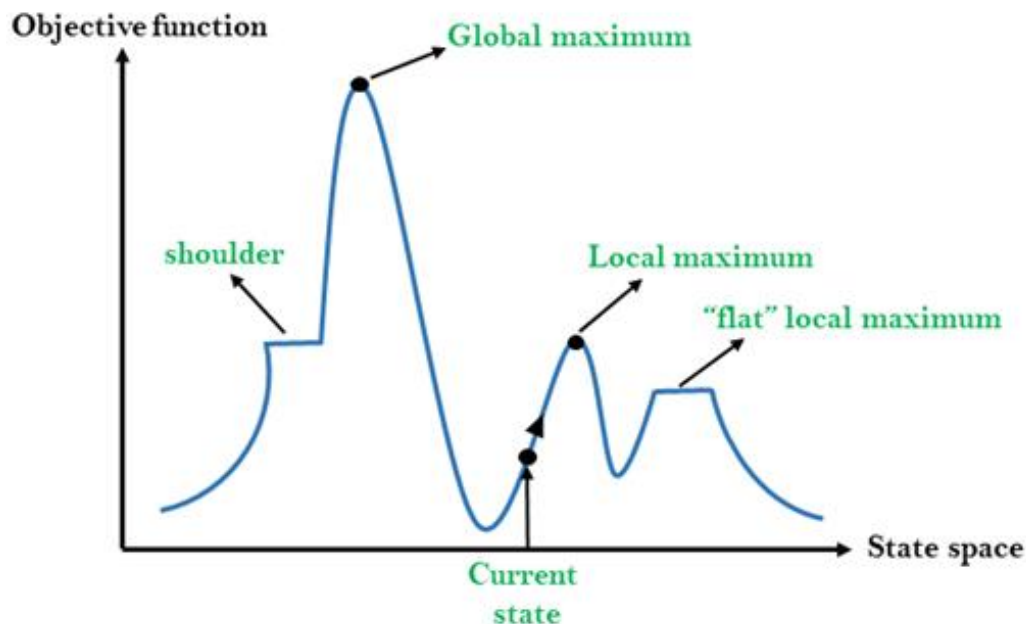
➤ 依赖人类知识与经验

4.2 局部搜索算法

- 基本思想：在搜索过程中，始终向着离目标最接近的方向搜索。
 - 目标可以是最大值，也可以是最小值，无特殊说明，均假定是最小值
 - 在邻域内搜集信息作出判断，前进
 - 反复构建邻域，直到达到局部最优
 - “目标最接近的方向” -> 在邻域内改善最大的方向





局部搜索(爬山法)





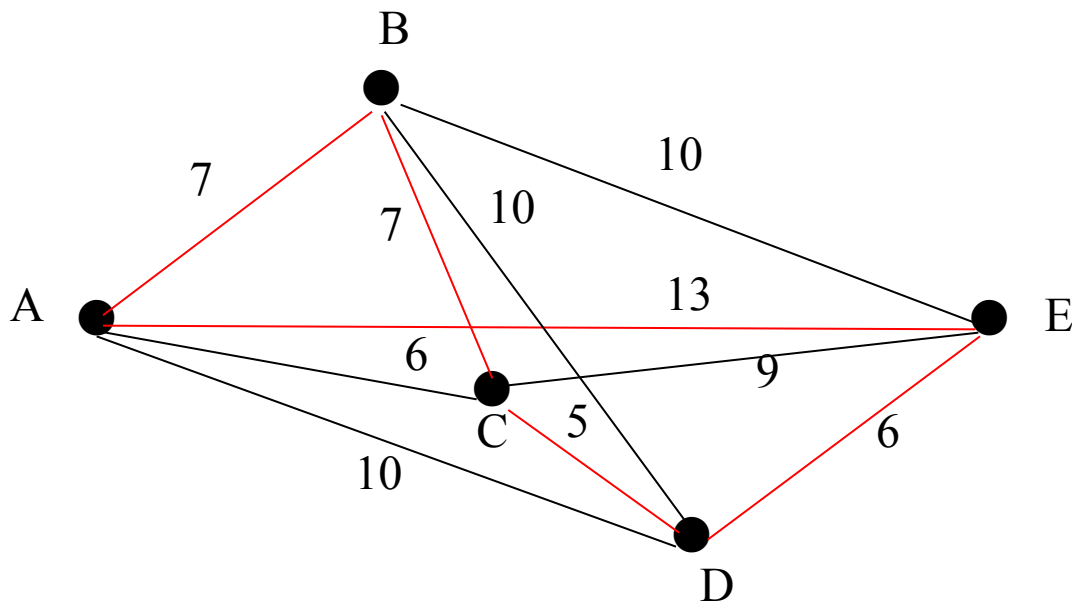
局部搜索算法 (Local Search, LS)

- 1 随机的选择一个初始的可能解 $x_0 \in D$, $x_b = x_0$, $P = N(x_b)$; // 初始化一个解
- 2 如果满足结束条件, 转6;  // 算法停止条件
- 3 选择P的一个子集P', x_n 为P'中的最优解; // 邻域子集构造与比较
- 4 如果 $f(x_n) < f(x_b)$, 则 $x_b = x_n$, $P = N(x_b)$, 转2; // 跳转至更优解
- 5 否则 $P = P - P'$, 转2; // 循环
- 6 输出计算结果。 



例：5城市TSP问题

- 问题描述



- 初始解： $\mathbf{x}_0 = (a, b, c, d, e)$, $f(\mathbf{x}_b) = f(\mathbf{x}_0) = 38$

- 邻域定义：通过交换两个城市获得邻域

$$\mathbf{P}(\mathbf{x}_0) = \{(a, c, b, d, e), (a, d, c, b, e), (a, e, c, d, b), \\ (a, b, d, c, e), (a, b, e, d, c), (a, b, c, e, d)\}$$

- 子集选择策略：每次随机从 \mathbf{P} 中选择一个邻居形成 \mathbf{P}'



第1次循环

从P中选择一个元素,

假设 $x_n = (a, c, b, d, e)$,

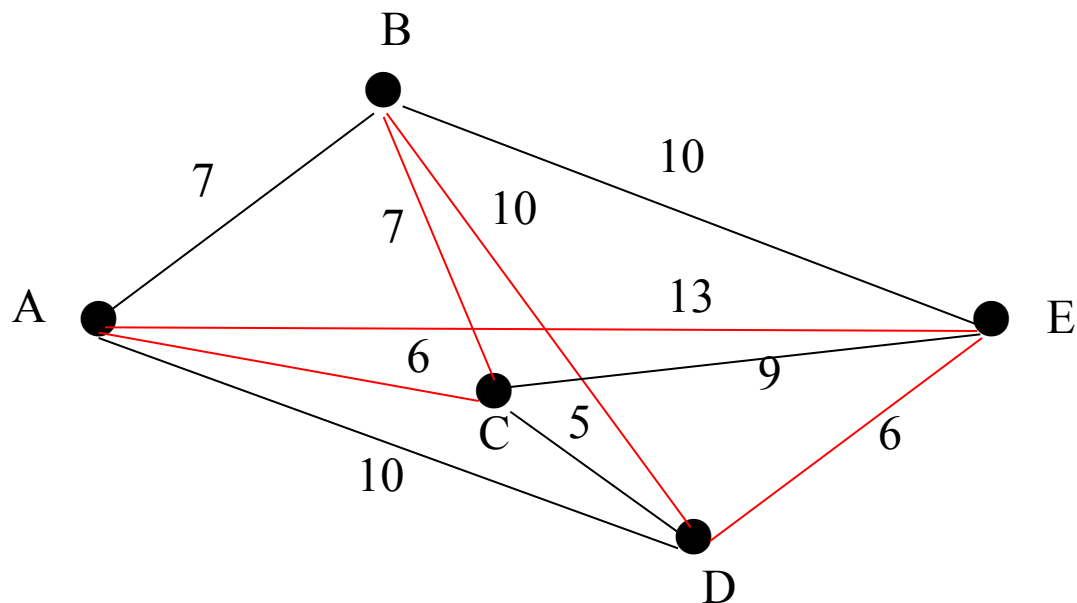
$f(x_n) = 42$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\}$

$= \{(a, d, c, b, e), (a, e, c, d, b), (a, b, d, c, e),$

$(a, b, e, d, c), (a, b, c, e, d)\}$





从P中选择一个元素,

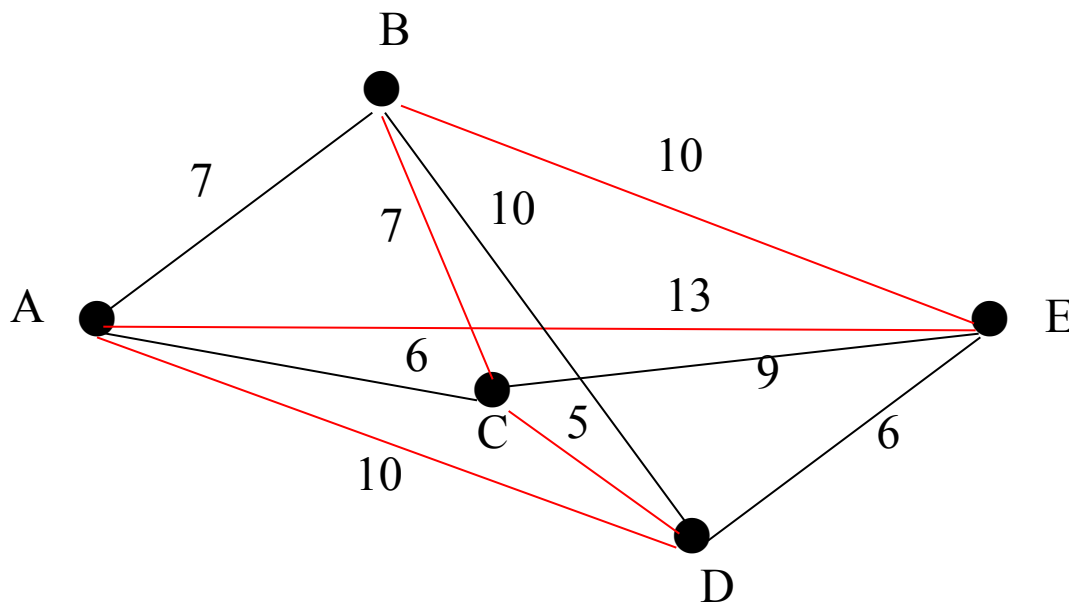
假设 $x_n = (a, d, c, b, e)$,

$f(x_n) = 45$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\}$

$= \{(a, e, c, d, b), (a, b, d, c, e), (a, b, e, d, c), (a, b, c, e, d)\}$





从P中选择一个元素,

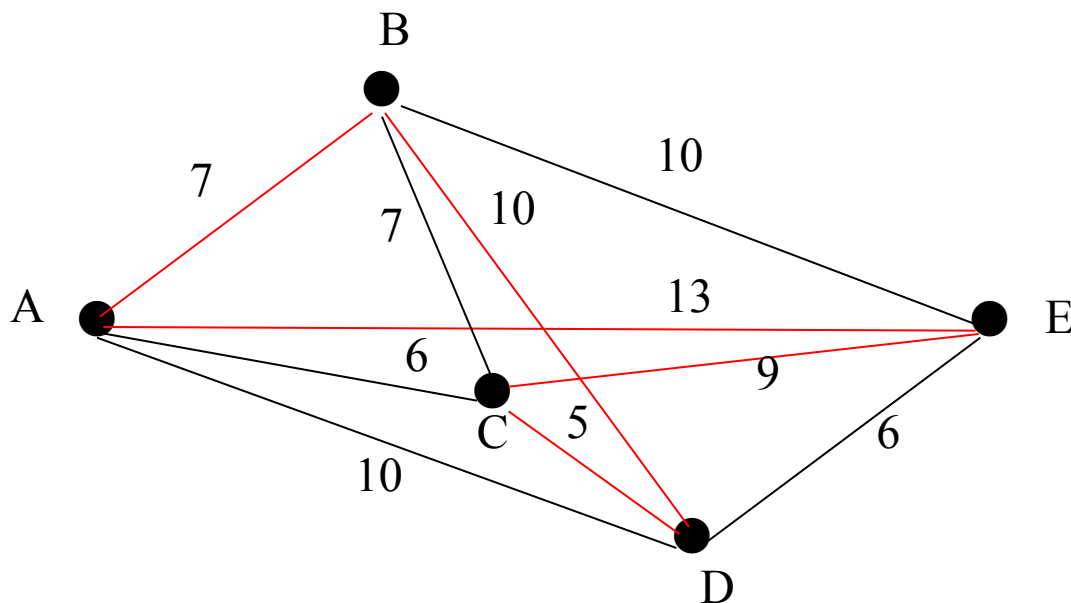
假设 $x_n = (a, e, c, d, b)$,

$f(x_n) = 44$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\}$

$= \{(a, b, d, c, e), (a, b, e, d, c), (a, b, c, e, d)\}$





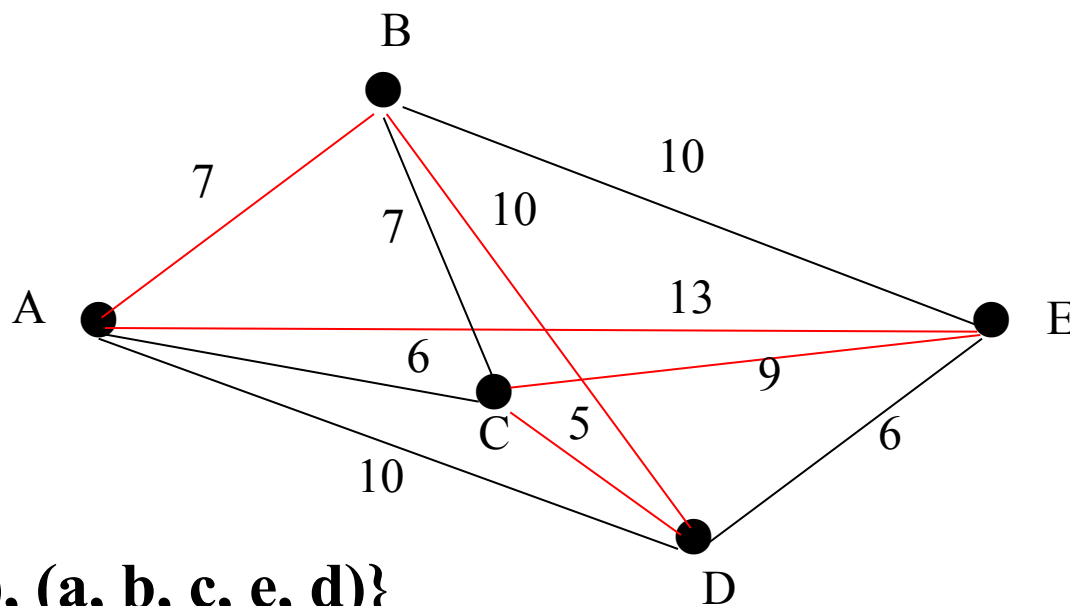
从P中选择一个元素,

假设 $x_n = (a, b, d, c, e)$,

$f(x_n) = 44$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\} = \{(a, b, e, d, c), (a, b, c, e, d)\}$





从P中选择一个元素,

假设 $x_n = (a, b, e, d, c)$,

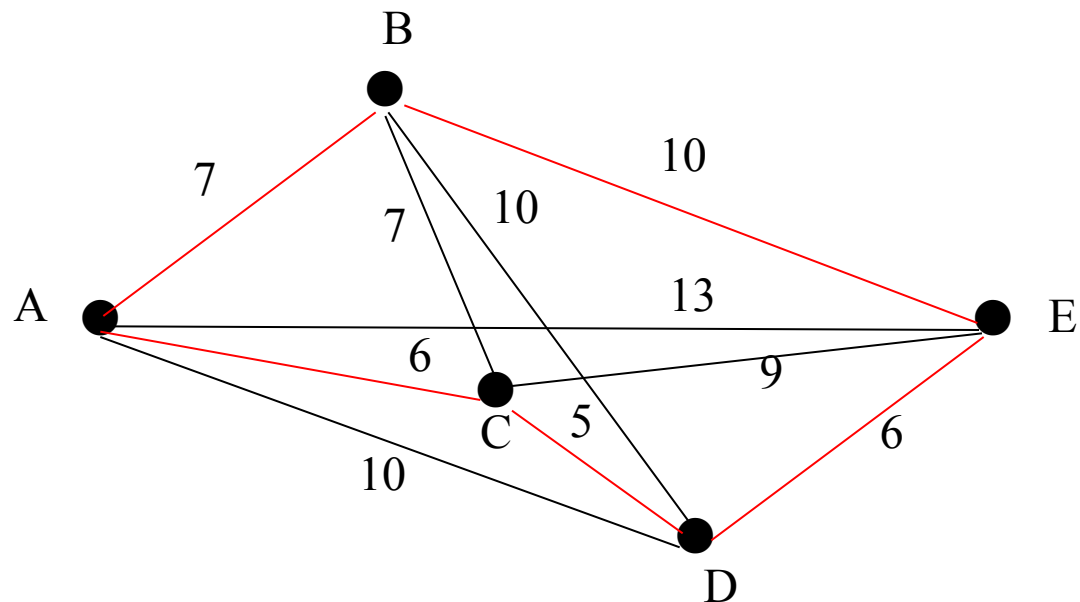
$f(x_n) = 34$,

$f(x_n) < f(x_b)$,

$x_b = (a, b, e, d, c)$,

$P = \{(a, e, b, d, c), (a, d, e, b, c), (a, c, e, d, b),$

$(a, b, d, e, c), (a, b, c, d, e), (a, b, e, c, d)\}$





从P中选择一个元素,

假设 $x_n = (a, e, b, d, c)$,

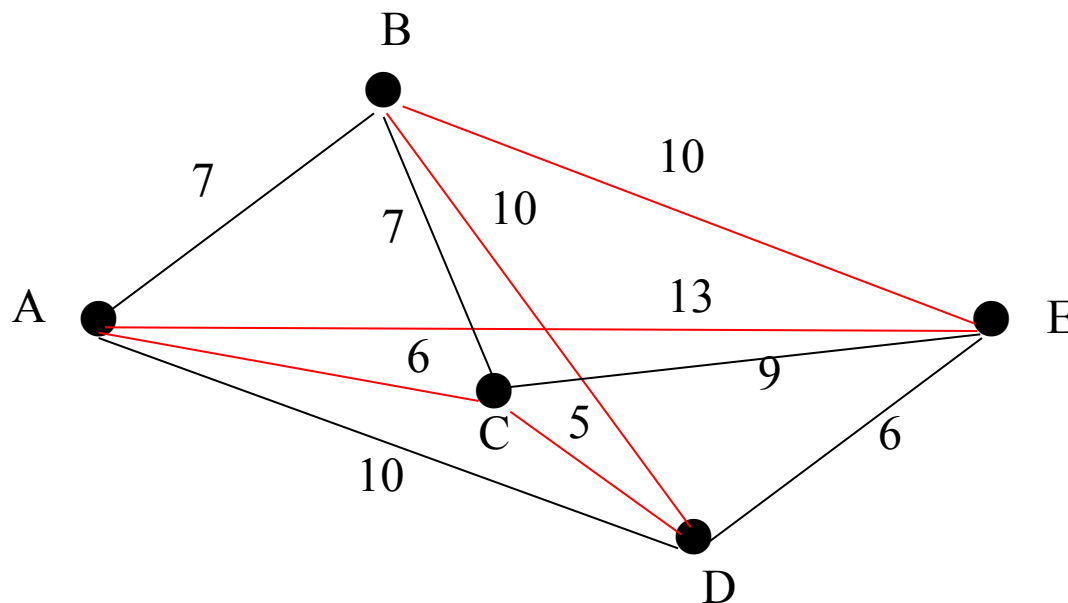
$f(x_n) = 44$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\}$

$= \{(a, d, e, b, c), (a, c, e, d, b), (a, b, d, e, c),$

$(a, b, c, d, e), (a, b, e, c, d)\}$





第7次循环

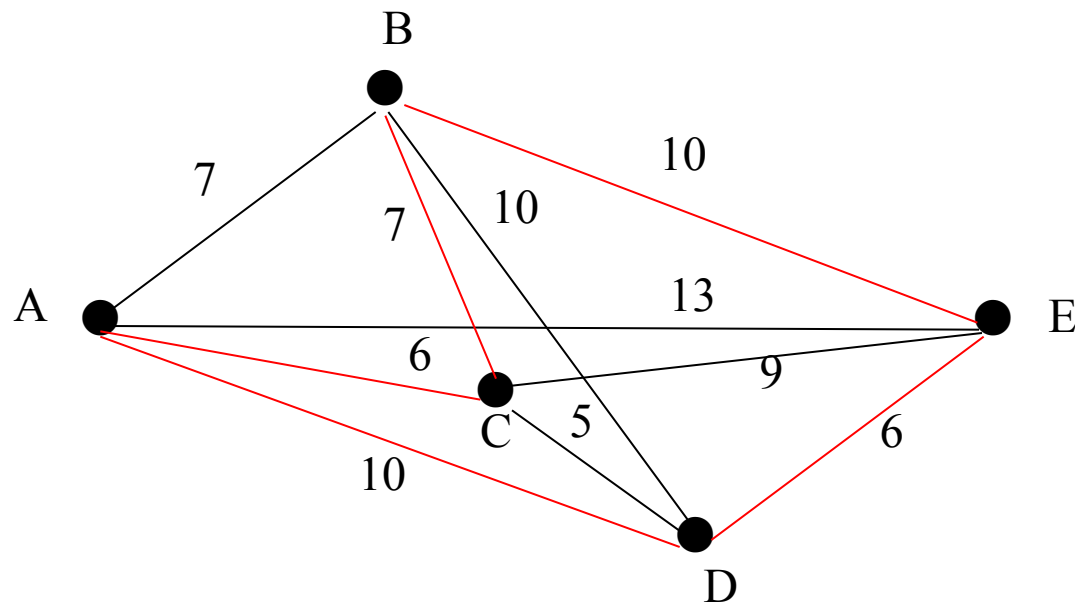
从P中选择一个元素,

假设 $x_n = (a, d, e, b, c)$,

$f(x_n) = 39, f(x_n) > f(x_b)$,

$P = P - \{x_n\}$

$= \{(a, c, e, d, b), (a, b, d, e, c), (a, b, c, d, e), (a, b, e, c, d)\}$





第8次循环

从P中选择一个元素,

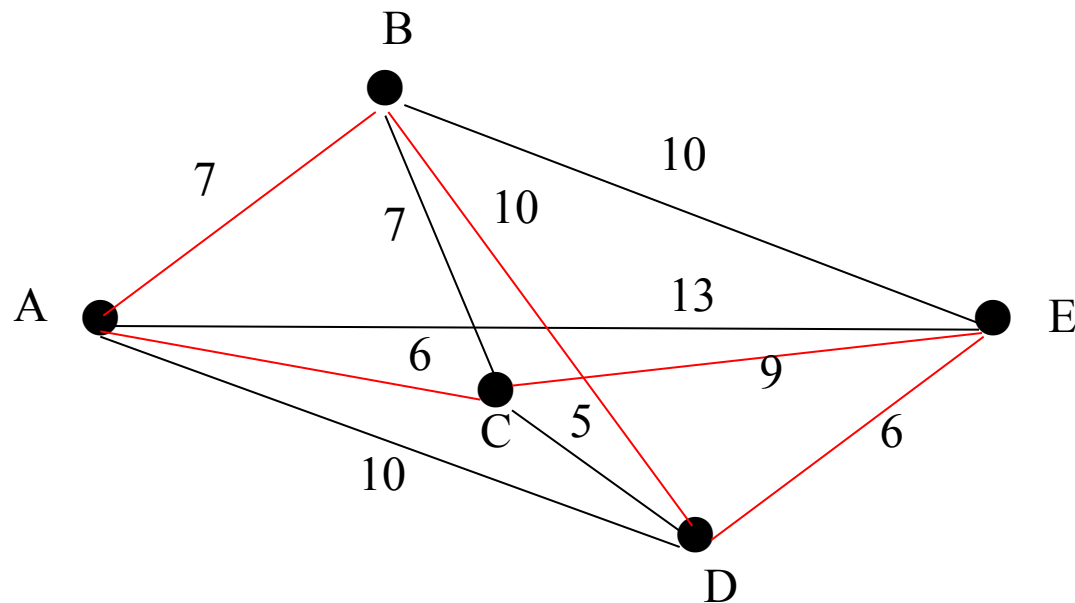
假设 $x_n = (a, c, e, d, b)$,

$f(x_n) = 38$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\}$

$= \{(a, b, d, e, c), (a, b, c, d, e), (a, b, e, c, d)\}$





第9次循环

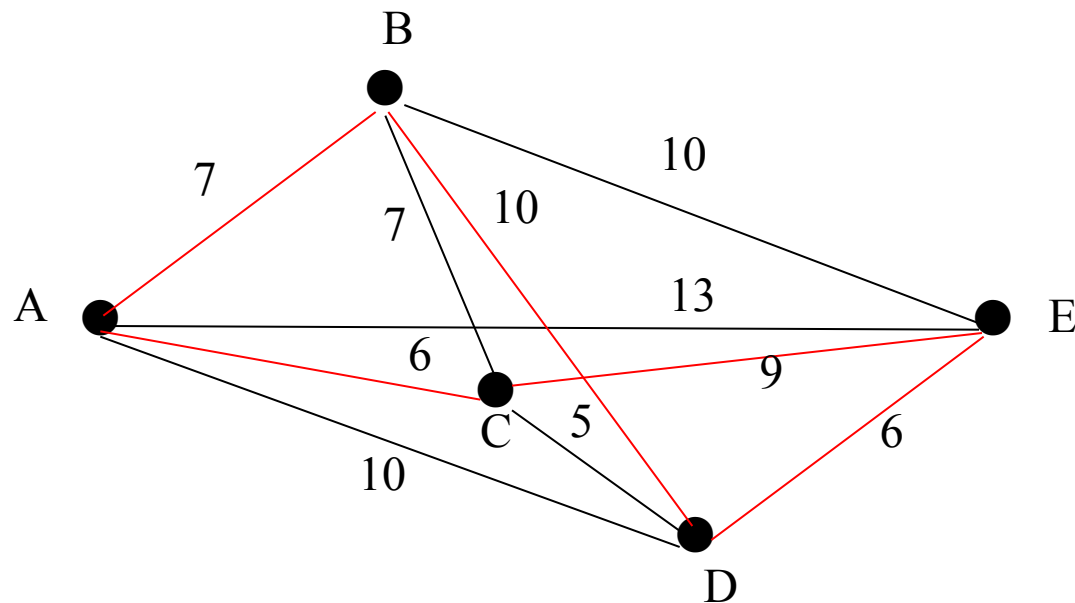
从P中选择一个元素,

假设 $x_n = (a, b, d, e, c)$,

$f(x_n) = 38$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\} = \{(a, b, c, d, e), (a, b, e, c, d)\}$





第10次循环

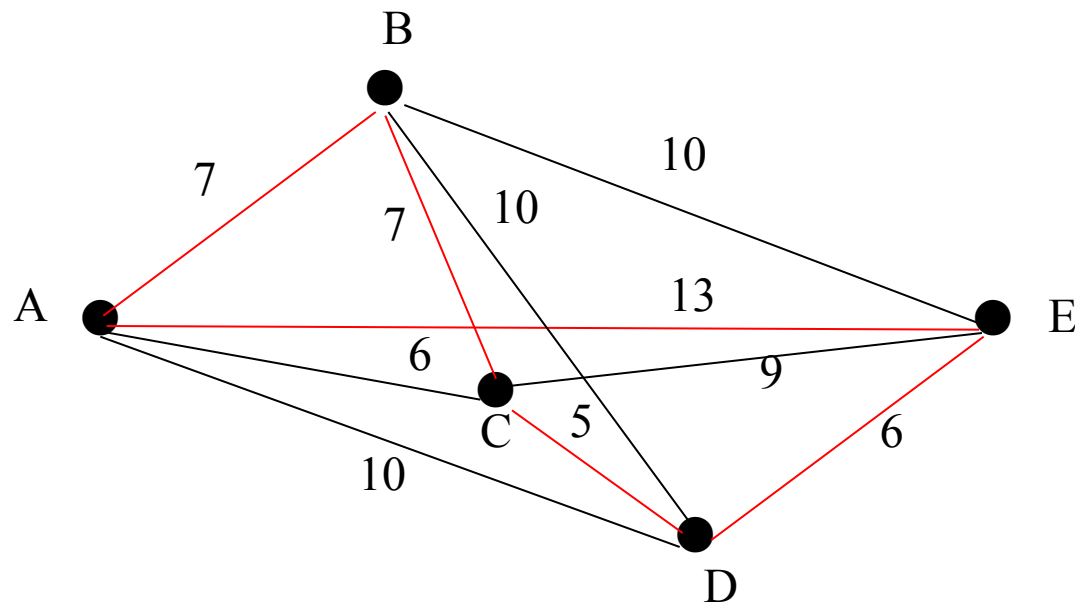
从P中选择一个元素,

假设 $x_n = (a, b, c, d, e)$,

$f(x_n) = 38$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\} = \{(a, b, e, c, d)\}$





从P中选择一个元素,

假设 $x_n = (a, b, e, c, d)$,

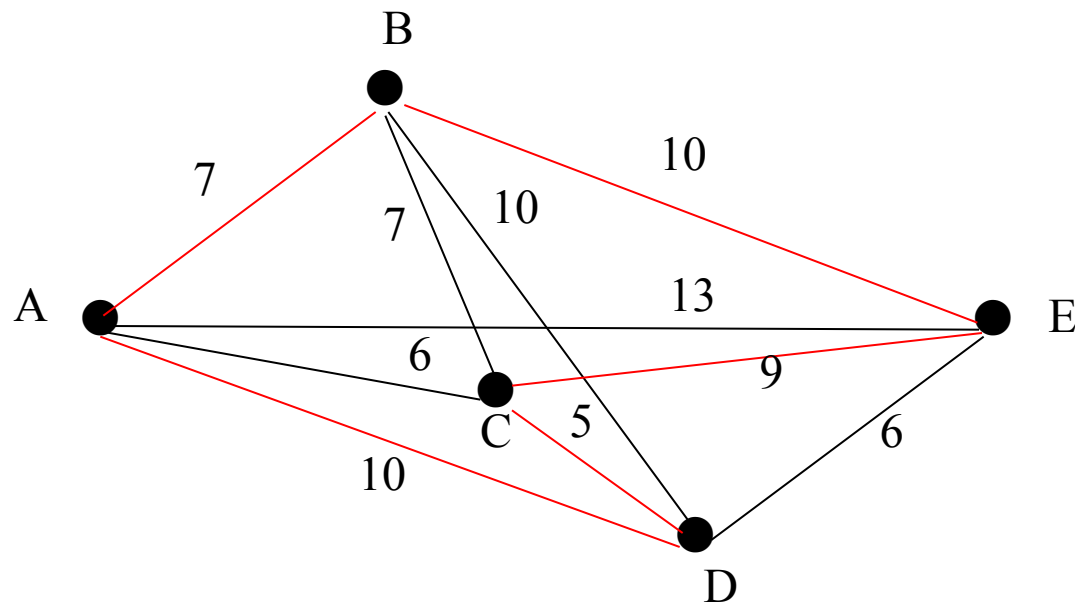
$f(x_n) = 41$,

$f(x_n) > f(x_b)$,

$P = P - \{x_n\} = \{\}$

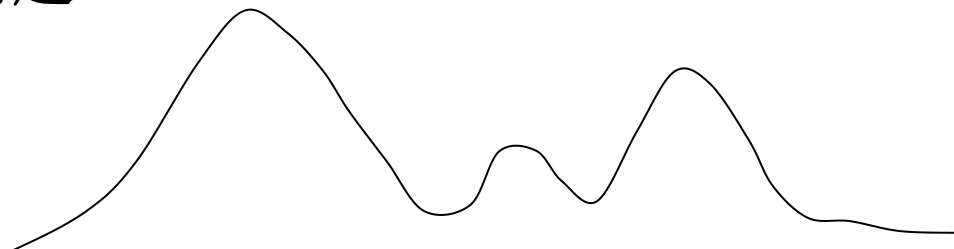
P等于空, 算法结束,

得到结果为 $x_b = (a, b, e, d, c)$, $f(x_b) = 34$ 。





- 局部最优问题



- 解决方法

- 每次并不一定选择邻域内最优的点，而是依据一定的概率，从邻域内选择一个点，指标函数优的点，被选中的概率比较大，而指标函数差的点，被选中的概率比较小

$$P_{\max}(x_i) = \frac{f(x_i)}{\sum_{x_j \in N(x)} f(x_j)}$$

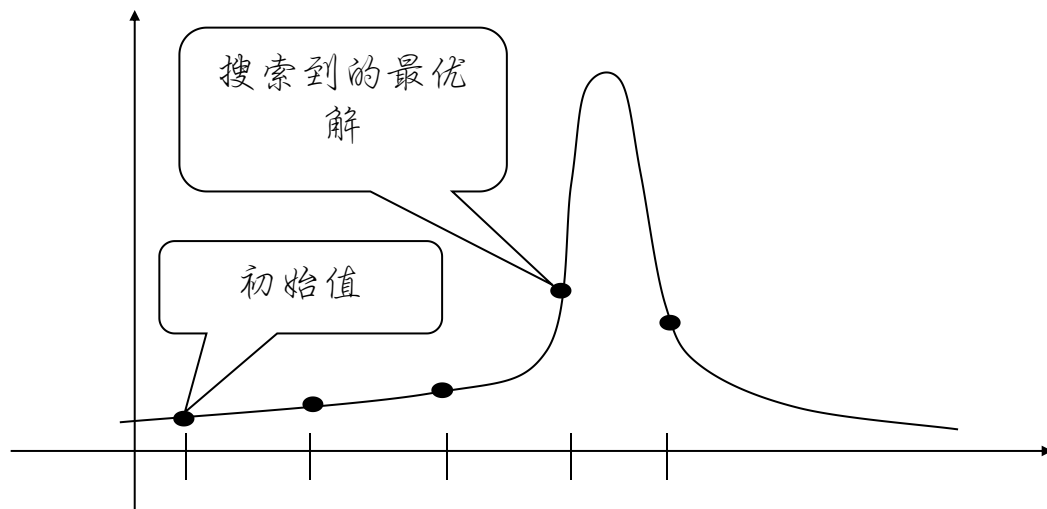
$$\begin{aligned} P_{\min}(x_i) &= \frac{1 - P_{\max}(x_i)}{\sum_{x_j \in N(x)} (1 - P_{\max}(x_j))} \\ &= \frac{1}{|N(x)| - 1} (1 - P_{\max}(x_i)) \end{aligned}$$



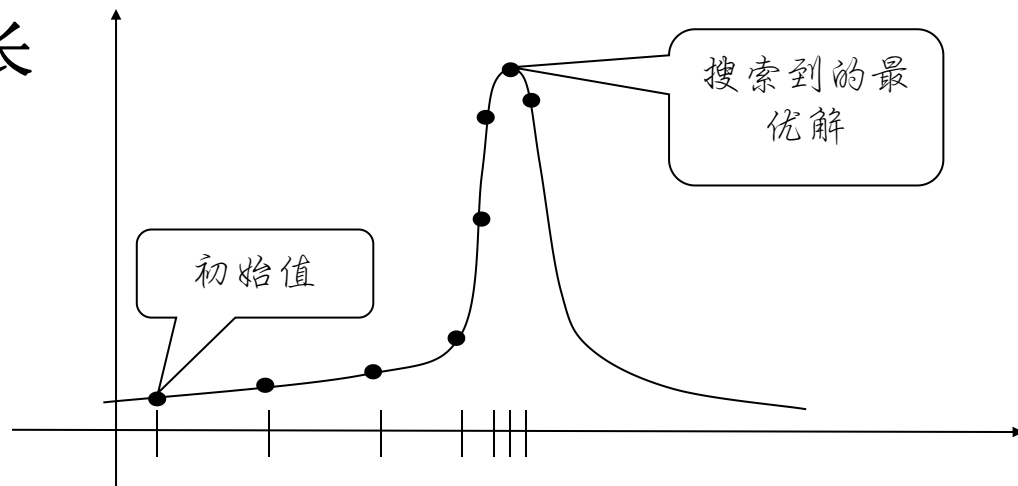
- 1 随机的选择一个初始的可能解 $x_0 \in D$, $x_b = x_0$, $P = N(x_b)$;
- 2 如果满足结束条件, 转5;
- 3 对于所有的 $x \in P$ 计算指标函数 $f(x)$, 并按上式计算每一个 x 的概率;
- 4 依计算的概率值, 从 P 中随机选择一个点 x_n , $x_b = x_n$, $P = N(x_b)$, 转2;
- 5 输出计算结果。



- 步长问题



- 解决方法：变步长





1 随机的选择一个初始的可能解 $x_0 \in D$, $x_b = x_0$, 确定一个初始步长计算

$$P = N(x_b)$$

2 如果满足结束条件, 转6

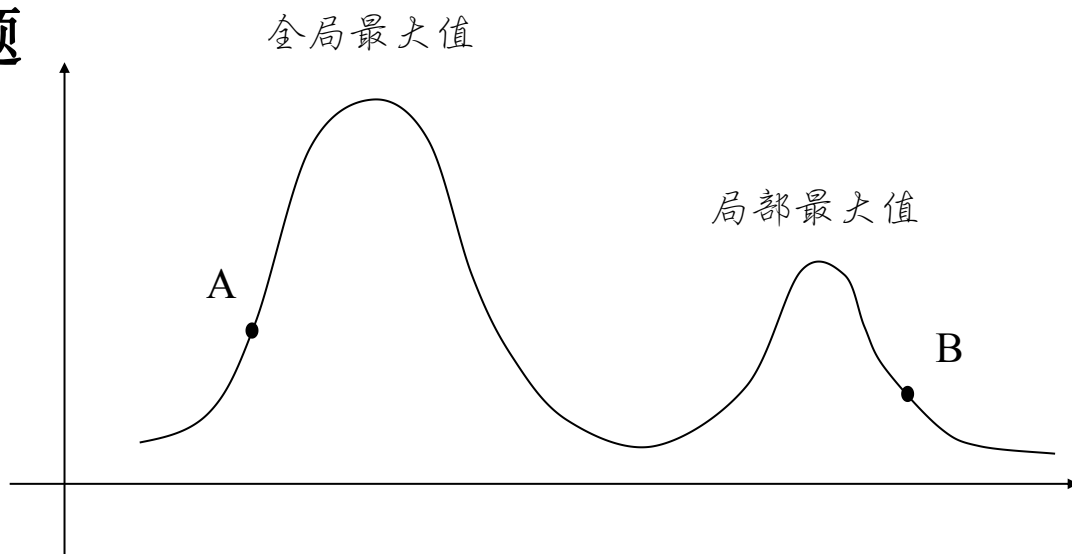
3 选择P的一个子集P', x_n 为P'中的最优解

4 如果 $f(x_n) < f(x_b)$, 则 $x_b = x_n$, 按照某种策略改变步长, 计算 $P = N(x_b)$, 转2

5 否则 $P = P - P'$, 转2。



- 起始点问题



- 解决方案

- 随机的生成一些初始点，从每个初始点出发进行搜索，找到各自的最优解。再从这些最优解中选择一个最好的结果作为最终的结果。



1 $k = 0$

2 随机的选择一个初始的可能解 $x_0 \in D$, $x_b = x_0$, $P = N(x_b)$

3 如果满足结束条件, 转9

4 选择P的一个子集P', x_n 为P'中的最优解

5 如果 $f(x_n) < f(x_b)$, 则 $x_b = x_n$, $P = N(x_b)$, 转3

6 否则 $P = P - P'$, 转3

7 $k = k + 1$

8 若k达到指定的次数, 则从k个结果中选择一个最好的结果输出, 否则转2

9 输出结果



- 多方法集成

- 以上几种解决方法可以结合在一起使用，比如第一、第二种方法的结合，就产生了我们将在后面介绍的模拟退火方法。

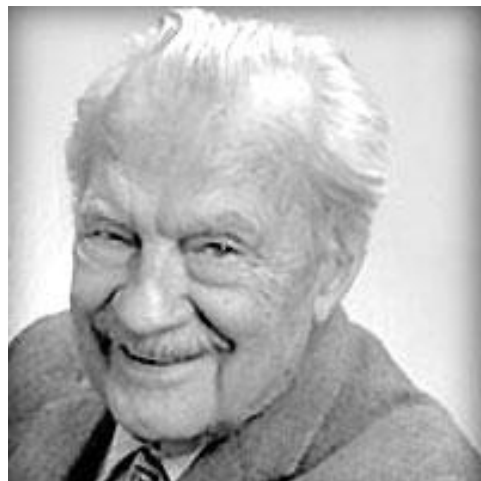


- 与图搜索算法的区别
 - LS初始化一个完整的解 x_0
 - LS依赖解的性能评估函数 $f(x)$
 - LS确定得到局部最优解
 - LS引入随机的概念：子邻域选择
- 算法输出
 - 局部最优解
- 算法停止条件
 - 邻域为空
 - 实际应用中可能需要复合别的条件，如最大迭代次数

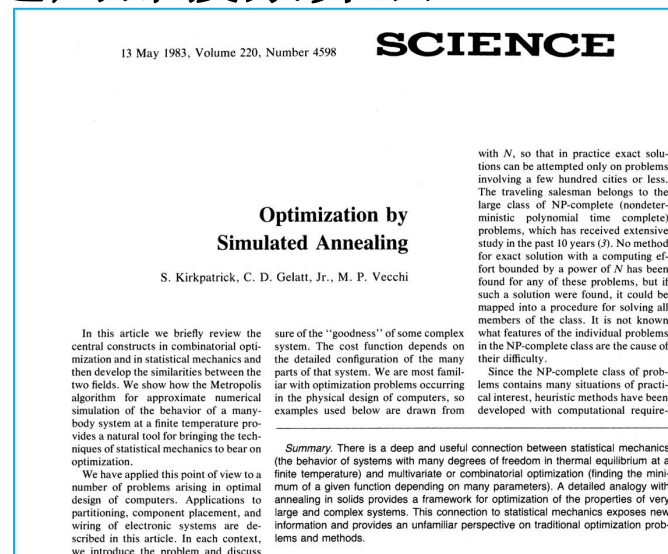


4.3 模拟退火算法

- 模拟退火（Simulated Annealing, SA）算法是局部搜索算法的一种扩展
- 最早由Metropolis在1953年提出，Kirkpatrick等人在1983年成功地将模拟退火算法用于求解组合优化问题
- **基本思想**是借用金属的退化过程改进局部搜索算法



Nicholas Metropolis (1915-1999)



1983年Science SA论文



- **溶解过程：**随着温度的不断上升，粒子逐渐脱离其平衡位置，变得越来越自由，直到达到固体的溶解温度，粒子排列从原来的有序状态变为完全的无序状态。
- **退火过程：**随着温度的下降，粒子的热运动逐渐减弱，粒子逐渐停留在不同的状态，其排列也从无序向有序方向发展，直至到温度很低时，粒子重新以一定的结构排列。
- 粒子不同的排列结构，对应着不同的能量水平。如果退火过程是缓慢进行的，也就是说，温度的下降如果非常缓慢的话，使得在每个温度下，粒子的排列都达到一种平衡态，则当温度趋于0（绝对温度）时，系统的能量将趋于最小值。
- 如果以粒子的排列或者相应的能量来表达固体所处的状态，在温度 T 下，固体所处的状态具有一定的随机性。一方面，物理系统倾向于能量较低的状态，另一方面，热运动又妨碍了系统准确落入低能状态。



固体退火达到能量极优条件

- (1) 初始温度必须足够高;
- (2) 在每个温度下, 状态的交换必须足够充分;
- (3) 温度 T 的下降必须足够缓慢。



- 从状态*i*转换为状态*j*的**准则**:

如果 $E(j) \leq E(i)$, 则状态转换被接受;

如果 $E(j) > E(i)$, 则状态转移被接受的概率为:

$$\exp\left\{\frac{E(i) - E(j)}{KT}\right\}$$

思想飞跃

其中 $E(i)$ 、 $E(j)$ 分别表示在状态*i*、*j*下的能量, T 是温度, $K > 0$ 是波尔兹曼常数。



1 随机选择一个解 i , $k=0$, $t_0=T_{max}$ (初始温度), 计算指标函数 $f(i)$;

2 如果满足结束条件, 则转Line 15;

3 **Begin**

4 如果在该温度内达到了平衡条件, 则转Line 13;

5 **Begin**

6 从 i 的邻域 $N(i)$ 中随机选择一个解 j ;

7 计算指标函数 $f(j)$;

8 如果 $f(j)<f(i)$, 则 $i=j$, $f(i)=f(j)$, 转Line 4;

9 计算 $P_t(i \Rightarrow j) = e^{-\frac{f(j)-f(i)}{t}}$

10 如果 $P_t(i \Rightarrow j) > Rand(0, 1)$, 则 $i=j$, $f(i)=f(j)$;

11 转Line 4;

12 **End**

13 $t_{k+1}=Drop(t_k)$, $k=k+1$;

14 **End**

15 输出结果。

内循环
状态平衡

外循环
逐步降温



- 初始温度的选取
- 内循环的结束条件，即每个温度状态交换何时结束
- 外循环的结束条件，即温度下降到什么时候结束
- 温度的下降方法



- 解的表示

➤ n 个城市的任何一种排列均是问题的一个可能解，表示为：

$$(\pi_1, \dots, \pi_n)$$

- 指标函数(能量函数)

$$f(\pi_1, \dots, \pi_n) = \sum_{i=1}^n d_{\pi_i \pi_{i+1}}$$

其中

$$\pi_{n+1} = \pi_1$$



• 新解的产生

- 采用第一节介绍的两个城市间的逆序交换方式得到问题的一个新解。
- 设当前解是 (π_1, \dots, π_n) 被选中要逆序交换的城市是第 u 和第 v 个到访的城市， $u < v$ 。则逆序排列 u 和 v 之间的城市，得到问题的新解为：

$$(\pi_1, \dots, \pi_u, \pi_{v-1}, \dots, \pi_{u+1}, \pi_v, \pi_{v+1}, \dots, \pi_n)$$

则两个路径的距离差为：

$$\Delta f = (d_{\pi_u \pi_{v-1}} + d_{\pi_{u+1} \pi_v}) - (d_{\pi_u \pi_{u+1}} + d_{\pi_{v-1} \pi_v})$$

• 新解的接受准则

$$A_t = \begin{cases} 1 & \text{如果 } \Delta f < 0 \\ e^{-\frac{\Delta f}{t}} & \text{其他} \end{cases}$$



• 初始参数的确定

➤ 康立山等人的方法：

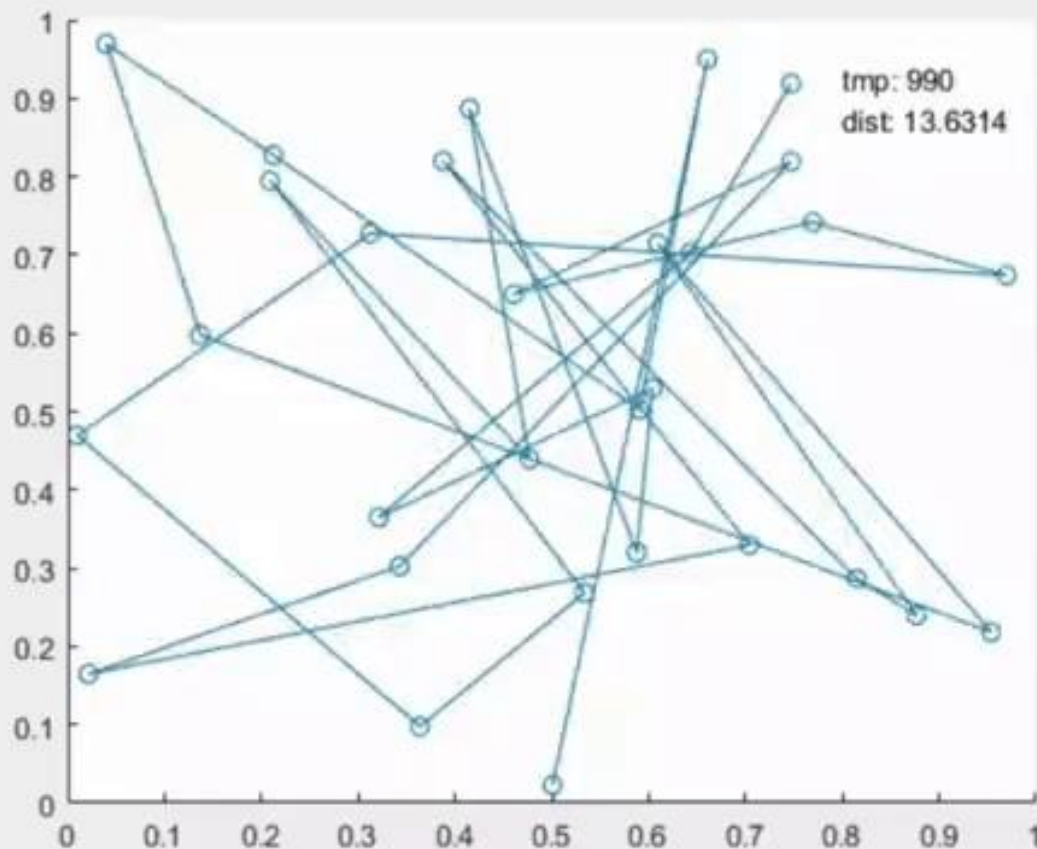
- 初始温度 $t_0=280$
- 在每个温度下采用固定的迭代次数， $L_k=100n$ ， n 为城市数
- 温度的衰减系数 $=0.92$ ，即 $t_{k+1}=0.92 \times t_k$
- 算法的停止准则为：当相邻两个温度得到的解无任何变化时算法停止

➤ Nirwan Ansari和Edwin Hou的方法：

- 初始温度 t_0 是这样确定的：从 $t_0=1$ 出发，并以 $t_0=1.05 \times t_0$ 对 t_0 进行更新，直到接受概率大于等于0.9时为止，此时得到的温度为初始温度
- 在每个温度下采用固定的迭代次数， $L_k=10n$ ， n 为城市数
- 温度的衰减系数 $=0.95$ ，即 $t_{k+1}=0.95 \times t_k$



应用举例：TSP问题





- 全局搜索算法
 - 可证明，SA算法依概率收敛
 - 与LS算法不同
- 算法设计思想上的飞跃
 - 依概率接收差解
 - 暂时的差，可能导致全局的优
- 算法起源借鉴了物理中的金属退火过程，是一种拟物算法



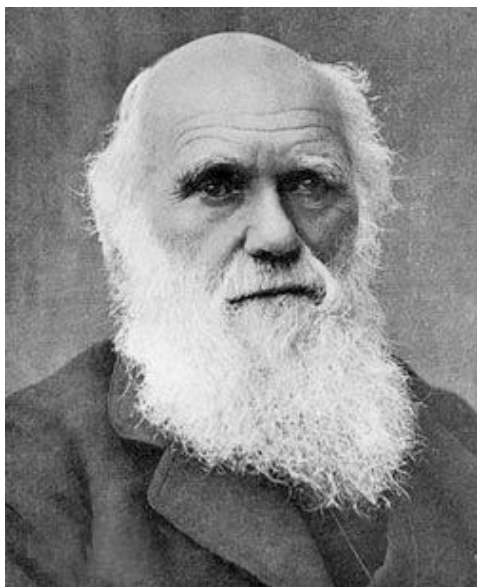
- 问题求解需要问题与算法匹配
 - 算法利用问题信息能够提高搜索效率
- 不确定性搜索算法
 - 局部搜索算法，模拟退火算法，演化算法
 - 一类逼近算法，从低质量的解到高质量解
 - 一类随机算法，搜索过程不确定，搜索结果具有随机性
 - SA和EA是全局搜索算法
 - 当前研究热点



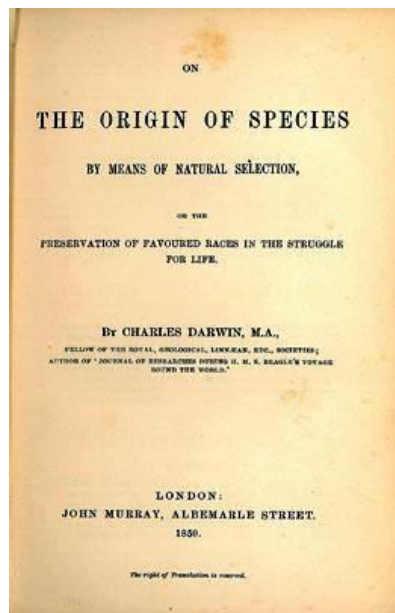
谢谢



- 起源借鉴达尔文进化论：“物竞天择、适者生存”
- 20世纪50-90年代由欧美科学家首先提出
- 演化算法已为一种最优化问题求解的方法论，形成了一个有效的问题求解工具算法族
- 当前算法除个别文字表述外，已与生物（进化论）无关



Charles Robert Darwin
(1809-1882)



《物种起源》



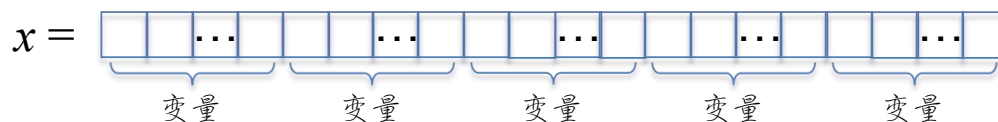
• 演化算法是一类基于群体的随机优化算法

1. 针对一个优化问题，**随机初始化**一组解（群体）
2. 从群体中**选择**一组较优个体
3. 利用较优个体，**产生**一组新的个体来替换原来的群体
- 4. 重复2-3，直到算法收敛**
5. 输出最后的最优个体

- 群体：采用多个点来搜索，类似多点爬山，每个解被称为一个“个体”
- 好解选择：好的个体里包含求解问题的“好的信息”----优胜劣汰
- 新解产生：通过个体“合作”，挖掘“好的信息”，探索好的搜索方向
- 随机：1-4步，均依赖随机性



- 解的二进制编码表示 $x = (x_1, x_2, \dots, x_n)$, $x_i \in \{0, 1\}$
 - 理论上，二进制数可以表达任意数据结构
 - 可以与实数、整数在误差范围内相互转换
 - 多个数可以用多个二进制组合而成



DECIMAL	HEX	BINARY
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

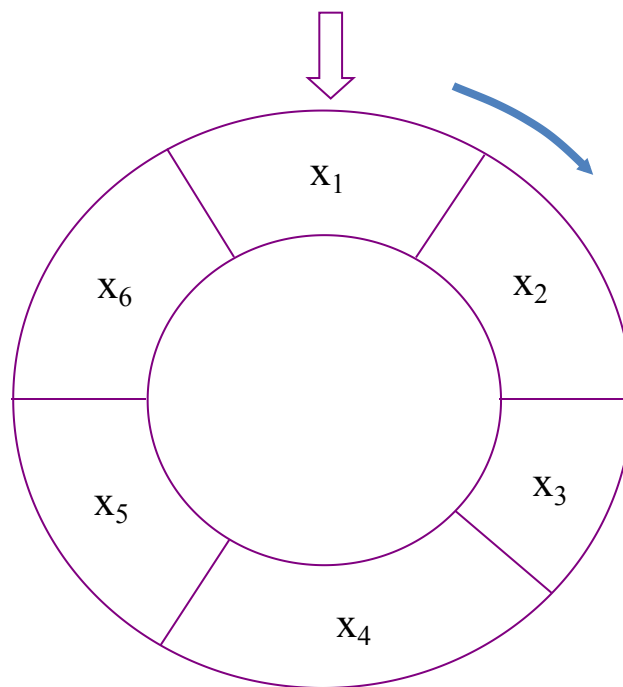
- “轮盘赌”法

- 设群体的规模为 N , $F(x_i)(i=1, \dots, N)$ 是其中 N 个个体的评估值, 则第 i 个个体被选中的概率由下式给出:

$$p(x_i) = \frac{F(x_i)}{\sum_{j=1}^N F(x_j)}$$

- 算法流程

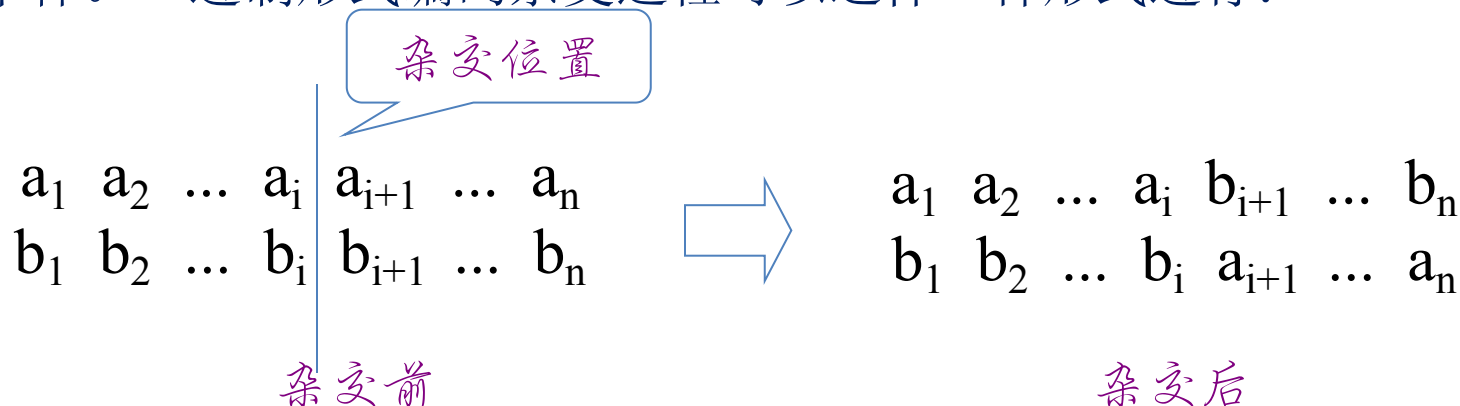
1. $r = \text{random}(0, 1)$, $s = 0$, $i = 0$;
2. 如果 $s \geq r$, 则转 (4) ;
3. $s = s + p(x_i)$, $i = i + 1$, 转 (2) ;
4. x_i 即为被选中的个体;
5. 输出 i , 结束。





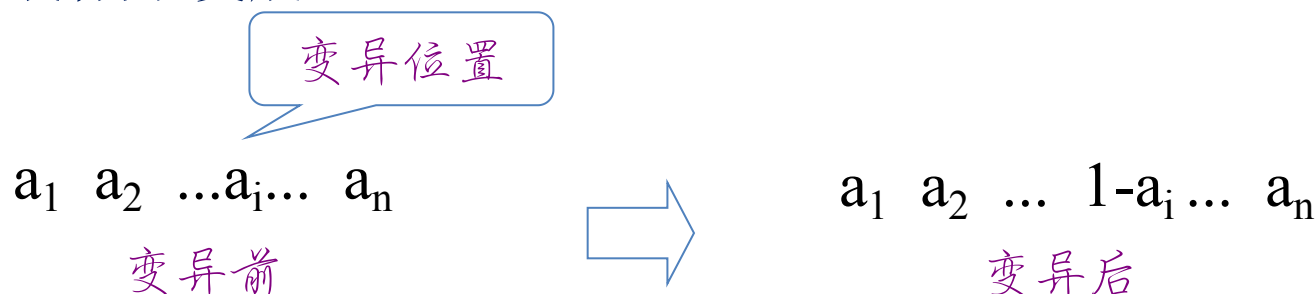
• 杂交

- 发生在两个个体之间，经杂交以后，产生两个具有双亲的部分信息的新的个体。二进制形式编码杂交过程可以这样一种形式进行：



• 变异

- 变异发生在个体某一个位置上，当以二进制编码时，变异的变量由0变成1，或者由1变成0：





演化算法(Evolutionary Algorithm, EA)

- 1 给定群体规模 N ，交配概率 p_c 和变异概率 p_m ， $t=0$ ；
- 2 随机生成 N 个个体作为初始群体；
- 3 对于群体中的每一个 x_i 分别计算其评估值 $F(x_i)$ ；
- 4 如果算法满足停止准则，则转Line 10；
- 5 对群体中的每一个 x_i 计算概率；
- 6 依据计算得到的概率值，从群体中随机的选取 N 个个体，得到种群；
- 7 依据交配概率 p_c 从种群中选择个体进行杂交，其子代进入新的群体，种群中未进行交配的个体，直接复制到新群体中；
- 8 依据变异概率 p_m 从新群体中选择个体进行变异，用变异后的个体代替新群体中的原个体；
- 9 用新群体代替旧群体， $t=t+1$ ，转Line 3；
- 10 进化过程中评估值最大的个体，经解码后作为最优解输出；



例：求函数的最大值

$$f(x) = x^2$$

其中 x 为 $[0, 31]$ 间的整数

编码：采用二进制形式编码

由于 x 的定义域是 $[0, 31]$ 间的整数，刚好可以用5位二进制数表示，因此可以用5位二进制数表示该问题的解，即染色体。如00000表示 $x=0$ ，10101表示 $x=21$ ，11111表示 $x=31$ 等



例：求函数的最大值

- 评估函数：
 - 直接使用函数 $f(\mathbf{x})$ 作为适应函数
- 假设群体的规模 $N=4$ ，交配概率 $p_c=100\%$ ，变异概率 $p_m=1\%$
- 设随机生成的初始群体为：
01101, 11000, 01000, 10011
- 选择方法：“轮盘”法



第0代选择情况

序号	群体	评估值	选择概率 (%)	选中次数
1	01101	169	14.44	1
2	11000	576	49.23	2
3	01000	64	5.47	0
4	10011	361	30.85	1

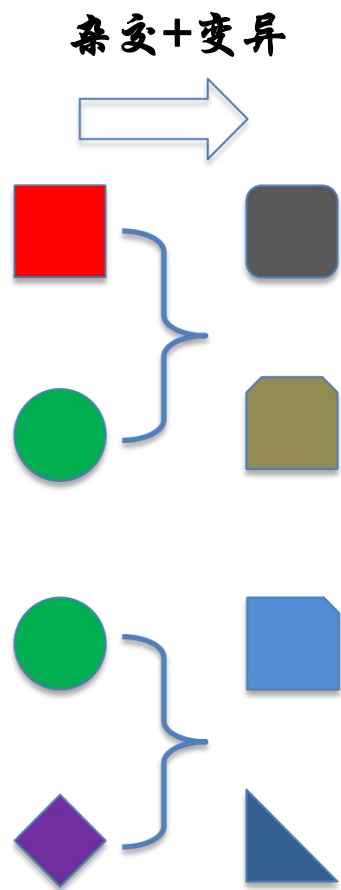
好解选择





第0代新解产生情况

序号	种群	交配对象	交配位	子代	评估值
1	01101	2	4	01100	144
2	11000	1	4	11001	625
3	11000	4	2	11011	729
4	10011	3	2	10000	256

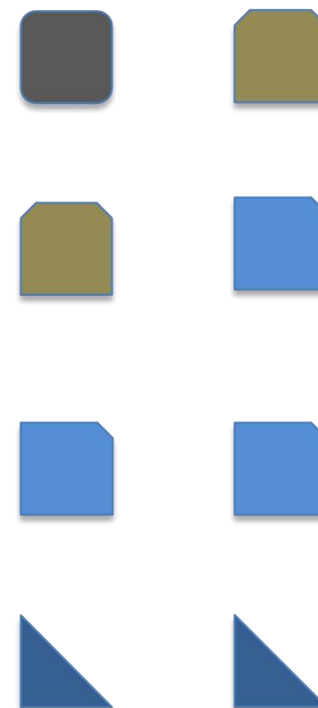




第1代选择情况

序号	群体	评估值	选择概率 (%)	选中次数
1	01100	144	8.21	0
2	11001	625	35.62	1
3	11011	729	41.56	2
4	10000	256	14.60	1

好解选择

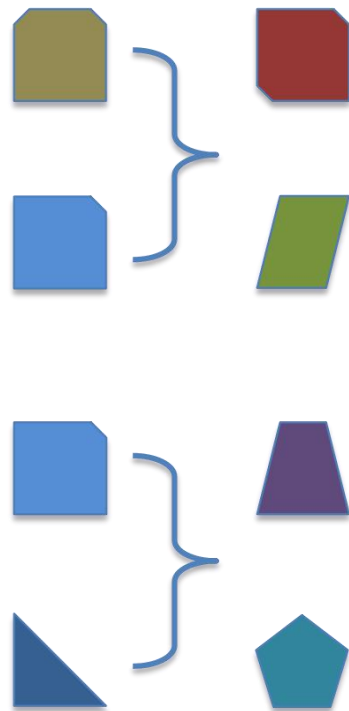




第1代新解产生情况

序号	种群	交配对象	交配位	子代	评估值
1	11001	2	3	11011	729
2	11011	1	3	11001	625
3	11011	4	1	10000	256
4	10000	3	1	11011	729

杂交+变异

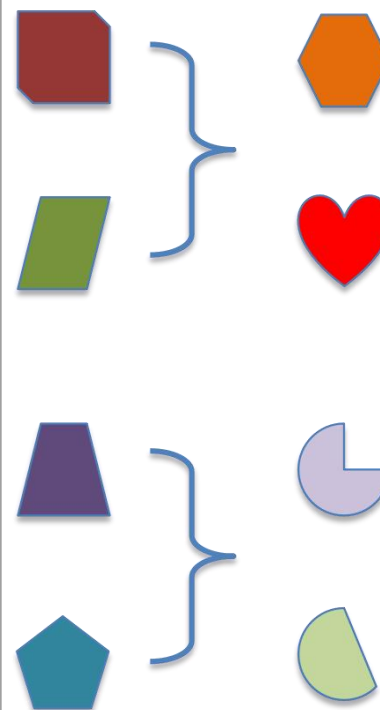




第2代新解产生情况

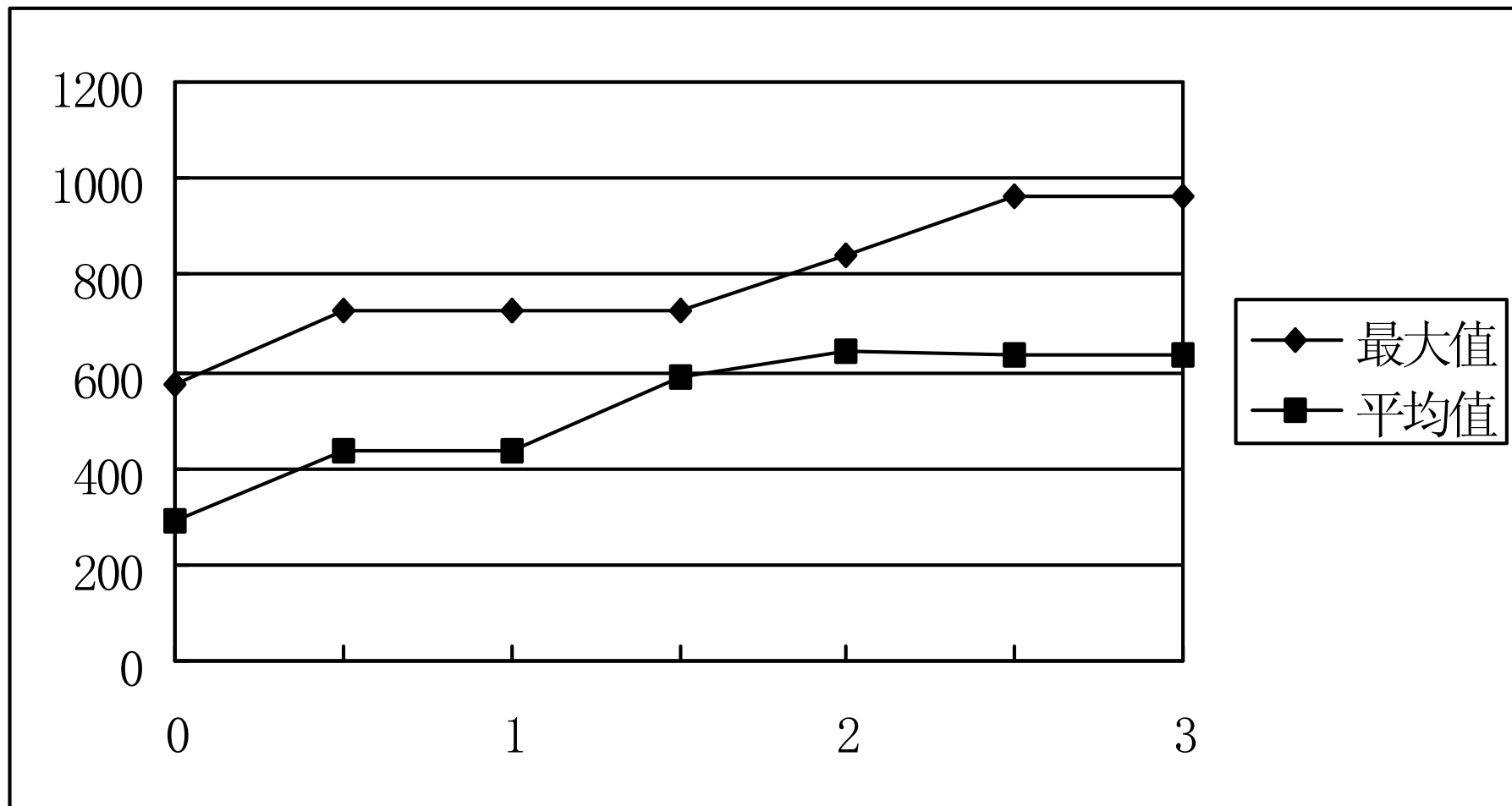
序号	种群	交配对象	交配位	子代	适应值
1	11011	2	3	11001	625
2	11101	1	3	11111	961
3	10000	4	2	10001	289
4	11011	3	2	11010	676

杂交+变异





最大评估值、平均评估值进化曲线





- 演化算法是一个**基于群体的随机搜索算法**，适用于数值求解具有多参数、多变量、多目标等复杂的最优化问题。
- 演化算法对**求解问题的评估函数没有什么特殊的要求**，比如不要求诸如连续性、导数存在、单峰值假设等，甚至于不需要显式的写出评估函数。
- 在经过编码以后，演化算法几乎**不需要任何与问题有关的知识**，唯一需要的信息是评估值的计算，但问题有关知识能帮助提高算法求解效率。
- 演化算法具有天然的并行性，适用于**并行求解**。