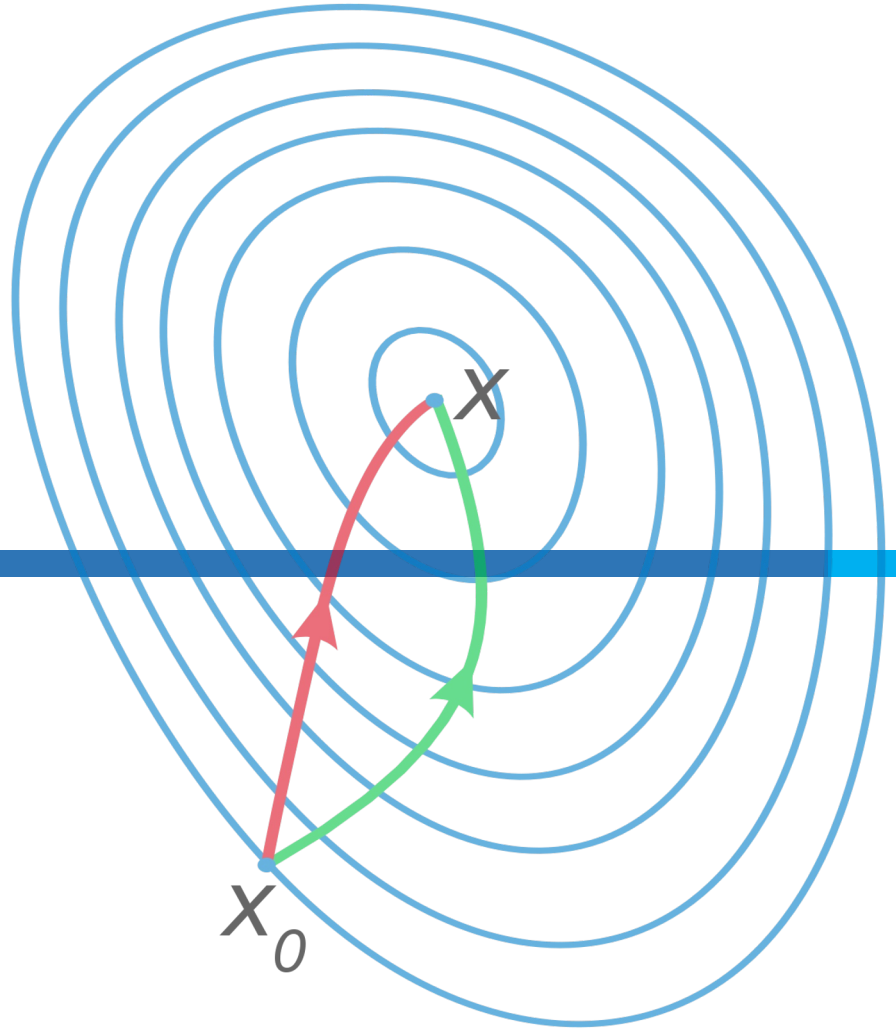
最优化方法



第六周

计算机学院
余皓然

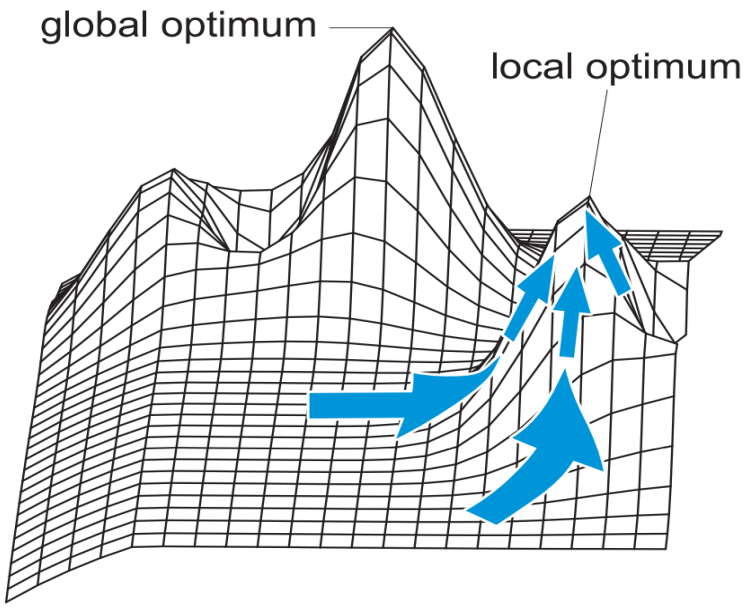
2024/4/1

模拟退火算法

Part1 如何从复杂度的角度衡量算法好坏？
Part2 针对一些有特定形式的最优化问题，如何找到最优解？
Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？
局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法
Part4 针对较复杂的多目标最优化问题，如何找到较好解？

登山搜索缺陷

（原始）登山搜索只考虑最优化目标函数的方向，易陷入局部最优

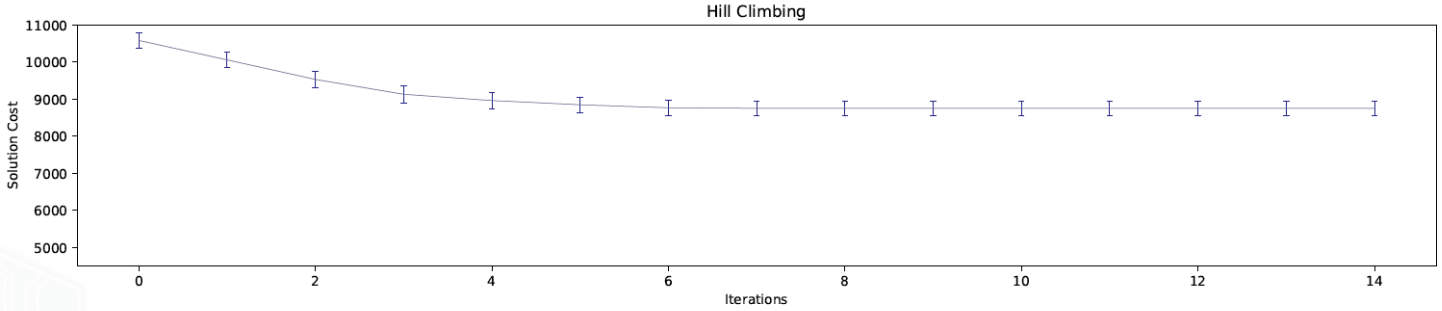


登山搜索缺陷

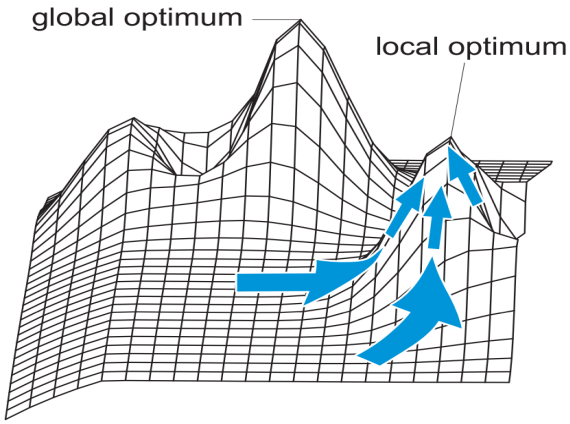
（原始）登山搜索只考虑最优化目标函数的方向，易陷入局部最优

a) 15 cities

前述例子中15个城市旅行商问题（最小化问题）



解决思路



允许朝下山方向移动

引入随机性——随机登山搜索

随机登山搜索

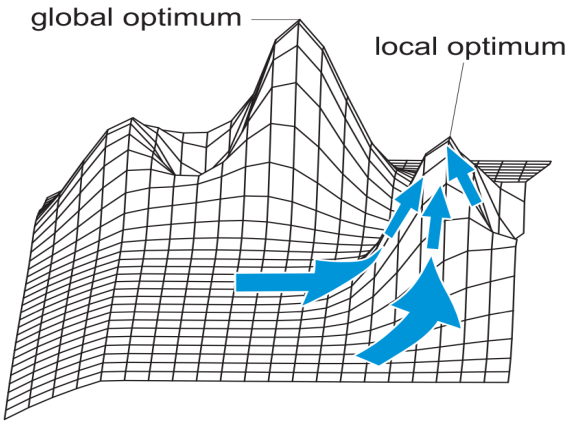
0 初始化当前解$\boldsymbol{x}_{\text{current}}$

1 随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$

2 以概率$\frac{1}{1+e^{\frac{f(x_{\text{current}})-f(x_{\text{neighbor}})}{T}}}$令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

3 若算法终止条件不满足，返回1

解决思路



允许朝下山方向移动

引入随机性——随机登山搜索

随机登山搜索

0 初始化当前解$\boldsymbol{x}_{\text{current}}$

1 随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$

2 以概率$\frac{1}{1+e^{\frac{f(x_{\text{current}})-f(x_{\text{neighbor}})}{T}}}$令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

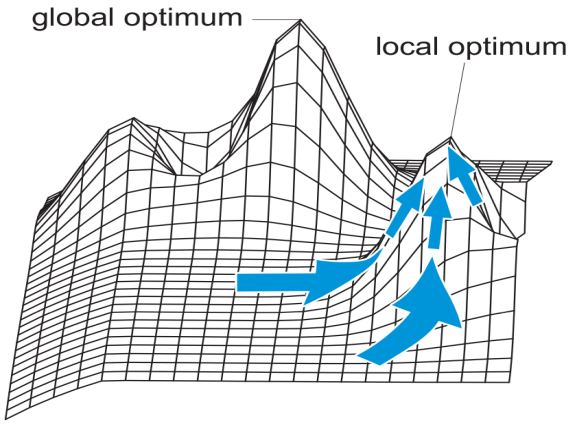
3 若算法终止条件不满足，返回1

仍旧存在问题?

如果已经抵达全局最优可能会跳出来

解决思路？

解决思路



允许朝下山方向移动

要动态降低下山移动概率

引入随机性——随机登山搜索

随机登山搜索

0 初始化当前解$\boldsymbol{x}_{\text{current}}$

1 随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$

2 以概率$\frac{1}{1+e^{\frac{f(x_{\text{current}})-f(x_{\text{neighbor}})}{T}}}$令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

3 若算法终止条件不满足，返回1

仍旧存在问题?

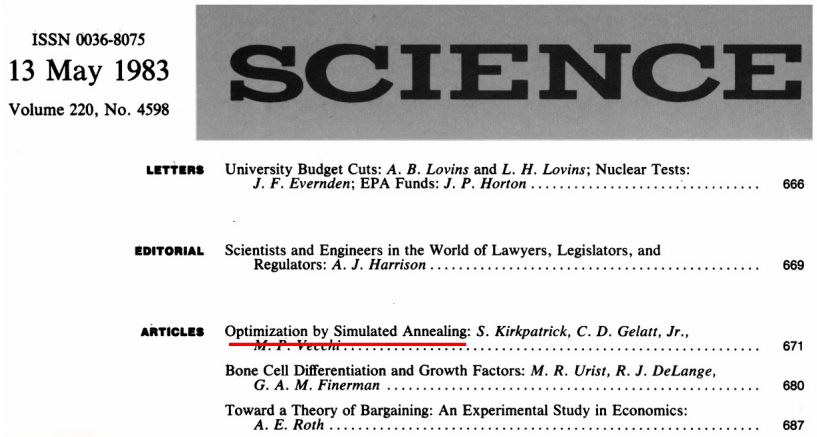
如果已经抵达全局最优可能会跳出来

解决思路？ 随着迭代次数变化，动态调整选择邻域解的概率

模拟退火算法

模拟退火算法（Simulated Annealing，SA）

- 在引入随机性的同时，随着迭代次数变化动态调整随机性


ISSN 0036-8075
13 May 1983
Volume 220, No. 4598

SCIENCE

Optimization by simulated annealing

S Kirkpatrick, CD Gelatt Jr, MP Vecchi - science, 1983 - science.org

… Of classic **optimization** problems, the traveling salesman problem has received the most intensive study. To test the power of **simulated annealing**, we used the algorithm on traveling …

☆ Save 99 Cite Cited by 53612 Related articles All 73 versions »

1983年提出用于解决旅行商问题等

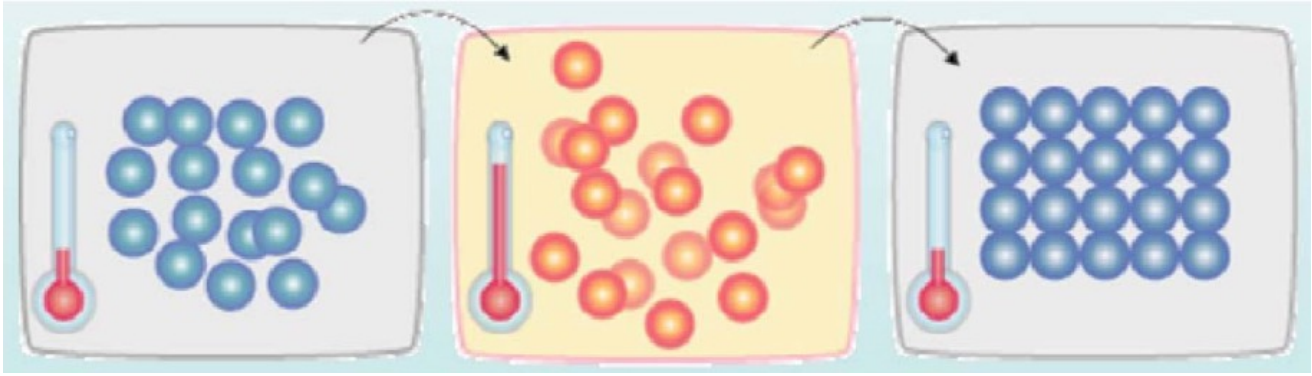
模拟退火算法

工业热处理：退火、正火、淬火、回火

“回火、淬火、正火、退火，这4把火终于整明白了！”

https://www.bilibili.com/video/BV1M3411q7DC?spm_id_from=333.337.search-card.all.click

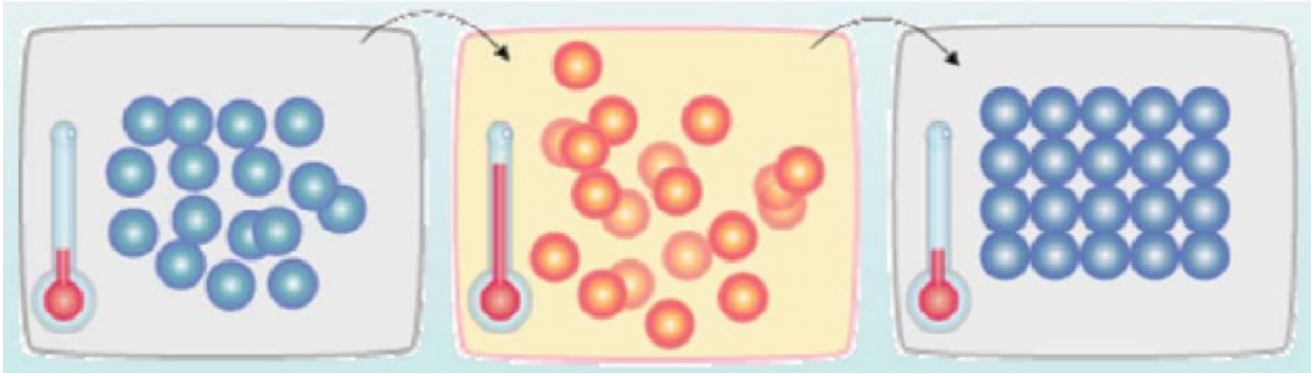
模拟退火算法



物理退火过程：将金属加热到高温，再将其缓慢冷却

- 退火过程中，温度高时，原子活跃度高；温度低时，原子活跃度低
- 经过退火过程，原子排列将改变
- 目的是恢复金属因冷加工而降低的性质，增加柔软性、延展性、韧性

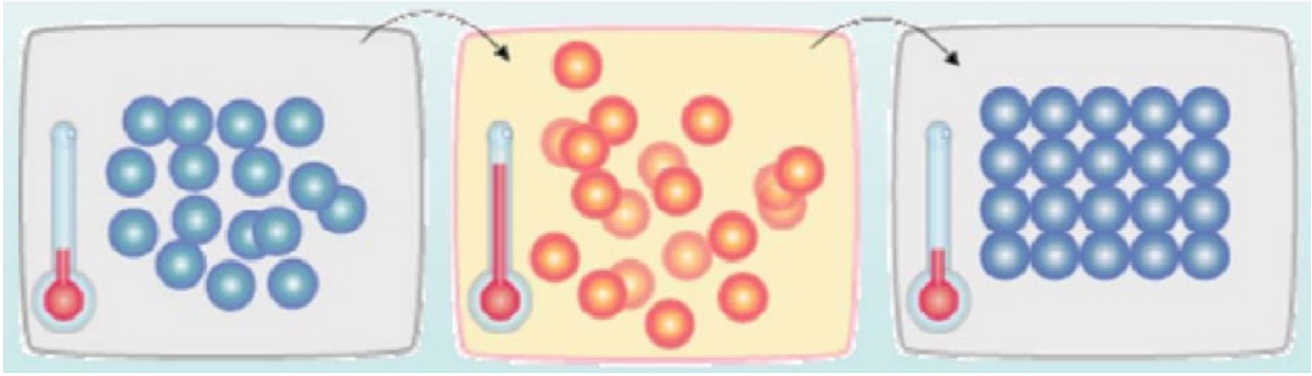
模拟退火算法



模拟退火算法：模仿物理退火过程

- 根据温度的变化调整用邻域解代替当前解的概率

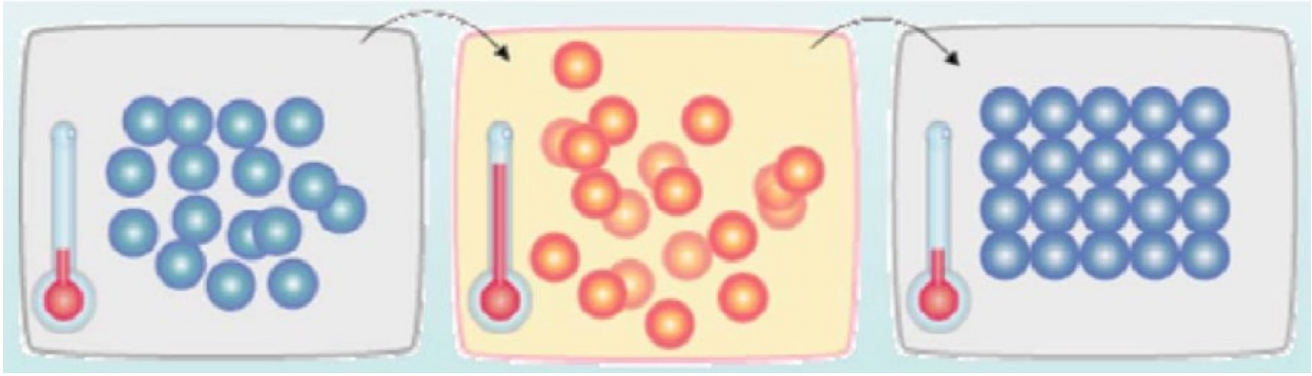
模拟退火算法



模拟退火算法：模仿物理退火过程

- 根据温度的变化调整用邻域解代替当前解的概率
  - 温度高时，随机性高：更接纳邻域解，即便质量较差
  - 温度低时，随机性低：趋向于只接纳质量较好的邻域解
  - 温度降为0时，停止搜索，结束算法

模拟退火算法



模拟退火算法与物理退火过程完整的对偶关系更为复杂（略）

当前解 —— 当前系统物理状态

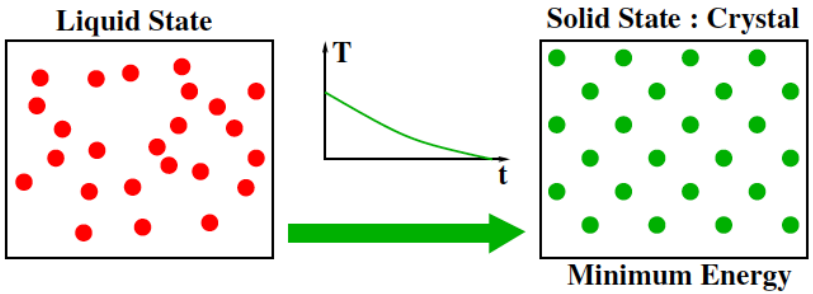
解的质量 —— 物理状态的能量

最优目标函数 —— 能量最小状态（基态）

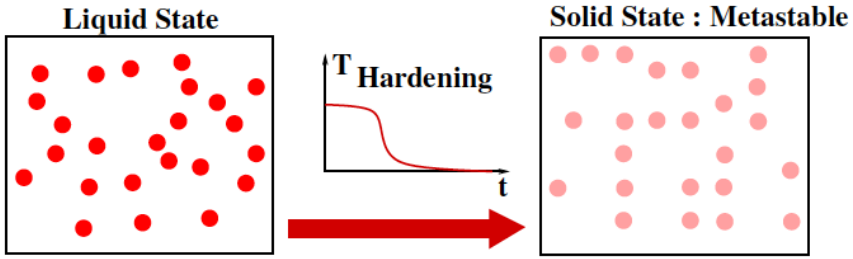
...

模拟退火算法

退火需要温度缓慢下降



退火：最终固体状态能量最小、原子对称排列



淬火：最终固体状态非能量最小状态

图片取自Delahaye, et al, “Simulated annealing: From basics to applications”

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$

1 for t=1 to ∞:

2 　根据温度下降规则和t，确定当前温度T　温度下降规则说明了t与T的函数关系

3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$

4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$　选中的不一定是邻域解中最优的

5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$

6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

关于T

当T为∞时（注意不是t为∞），概率为1，即无条件接纳邻域解

随着T减小，概率减小

当T趋近于0时，概率趋近于0，即仅接纳比当前解更优的邻域解

模拟退火算法

模拟退火算法（求解最大化问题）

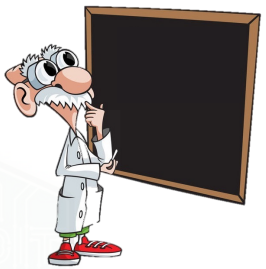
0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$　注意ΔE小于等于0

关于ΔE
- 当ΔE为0时，概率为1，即接纳邻域解
- 随着ΔE减小，概率减小
- 当ΔE为-∞时，概率趋近于0，即仅接纳比当前解更优的邻域解

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$



可以理论证明，在满足一系列条件时，当T趋近于0时，$\boldsymbol{x}_{\text{current}}$能收敛到全局最优解

证明思路

第一步：分析在一个固定的温度值T下，当算法执行时间足够长时，解空间中的每个解成为x_{current}的概率

例如，解空间中有4个解。如果固定$T = 10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为x_{current}的概率？

$f = 10$ ①	$f = 15$ ②
$f = 4$ ③	$f = 6$ ④

证明思路

第一步：分析在一个固定的温度值T下，当算法执行时间足够长时，解空间中的每个解成为$x_{current}$的概率

例如，解空间中有4个解。如果固定$T = 10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为$x_{current}$的概率？

$f = 10$ ①	$f = 15$ ②
$f = 4$ ③	$f = 6$ ④

第二步：求证当温度值T趋近于0时，最优解成为$x_{current}$的概率趋近于1

证明思路

例如，解空间中有4个解。如果固定$T=10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为x_{current}的概率？

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

假设选择上/下/左/右一格范围为邻域

当前解是1号解时，下一个回合的当前解是2/3/4/1号解的条件概率：

$$\Pr\left(x_{\text{current}}^{t+1}=2\middle|x_{\text{current}}^{t}=1\right)=\frac{1}{2}$$

$$\Pr\left(x_{\text{current}}^{t+1}=3\middle|x_{\text{current}}^{t}=1\right)=\frac{1}{2}e^{-\frac{6}{10}}$$

$$\Pr\left(x_{\text{current}}^{t+1}=4\middle|x_{\text{current}}^{t}=1\right)=0$$

$$\Pr\left(x_{\text{current}}^{t+1}=1\middle|x_{\text{current}}^{t}=1\right)=1-\frac{1}{2}-\frac{1}{2}e^{-\frac{6}{10}}$$

证明思路

例如，解空间中有4个解。如果固定$T=10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为$\boldsymbol{x}_{\text{current}}$的概率？

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

当前解是2号解时，下一个回合的当前解是1/3/4/2号解的条件概率：

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=1\middle|\boldsymbol{x}_{\text{current}}^{t}=2\right)=\frac{1}{2}e^{-\frac{5}{10}}$$

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=3\middle|\boldsymbol{x}_{\text{current}}^{t}=2\right)=0$$

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=4\middle|\boldsymbol{x}_{\text{current}}^{t}=2\right)=\frac{1}{2}e^{-\frac{9}{10}}$$

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=2\middle|\boldsymbol{x}_{\text{current}}^{t}=2\right)=1-\frac{1}{2}e^{-\frac{5}{10}}-\frac{1}{2}e^{-\frac{9}{10}}$$

证明思路

例如，解空间中有4个解。如果固定$T = 10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为$\boldsymbol{x}_{\text{current}}$的概率？

$f = 10$ ①	$f = 15$ ②
$f = 4$ ③	$f = 6$ ④

当前解是3号解时，下一个回合的当前解是1/2/4/3号解的条件概率：

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1} = 1 \middle| \boldsymbol{x}_{\text{current}}^{t} = 3\right) = \frac{1}{2}$$
$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1} = 2 \middle| \boldsymbol{x}_{\text{current}}^{t} = 3\right) = 0$$
$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1} = 4 \middle| \boldsymbol{x}_{\text{current}}^{t} = 3\right) = \frac{1}{2}$$
$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1} = 3 \middle| \boldsymbol{x}_{\text{current}}^{t} = 3\right) = 0$$

证明思路

例如，解空间中有4个解。如果固定$T=10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为$\boldsymbol{x}_{\text{current}}$的概率？

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

当前解是4号解时，下一个回合的当前解是1/2/3/4号解的条件概率：

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=1\middle|\boldsymbol{x}_{\text{current}}^{t}=4\right)=0$$

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=2\middle|\boldsymbol{x}_{\text{current}}^{t}=4\right)=\frac{1}{2}$$

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=3\middle|\boldsymbol{x}_{\text{current}}^{t}=4\right)=\frac{1}{2}e^{-\frac{2}{10}}$$

$$\Pr\left(\boldsymbol{x}_{\text{current}}^{t+1}=4\middle|\boldsymbol{x}_{\text{current}}^{t}=4\right)=1-\frac{1}{2}-\frac{1}{2}e^{-\frac{2}{10}}$$

证明思路

例如，解空间中有4个解。如果固定$T=10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为x_{current}的概率？

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

$$\mathbf{P}=\begin{bmatrix} \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=4) \\ \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=4) \\ \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=4) \\ \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=4) \end{bmatrix}$$

$$\begin{bmatrix} \Pr(x_{\text{current}}^{t+1}=1) \\ \Pr(x_{\text{current}}^{t+1}=2) \\ \Pr(x_{\text{current}}^{t+1}=3) \\ \Pr(x_{\text{current}}^{t+1}=4) \end{bmatrix} = \mathbf{P} \begin{bmatrix} \Pr(x_{\text{current}}^{t}=1) \\ \Pr(x_{\text{current}}^{t}=2) \\ \Pr(x_{\text{current}}^{t}=3) \\ \Pr(x_{\text{current}}^{t}=4) \end{bmatrix}$$

证明思路

例如，解空间中有4个解。如果固定$T=10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为x_{current}的概率？

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

$$\mathbf{P}=\begin{bmatrix} \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=1|x_{\text{current}}^{t}=4) \\ \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=2|x_{\text{current}}^{t}=4) \\ \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=3|x_{\text{current}}^{t}=4) \\ \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=1) & \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=2) & \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=3) & \Pr(x_{\text{current}}^{t+1}=4|x_{\text{current}}^{t}=4) \end{bmatrix}$$

$$\begin{bmatrix} \Pr(x_{\text{current}}^{100000}=1) \\ \Pr(x_{\text{current}}^{100000}=2) \\ \Pr(x_{\text{current}}^{100000}=3) \\ \Pr(x_{\text{current}}^{100000}=4) \end{bmatrix} = \mathbf{P}^{100000} \begin{bmatrix} \Pr(x_{\text{current}}^{0}=1) \\ \Pr(x_{\text{current}}^{0}=2) \\ \Pr(x_{\text{current}}^{0}=3) \\ \Pr(x_{\text{current}}^{0}=4) \end{bmatrix}$$

证明思路

例如，解空间中有4个解。如果固定$T = 10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为$\boldsymbol{x}_{\text{current}}$的概率？

$f = 10$ ①	$f = 15$ ②
$f = 4$ ③	$f = 6$ ④

Users ▸ haoranyu ▸ Documents ▸ MATLAB ▸

Command Window

```
>> clear all;
close all;

T=10;
p11=1-0.5-0.5*exp(-6/T);
p21=0.5;
p31=0.5*exp(-6/T);
p41=0;

p12=0.5*exp(-5/T);
p22=1-0.5*exp(-5/T)-0.5*exp(-9/T);
p32=0;
p42=0.5*exp(-9/T);

p13=0.5;
p23=0;
p33=0;
p43=0.5;

p14=0;
p24=0.5;
p34=0.5*exp(-2/10);
p44=1-0.5-0.5*exp(-2/10);

P=[p11, p12, p13, p14; p21, p22, p23, p24; p31, p32, p33, p34; p41, p42, p43, p44];

distribution=P^100000*[1/4; 1/4; 1/4; 1/4]

distribution =

    0.2585
    0.4263
    0.1419
    0.1733
```

$$\begin{bmatrix} \Pr(\boldsymbol{x}_{\text{current}}^{100000} = 1) \\ \Pr(\boldsymbol{x}_{\text{current}}^{100000} = 2) \\ \Pr(\boldsymbol{x}_{\text{current}}^{100000} = 3) \\ \Pr(\boldsymbol{x}_{\text{current}}^{100000} = 4) \end{bmatrix} = \begin{bmatrix} 0.2585 \\ 0.4263 \\ 0.1419 \\ 0.1733 \end{bmatrix}$$

证明思路

例如，解空间中有4个解。如果固定$T=10$，调用模拟退火算法执行10^5次（t从1增长到10^5）对这4个解分别求其成为x_{current}的概率？

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

```
>> q1=exp(1);
>> q2=exp(15/10);
>> q3=exp(4/10);
>> q4=exp(6/10);
>> redvector=1/(q1+q2+q3+q4)*[q1;q2;q3;q4]

redvector =

    0.2585
    0.4263
    0.1419
    0.1733
```

Users ▸ haoranyu ▸ Documents ▸ MATLAB ▸

Command Window

```
>> clear all;
close all;

T=10;
p11=1-0.5-0.5*exp(-6/T);
p21=0.5;
p31=0.5*exp(-6/T);
p41=0;

p12=0.5*exp(-5/T);
p22=1-0.5*exp(-5/T)-0.5*exp(-9/T);
p32=0;
p42=0.5*exp(-9/T);

p13=0.5;
p23=0;
p33=0;
p43=0.5;

p14=0;
p24=0.5;
p34=0.5*exp(-2/10);
p44=1-0.5-0.5*exp(-2/10);

P=[p11, p12, p13, p14; p21, p22, p23, p24; p31, p32, p33, p34; p41, p42, p43, p44];

distribution=P^100000*[1/4; 1/4; 1/4; 1/4]

distribution =

    0.2585
    0.4263
    0.1419
    0.1733
```

$$\begin{bmatrix}\Pr(x_{\text{current}}^{100000}=1)\\ \Pr(x_{\text{current}}^{100000}=2)\\ \Pr(x_{\text{current}}^{100000}=3)\\ \Pr(x_{\text{current}}^{100000}=4)\end{bmatrix}=\begin{bmatrix}0.2585\\0.4263\\0.1419\\0.1733\end{bmatrix}\approx\frac{1}{\text{归一化因子}}\begin{bmatrix}e^{\frac{10}{10}}\\e^{\frac{15}{10}}\\e^{\frac{4}{10}}\\e^{\frac{6}{10}}\end{bmatrix}$$

证明思路

在给定T时，在各解之间跳变的过程可以由如下转移概率表征：

$$P_{ij}(T)=\begin{cases} G_{ij}(T)A_{ij}(T), & i\neq j, \\ 1-\sum_{l\in S, l\neq i} G_{il}(T)A_{il}(T), & i=j. \end{cases}$$

解i到解j的转移概率

$$G_{ij}(T)=\begin{cases} \dfrac{1}{|N(i)|} & j\in N(i), \\ 0, & otherwise. \end{cases}$$

当前解是i时，解j被选中的概率

$$A_{ij}(T)=\begin{cases} 1, & Val(j)>Val(i), \\ e^{(Val(j)-Val(i))/T}, & otherwise. \end{cases}$$

当前解是i且解j被选中时，i转移到j的概率

图片取自Kathryn A. Dowsland, Jonathan M. Thompson, "Simulated Annealing," Handbook of Natural Computing

证明思路

$$P_{ij}(T)=\begin{cases} G_{ij}(T)A_{ij}(T), & i\neq j, \\ 1-\sum_{l\in S,l\neq i} G_{il}(T)A_{il}(T), & i=j. \end{cases}$$

解i到解j的转移概率

利用马尔可夫链的相关知识，可证明在一定条件下*，各个解成为当前解的概率存在唯一稳态分布

* 解空间中各个解之间是强互联关系。即给定一个解，从任意解可以不断沿着邻域解组成的路径达到该解

证明思路

$$P_{ij}(T)=\begin{cases} G_{ij}(T)A_{ij}(T), & i\neq j, \\ 1-\sum_{l\in S, l\neq i} G_{il}(T)A_{il}(T), & i=j. \end{cases}$$

解i到解j的转移概率

利用马尔可夫链的相关知识，可证明在一定条件下*，各个解成为当前解的概率存在唯一稳态分布

$$Q_T(i)=\frac{\exp\left(\frac{Val(i)}{T}\right)}{\sum_{j\in S}\exp\left(\frac{Val(j)}{T}\right)}$$

* 解空间中各个解之间是强互联关系。即给定一个解，从任意解可以不断沿着邻域解组成的路径达到该解

证明思路

$$P_{ij}(T)=\begin{cases} G_{ij}(T)A_{ij}(T), & i\neq j, \\ 1-\sum_{l\in S, l\neq i} G_{il}(T)A_{il}(T), & i=j. \end{cases}$$

解i到解j的转移概率

利用马尔可夫链的相关知识，可证明在一定条件下*，各个解成为当前解的概率存在唯一稳态分布

$$Q_T(i)=\frac{\exp\left(\frac{Val(i)}{T}\right)}{\sum_{j\in S}\exp\left(\frac{Val(j)}{T}\right)}$$

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

$$\begin{bmatrix} \Pr(x_{current}^{100000}=1) \\ \Pr(x_{current}^{100000}=2) \\ \Pr(x_{current}^{100000}=3) \\ \Pr(x_{current}^{100000}=4) \end{bmatrix} = \begin{bmatrix} 0.2585 \\ 0.4263 \\ 0.1419 \\ 0.1733 \end{bmatrix} \approx \frac{1}{\text{归一化因子}} \begin{bmatrix} e^{\frac{10}{10}} \\ e^{\frac{15}{10}} \\ e^{\frac{4}{10}} \\ e^{\frac{6}{10}} \end{bmatrix}$$

* 解空间中各个解之间是强互联关系。即给定一个解，从任意解可以不断沿着邻域解组成的路径达到该解

证明思路

$$Q_T(i)=\frac{\exp\left(\frac{Val(i)}{T}\right)}{\sum_{j\in S}\exp\left(\frac{Val(j)}{T}\right)}$$

当温度值T趋近于0时？

证明思路

$$Q_T(i)=\frac{\exp\left(\frac{Val(i)}{T}\right)}{\sum_{j\in S}\exp\left(\frac{Val(j)}{T}\right)}$$

当温度值T趋近于0时？

$T=10$

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

$$\begin{bmatrix}\Pr(x^{100000}_{current}=1)\\ \Pr(x^{100000}_{current}=2)\\ \Pr(x^{100000}_{current}=3)\\ \Pr(x^{100000}_{current}=4)\end{bmatrix}=\begin{bmatrix}0.2585\\0.4263\\0.1419\\0.1733\end{bmatrix}\approx\frac{1}{\text{归一化因子}}\begin{bmatrix}e^{\frac{10}{10}}\\e^{\frac{15}{10}}\\e^{\frac{4}{10}}\\e^{\frac{6}{10}}\end{bmatrix}$$

$T=1$

$f=10$ ①	$f=15$ ②
$f=4$ ③	$f=6$ ④

$$\begin{bmatrix}\Pr(x^{100000}_{current}=1)\\ \Pr(x^{100000}_{current}=2)\\ \Pr(x^{100000}_{current}=3)\\ \Pr(x^{100000}_{current}=4)\end{bmatrix}=\begin{bmatrix}0.0067\\0.9931\\0.0000\\0.0001\end{bmatrix}\approx\frac{1}{\text{归一化因子}}\begin{bmatrix}e^{\frac{10}{1}}\\e^{\frac{15}{1}}\\e^{\frac{4}{1}}\\e^{\frac{6}{1}}\end{bmatrix}$$

证明思路

$$Q_T(i)=\frac{\exp\left(\frac{Val(i)}{T}\right)}{\sum_{j\in S}\exp\left(\frac{Val(j)}{T}\right)}$$

当温度值T趋近于0时？

当在每个T温度下，在各种解之间的跳变都到达到稳态分布时，算法将逐渐趋近于最优解（注意是在一定理论条件满足的情况下达到全局最优解）

证明思路

第一步：分析在一个固定的温度值T下，当算法执行时间足够长时，解空间中的每个解成为x_{current}的概率

第二步：求证当温度值T趋近于0时，最优解成为x_{current}的概率趋近于1

完整证明需要考虑T逐步减小的过程，即非齐次马尔可夫链

T随时间变化需要满足一定条件（T不能下降得太快）

[BOOK] **Simulated annealing** and **Boltzmann machines**: a stochastic approach to combinatorial optimization and neural computing

E **Aarts**, J **Korst** - **1989** - dl.acm.org

Large combinatorial optimization problems (the most famous example is the traveling salesman problem) are notably difficult because the number of feasible configurations grows exponentially with the size of the problem. Algorithms exist that require fewer calculations and find the optimum solution most of the time or a good suboptimal solution, but they are hard to design and depend strongly upon the problem. A stochastic (ie, Monte Carlo) approach is an attractive alternative, for the same reasons that such an approach is applied …

☆ Save 99 Cite Cited by 5706 Related articles All 8 versions »

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

哪些需要人为设置的？

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

一、起始温度：需要足够高以使得所有的比当前解差的邻域解都能以一个足够高的概率（如0.98）被接纳

模拟退火算法

一、起始温度：需要足够高以使得所有的比当前解差的邻域解都能以一个足够高的概率被接纳

若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

模拟退火算法

一、起始温度：需要足够高以使得所有的比当前解差的邻域解都能以一个足够高的概率被接纳

若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

T_0=(-所有“下山”跳变的质量下降程度的平均值)/ln(需要达到的概率)

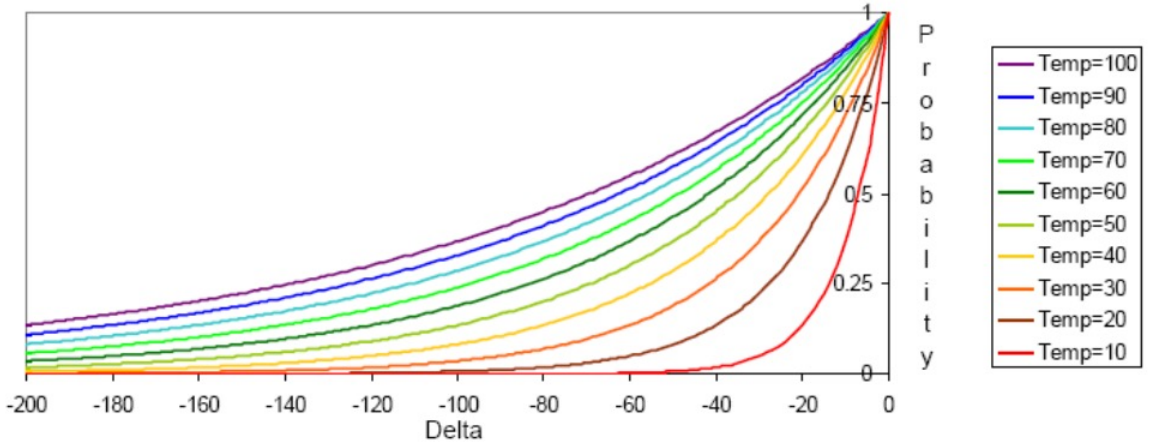
↓

在实际中用在解空间上的随机游走估测该项目

或者可以（从小到大）尝试不同起始值，观察在起始温度下邻域解被接纳的比例，选择使接纳比例较大（比如接近于1）的初始值

模拟退火算法

一、起始温度：需要足够高以使得所有的比当前解差的邻域解都能以一个足够高的概率被接纳



Initially temperature is very high (most bad moves accepted)

Temp slowly goes to 0, with multiple moves attempted at each temperature

Final runs with temp=0 (always reject bad moves) greedily “quench” the system

图片取自Scott Hauck <UW EE 541: Automated Layout of Integrated Circuits>

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

二、降温过程：需要相对缓慢

模拟退火算法

二、降温过程：

（1）温度取值

常见几何递减：$T_k = T_0\alpha^k$ (α一般取0.8到0.99之间）

另外还有线性、对数递减等不同选择

（2）每个温度下执行的次数（每个温度下不止一次跳变）

尽量确保在每个温度下，当前解的邻域解大都能被随机选择到

The value of *nrep* is often related to the size of the neighborhood, and may vary from temperature to temperature. For example, it is important to spend enough time at low temperatures to ensure that the regions around a local optimum have been fully explored.

模拟退火算法

模拟退火算法（求解最大化问题）

0 初始化当前解$\boldsymbol{x}_{\text{current}}$
1 for t=1 to ∞:
2 　根据温度下降规则和t，确定当前温度T
3 　若T=0，终止算法，输出$\boldsymbol{x}_{\text{current}}$
4 　随机选择一个邻域解$\boldsymbol{x}_{\text{neighbor}}$
5 　计算$\Delta E = f(\boldsymbol{x}_{\text{neighbor}}) - f(\boldsymbol{x}_{\text{current}})$
6 　若$\Delta E > 0$，则令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$
7 　若$\Delta E \leq 0$，则以$e^{\Delta E/T}$的概率令$\boldsymbol{x}_{\text{current}} \leftarrow \boldsymbol{x}_{\text{neighbor}}$

三、**结束条件**：理论上等温度降为0，但实际可能需要时间过长、且早已达到理想的解

模拟退火算法

三、结束条件：

—— 如果连续不接纳邻域解的次数超过限制

—— 如果最近若干解的质量的平均值几乎不再上升

—— acceptance ratio（即接纳邻域解次数/总尝试次数）低于一定值

……

旅行商问题

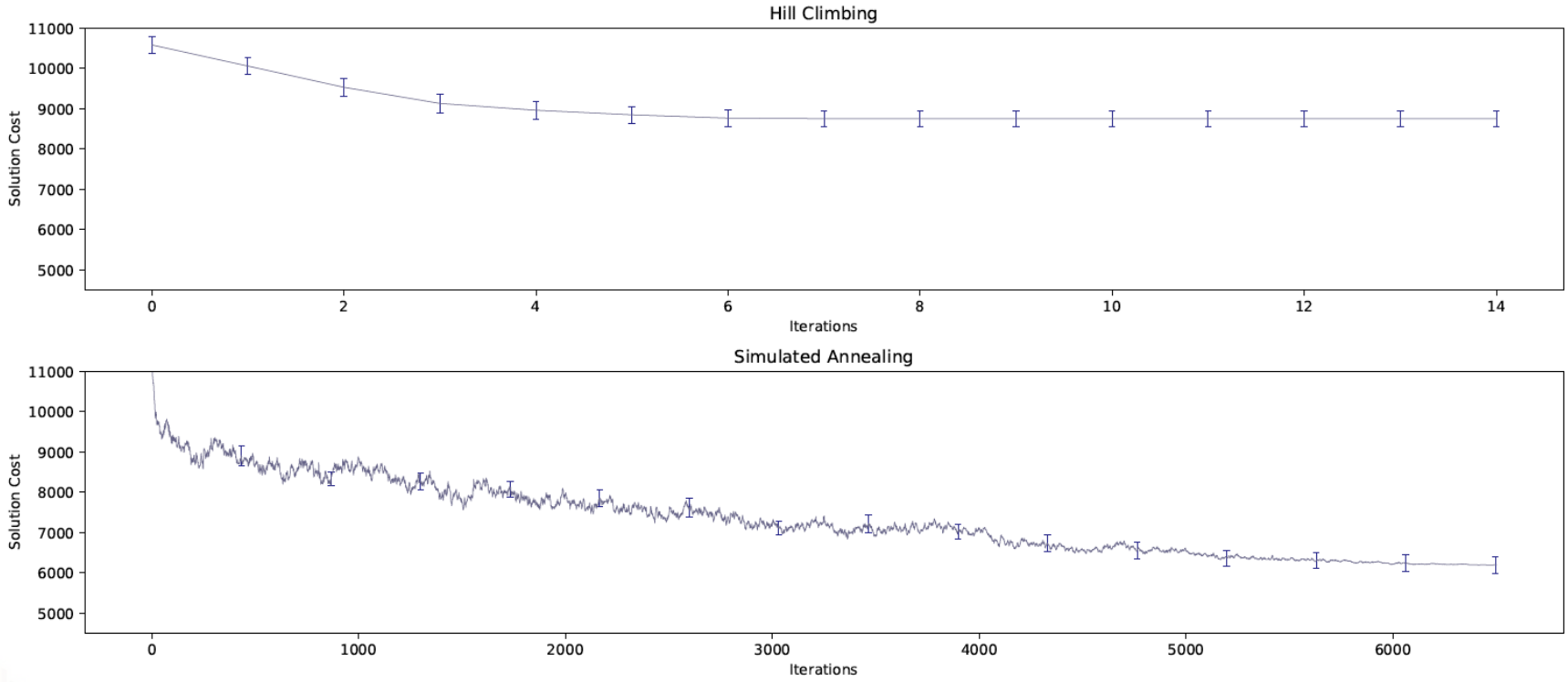
用模拟退火算法求解旅行商问题

- *proposal mechanism:* uniform random choice from 2-exchange neighbourhood; 定义邻域
- *acceptance criterion:* Metropolis condition (always accept improving steps, accept worsening steps with probability $\exp[(f(s) - f(s'))/T]$); 注意旅行商问题是最小化问题
- *annealing schedule*: geometric cooling $T := 0.95 \cdot T$ with $n \cdot (n-1)$ steps at each temperature (n = number of vertices in given graph), T_0 chosen such that 97% of proposed steps are accepted; 起始温度、降温过程
- *termination:* when for five successive temperature values no improvement in solution quality and acceptance ratio $< 2\%$. 结束条件

例子取自Thomas Stutzle, Heuristic Optimization 2014

旅行商问题

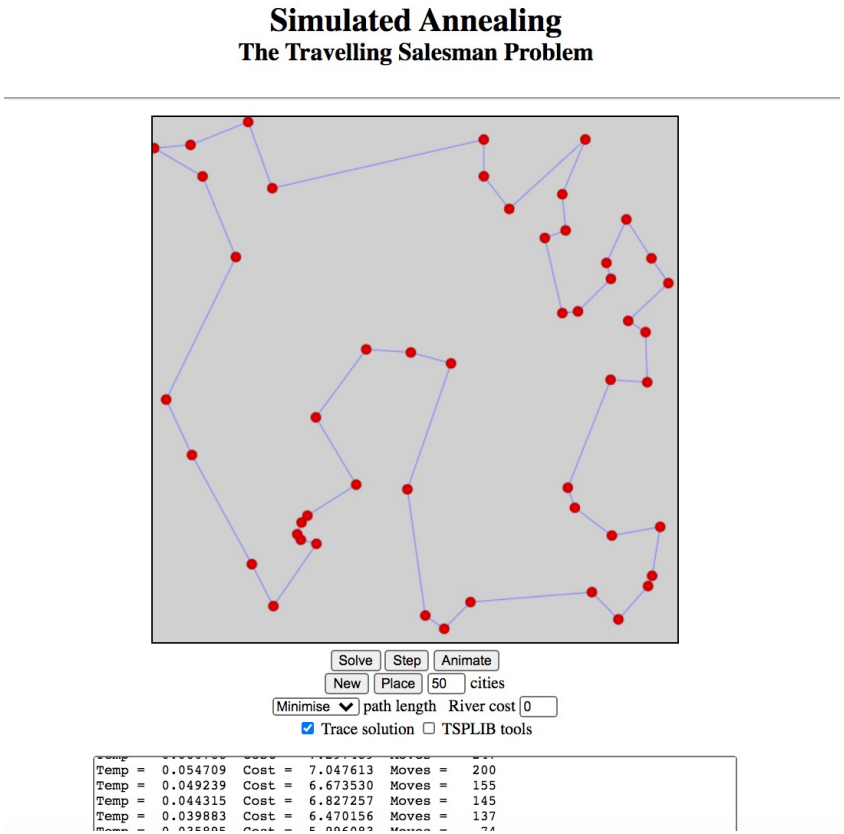
上讲中15个城市旅行商问题



旅行商问题

求解旅行商问题演示

https://www.fourmilab.ch/documents/travelling/anneal/



0-1背包问题

如何考虑约束条件？

$$\max f(x) = \sum_{i=1}^{n} v_i x_i$$

$$\sum_{i=1}^{n} w_i . x_i . \leq P.$$

$$x_i \in \{0,1\}, i = 1,2,\ldots,n$$

0-1背包问题

如何考虑约束条件？

$$\max f(x) = \sum_{i=1}^{n} v_i x_i$$

$$\sum_{i=1}^{n} w_i . x_i . \leq P.$$

$$x_i \in \{0,1\}, i = 1,2,\ldots,n$$

将解的质量定义为

$$\sum_{i=1}^{n} v_i x_i - \mu \frac{\max\left\{0, \left(\sum_{i=1}^{n} w_i x_i\right) - P\right\}}{P}$$

0-1背包问题

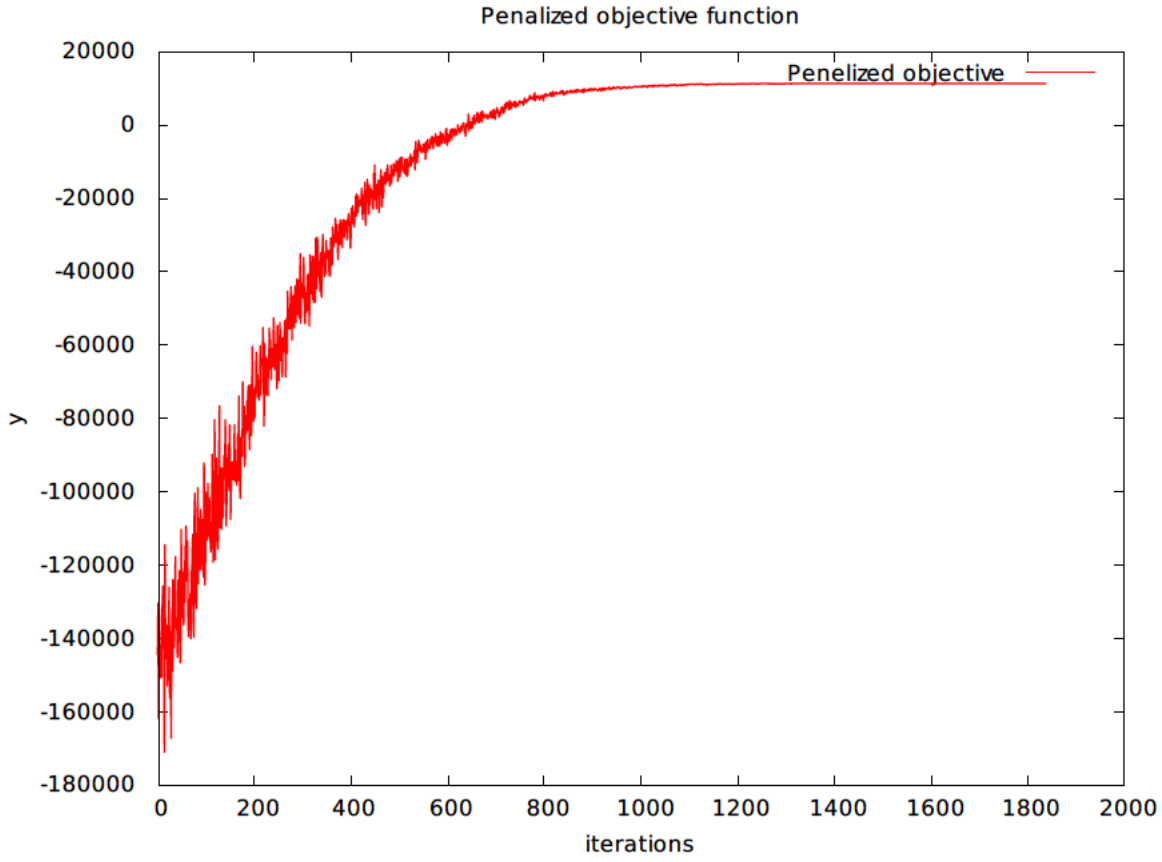
$$\sum_{i=1}^{n} v_i x_i - \mu \frac{\max\left\{0, \left(\sum_{i=1}^{n} w_i x_i\right) - P\right\}}{P}$$

In order to test the simulated annealing algorithm on this problem, we first build an instance of the problem by randomly generating 100 objects for which the weights have also been selected randomly between 1 and 100 with a uniform probability density function. For this instance, the capacity of the bag is set to $P = 2000$. We choose $\mu = 1$ for the penalty parameter and we apply the basic SA algorithm with the initial temperature set to a value of c_0 such that $\chi(c) = 0.8$, a geometric cooling schedule with $\alpha = 0.995$, and $L_k = 1{,}000$ for every iteration k. The algorithm is stopped when the temperature reaches $\frac{c_o}{1{,}000}$.

例子取自Delahaye, et al, “Simulated annealing: From basics to applications”

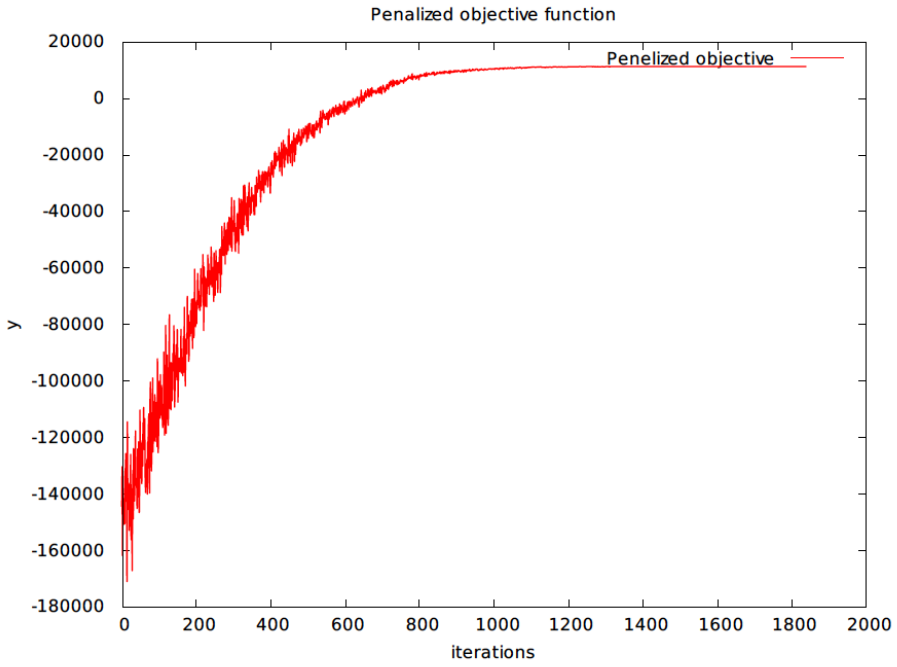
0-1背包问题

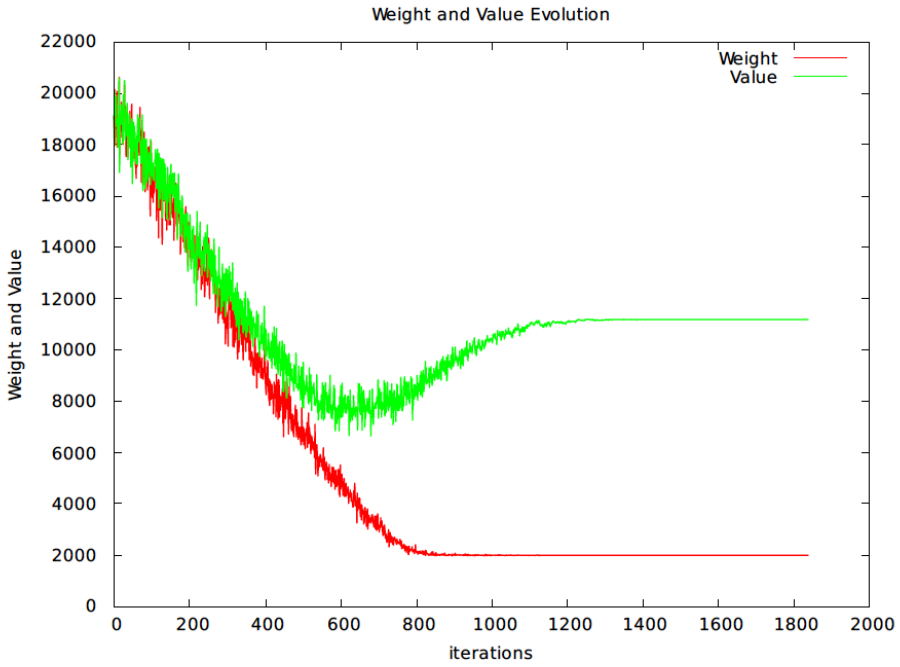
$$\sum_{i=1}^{n} v_i x_i - \mu \frac{\max\left\{0, \left(\sum_{i=1}^{n} w_i x_i\right) - P\right\}}{P}$$



0-1背包问题

$$\sum_{i=1}^{n} v_i x_i - \mu \frac{\max\left\{0, \left(\sum_{i=1}^{n} w_i x_i\right) - P\right\}}{P}$$





背包承重上限是2000

（二进制）遗传算法

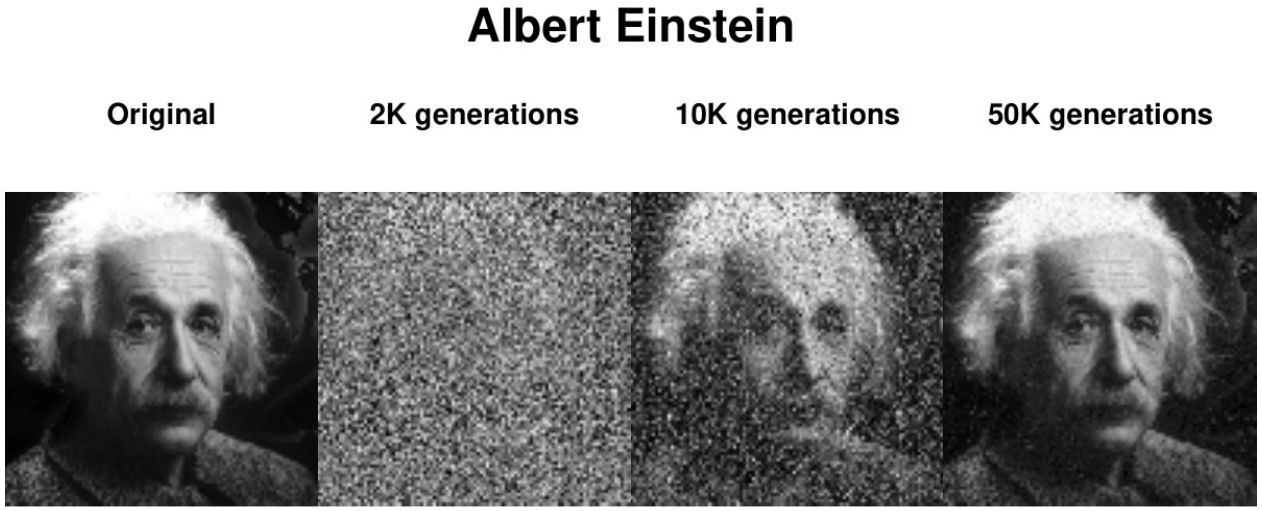
Part1 如何从复杂度的角度衡量算法好坏？
Part2 针对一些有特定形式的最优化问题，如何找到最优解？
Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？
局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法
Part4 针对较复杂的多目标最优化问题，如何找到较好解？

进化算法

通过模仿自然界生物遗传、选择、变异等过程求解问题

代表性算法：遗传算法（Genetic Algorithm，GA）

遗传算法



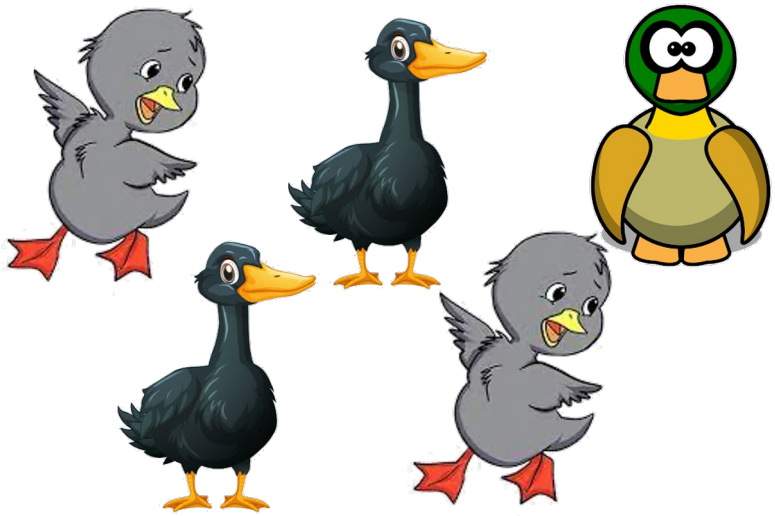
遗传算法酷炫特效——图片再生成（by Frédéric MARQUER）

https://www.bilibili.com/video/BV1js411r7NK?from=search&seid=14798324289557359101

遗传算法

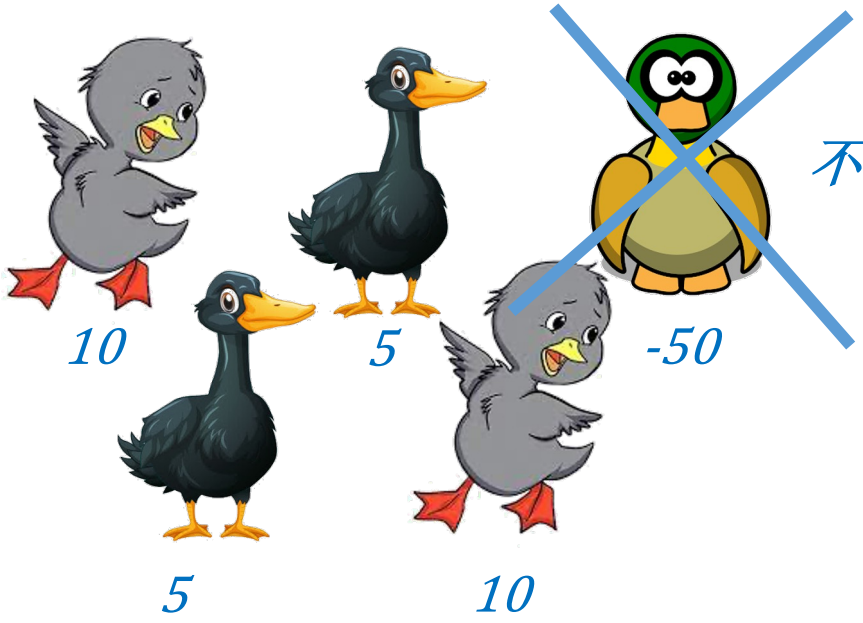
从前有个鸭子村...

第一年



故事修改自B站UP主“笔记鲨”【10分钟算法】遗传算法-带例子和动画/Genetic Algorithm

遗传算法



故事修改自B站UP主“笔记鲨”【10分钟算法】遗传算法-带例子和动画/Genetic Algorithm

遗传算法

10

1	0	1	1

5

0	0	1	0

分数只是外在表达
由内在基因决定

故事修改自B站UP主“笔记鲨”【10分钟算法】遗传算法-带例子和动画/Genetic Algorithm

遗传算法

10

5

1	1	1	1

0	0	1	0

基因交叉

0	0	1	1

1	1	1	0

基因突变

0	1	1	1

1	1	0	0

遗传算法

10

5

1	1	1	1

0	0	1	0

0	1	1	1

1	1	0	0

假设表达出分数为15、3

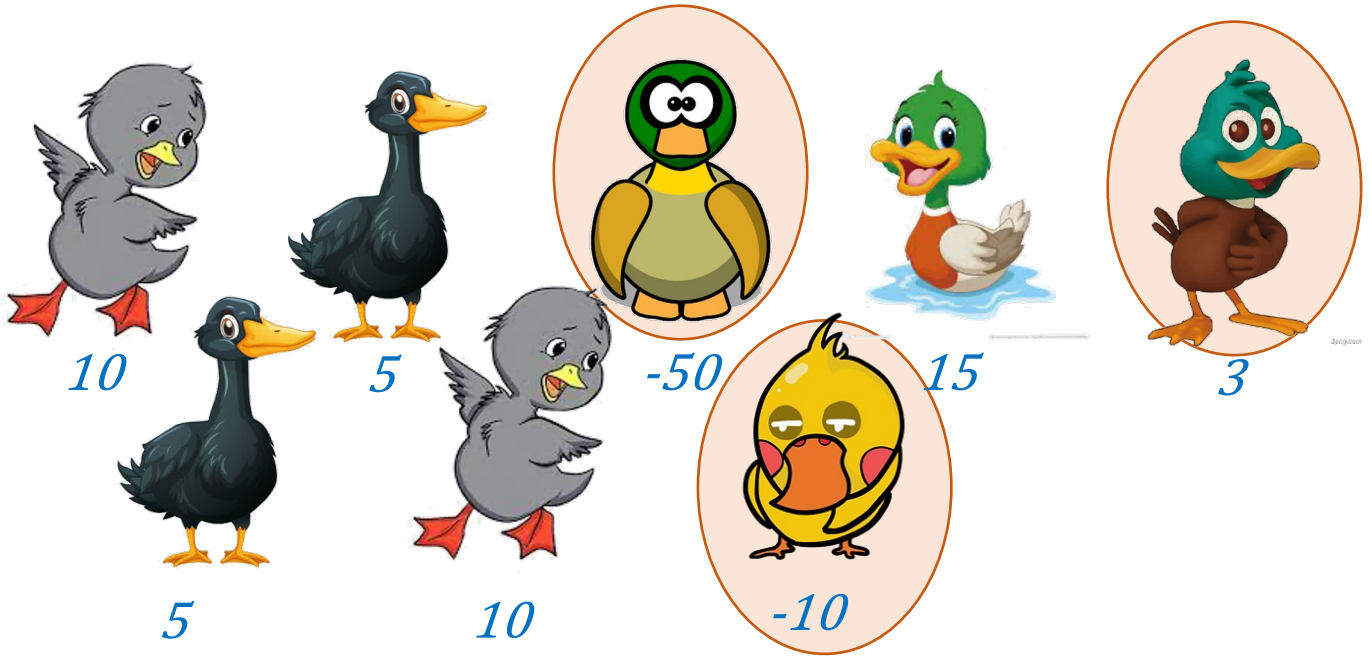
遗传算法

第一年

评测分数

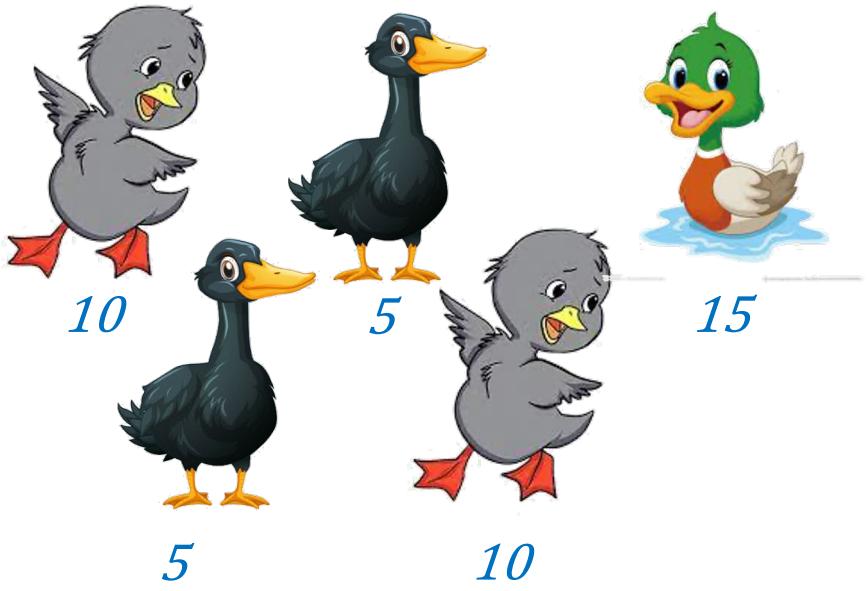
遗传算法

第一年



遗传算法

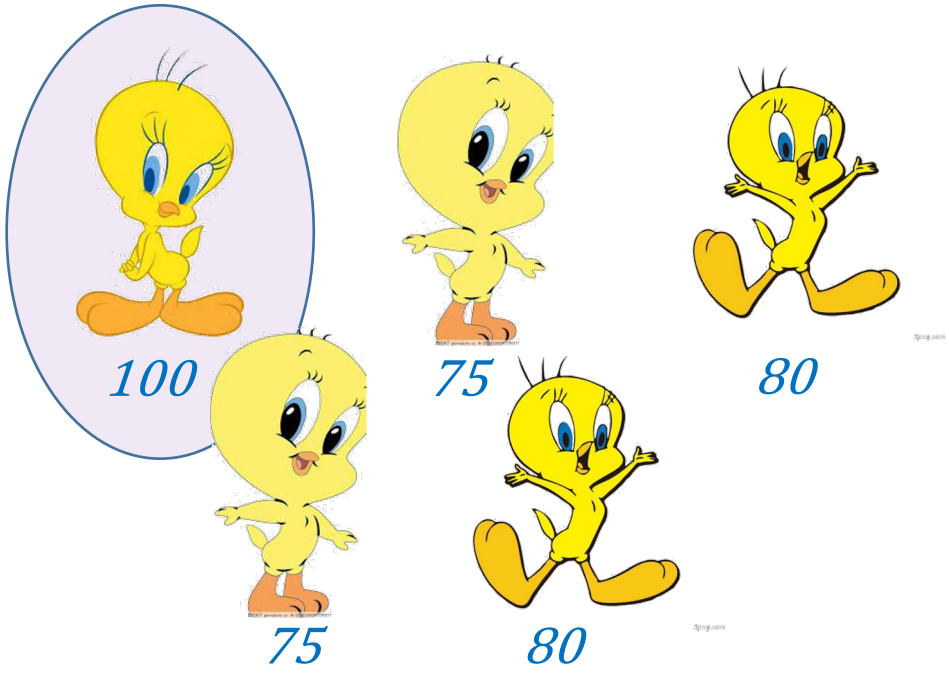
第二年



遗传算法

遗传算法

第N年



遗传算法

遗传算法

0 初始化当前种群$P(t)$ 即若干个解

1 评估当前种群$P(t)$ 计算其中每个解对应的目标函数值

2 for t=1 to ∞:

3 根据当前种群$P(t)$，确定父代种群$P_P(t)$

4 根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$

5 对子代种群$P_C(t)$进行基因突变

6 评估子代种群$P_C(t)$

7 根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$

8 检查循环终止条件，若满足则结束算法

所有的解需要有基因的表达形式

遗传算法

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$ 如何选择？
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

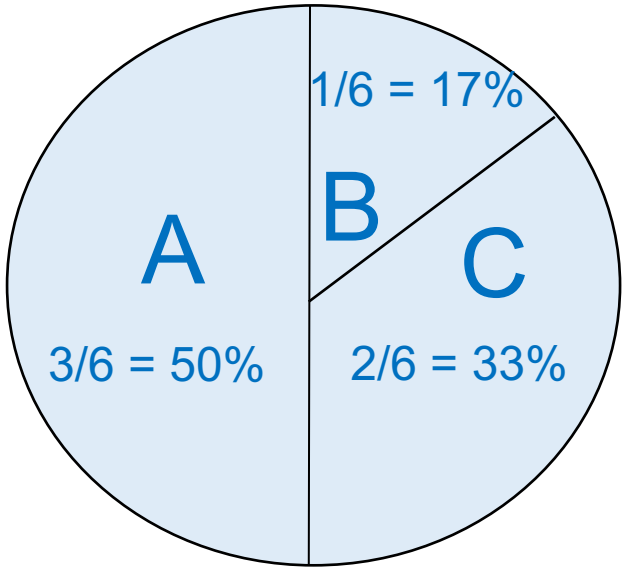
遗传算法

以转动轮盘的方式抽取若干个解，每次每个解被抽中的概率正比于该解的质量

解A的质量=3

解B的质量=1

解C的质量 =2



遗传算法

遗传算法

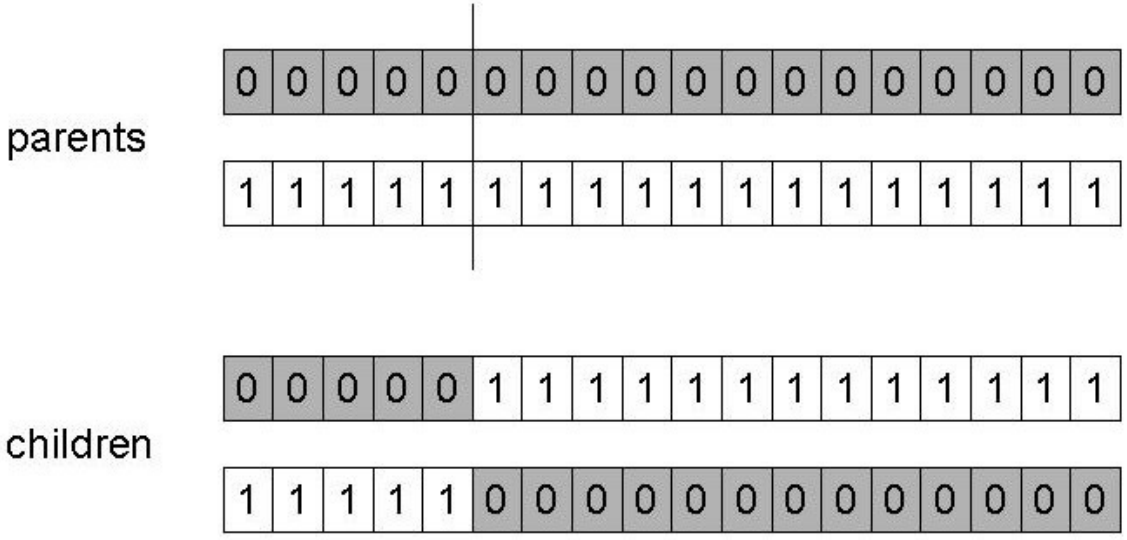
0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

先配对（顺序/随机）
再进行基因交叉

遗传算法

基因交叉

随机选取一个点，将基因序列分成两段，交叉生成后代基因序列

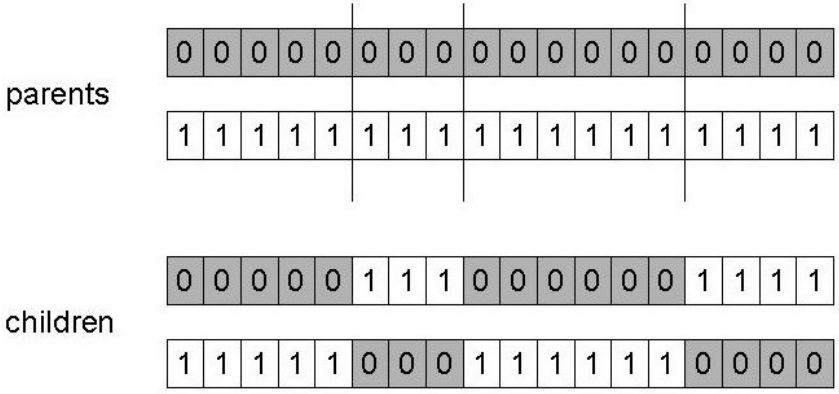


图片取自A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing课件

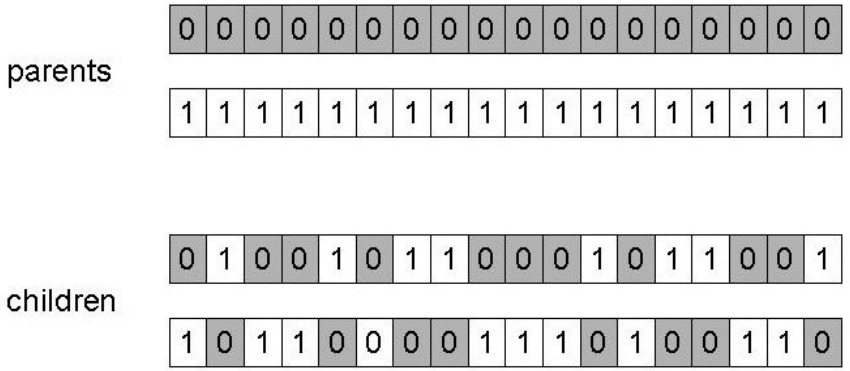
遗传算法

基因交叉（其它版本）

多点交叉



均匀交叉



图片取自A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing课件

遗传算法

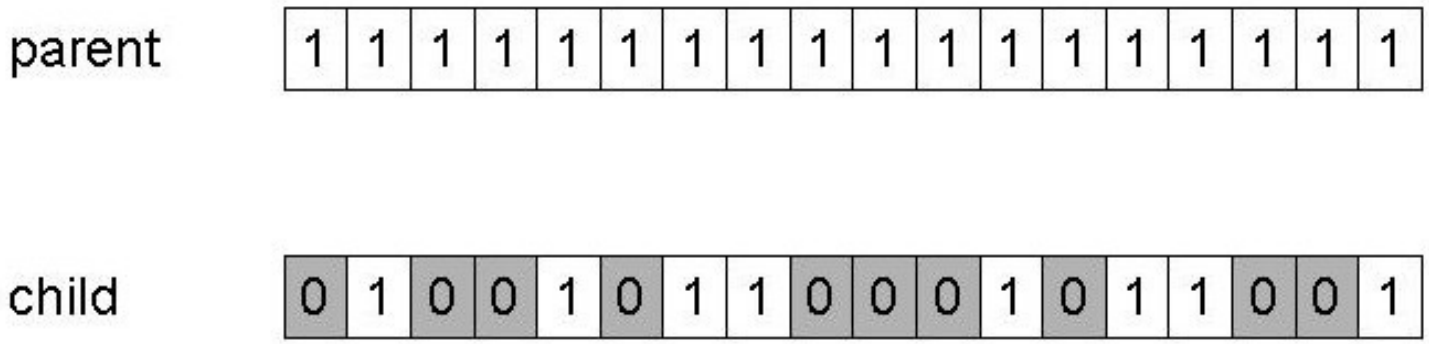
遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

遗传算法

基因突变

子代基因序列中每个基因独立地以p概率突变



图片取自A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing课件

例子

目标：从集合{0,1,…,31}中选择一个数从而使x^2最大

- 基因表示形式：例如 01101 ↔ 13
- 每轮考虑解的个数：4
- 初始解：随机初始化
- 选择允许配对繁殖的解：转动轮盘
- 基因交叉：单处切开基因序列、进行交叉
- 基因突变：每个基因以一定概率从0变为1或从1变为0

例子取自A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing课件

例子

目标：从集合{0,1,…,31}中选择一个数从而使x^2最大

- 基因表示形式：例如 01101 ↔ 13
- 每轮考虑解的个数：4
- 初始解：随机初始化
- 选择允许配对繁殖的解：转动轮盘

String no.	Initial population	x Value	Fitness $f(x) = x^2$	$Prob_i$	Expected count	Actual count
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1
Sum			1170	1.00	4.00	4
Average			293	0.25	1.00	1
Max			576	0.49	1.97	2

例子

目标：从集合{0,1,…,31}中选择一个数从而使x^2最大

- 基因交叉：单处切开基因序列、进行交叉

String no.	Mating pool	Crossover point	Offspring after xover	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 | 1	4	0 1 1 0 0	12	144
2	1 1 0 0 | 0	4	1 1 0 0 1	25	625
2	1 1 | 0 0 0	2	1 1 0 1 1	27	729
4	1 0 | 0 1 1	2	1 0 0 0 0	16	256
Sum					1754
Average					439
Max					729

例子

目标：从集合{0,1,…,31}中选择一个数从而使x^2最大

- 基因突变：每个基因以一定概率从0变为1或1变为0

String no.	Offspring after xover	Offspring after mutation	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 0	1 1 1 0 0	26	676
2	1 1 0 0 1	1 1 0 0 1	25	625
2	1 1 0 1 1	1 1 0 1 1	27	729
4	1 0 0 0 0	1 0 1 0 0	18	324
Sum				2354
Average				588.5
Max				729

旅行商问题

假设有9个城市，不能用0-1基因序列表达。如何做基因交叉和基因突变？

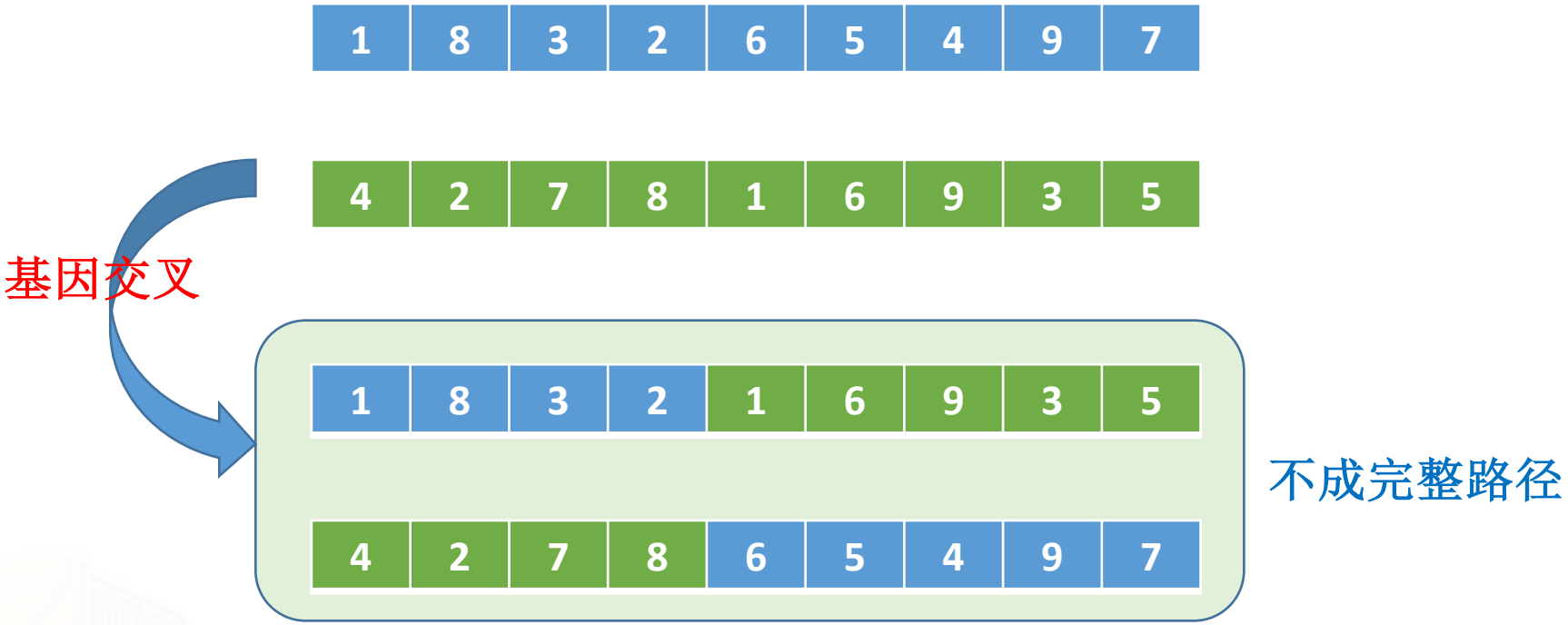
1	8	3	2	6	5	4	9	7

4	2	7	8	1	6	9	3	5

例子取自A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing课件

旅行商问题

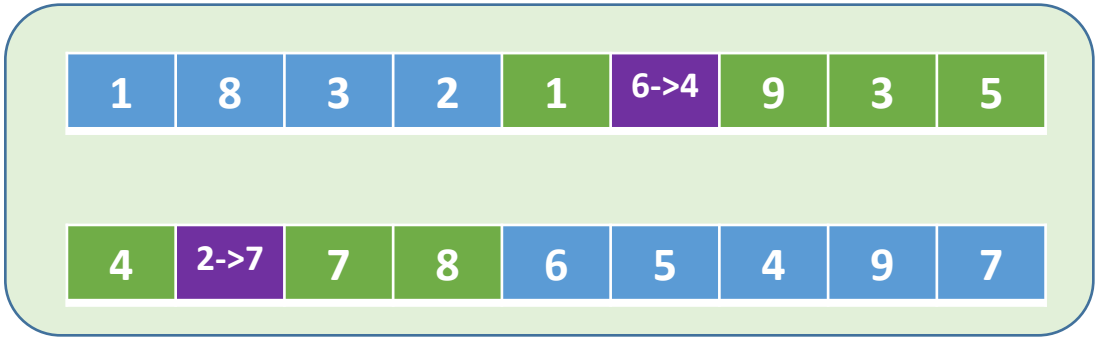
假设有9个城市，不能用0-1基因序列表达。如何做基因交叉和基因突变？



旅行商问题

假设有9个城市，不能用0-1基因序列表达。如何做基因交叉和基因突变？

基因突变



不成完整路径

旅行商问题

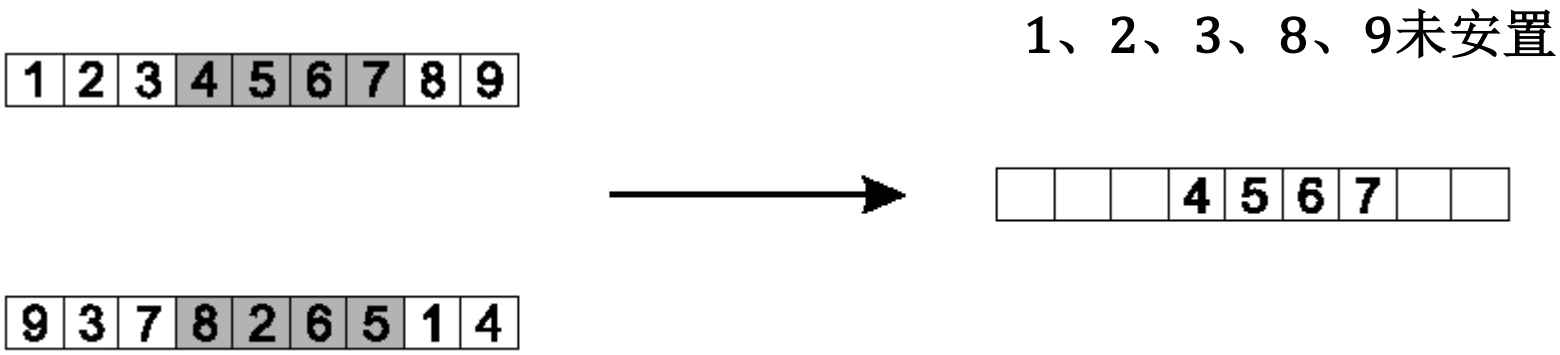
假设有9个城市，不能用0-1基因序列表达。如何做基因交叉？

旅行商问题

假设有9个城市，不能用0-1基因序列表达。如何做基因交叉？

先随机从父序列中选取一段，给第一个子代

找到父序列中未被选取的数子

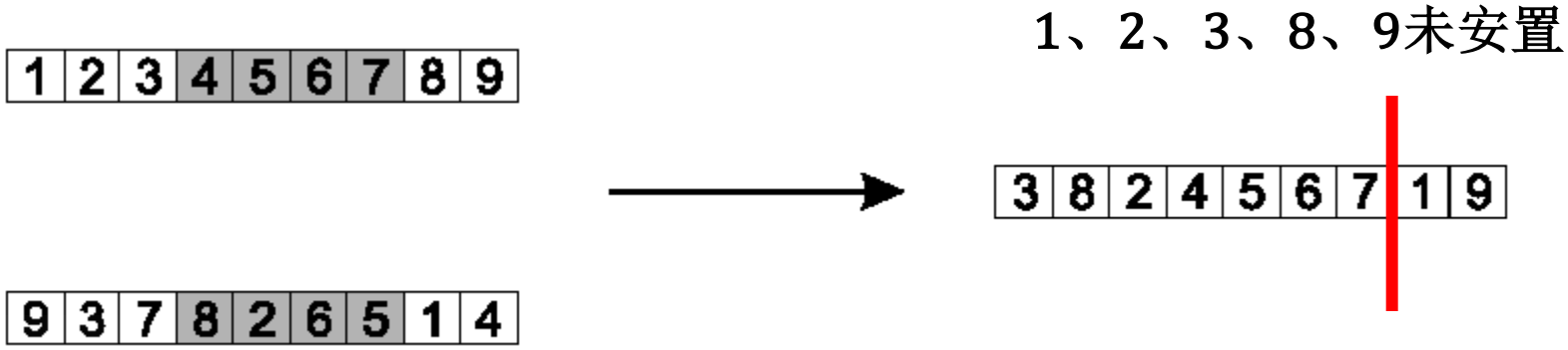


旅行商问题

假设有9个城市，不能用0-1基因序列表达。如何做基因交叉？

从第一个子代承继自父序列基因段的右边起，依照母序列的次序，将剩余数字置入

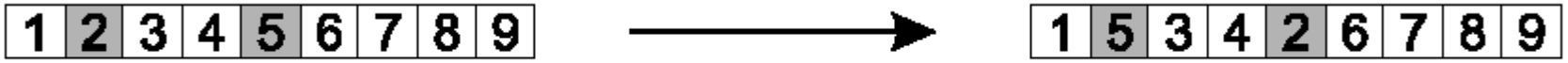
将父序列、母序列角色互换，同样方法生成第二个子代



旅行商问题

假设有9个城市，不能用0-1基因序列表达。如何做基因突变？

随机选择两个城市，互换位置



为什么不推荐这种做法？
会影响多对城市之间的相邻关系

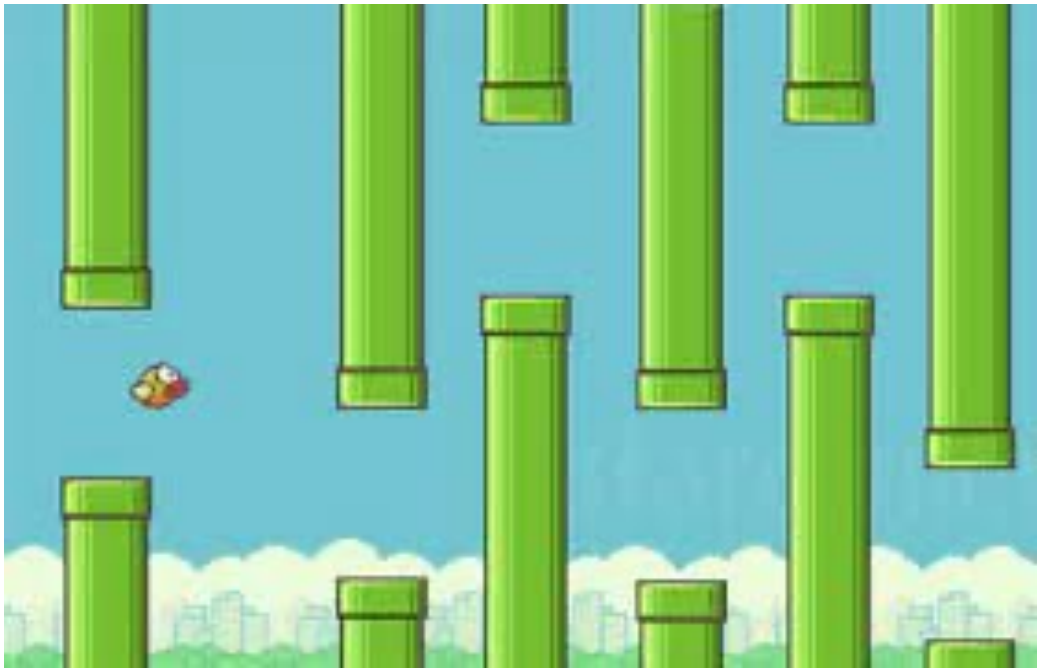
旅行商问题

假设有9个城市，不能用0-1基因序列表达。如何做基因突变？

随机选择两个城市，将第二个城市插到第一个城市之后

保护多对城市之间的相邻关系

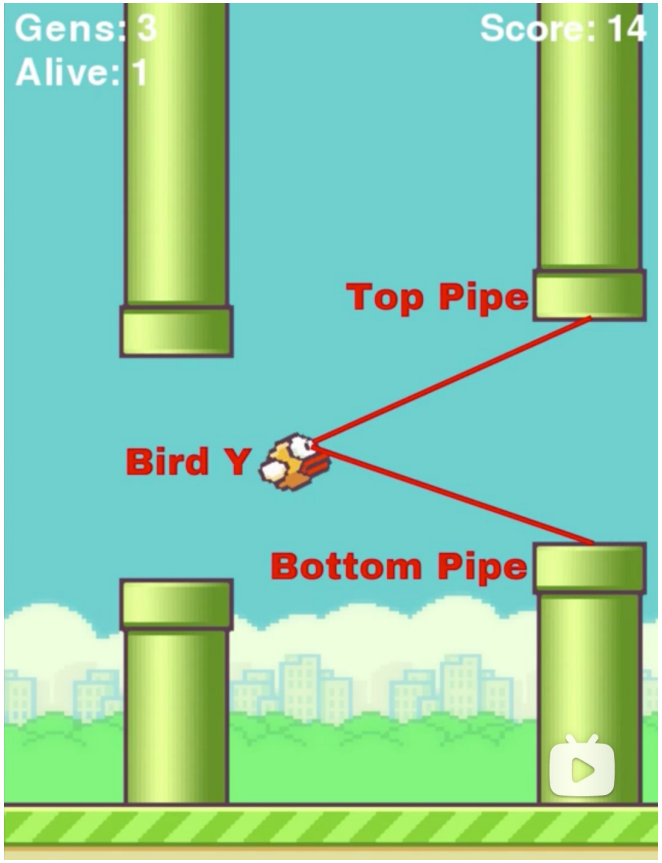
像素鸟游戏



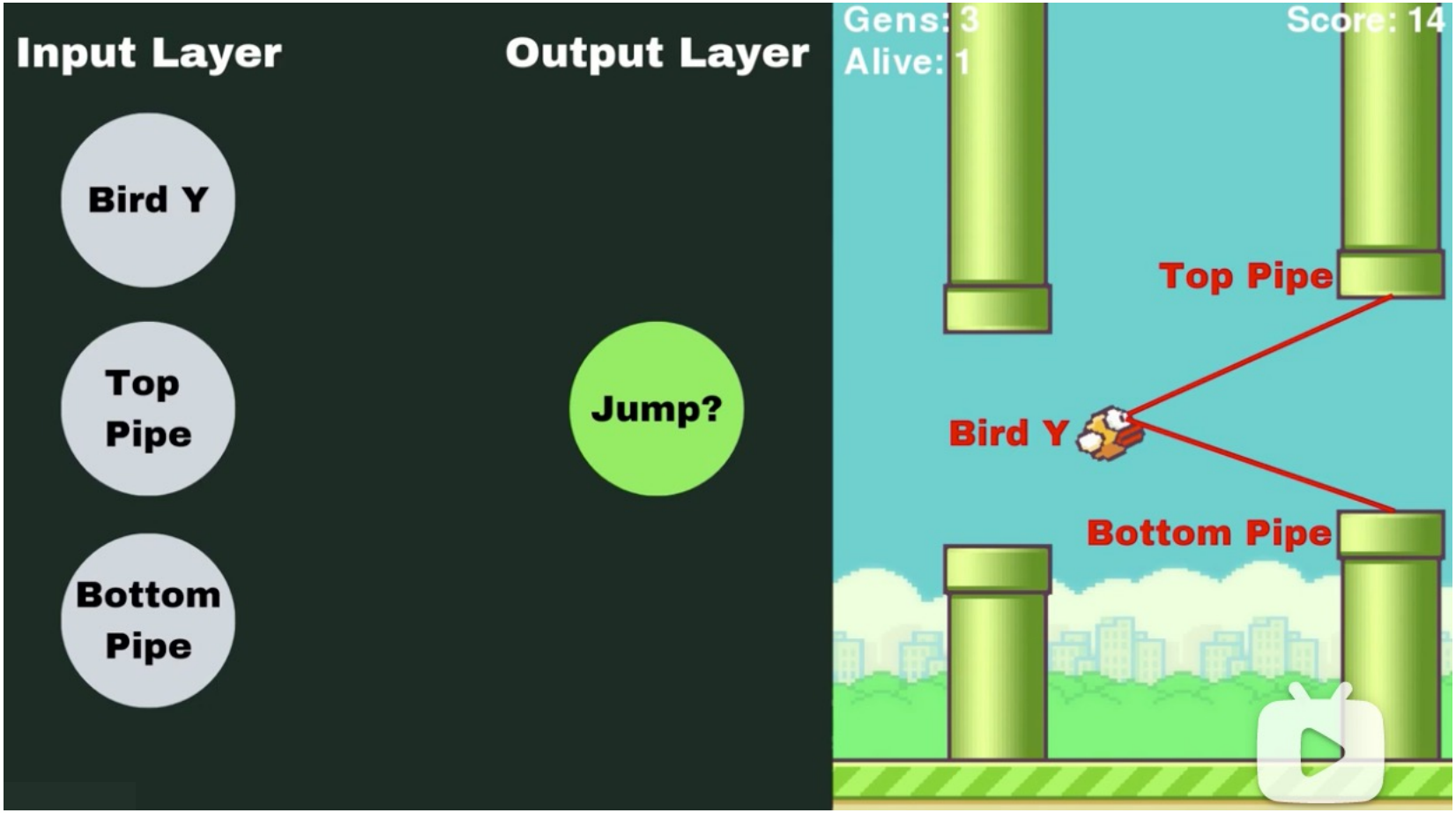
Tech With Tim -- AI Teaches Itself to Play Flappy Bird - Using NEAT Python!

https://www.bilibili.com/video/BV1tE411E7rF?from=search&seid=4747312751760522082

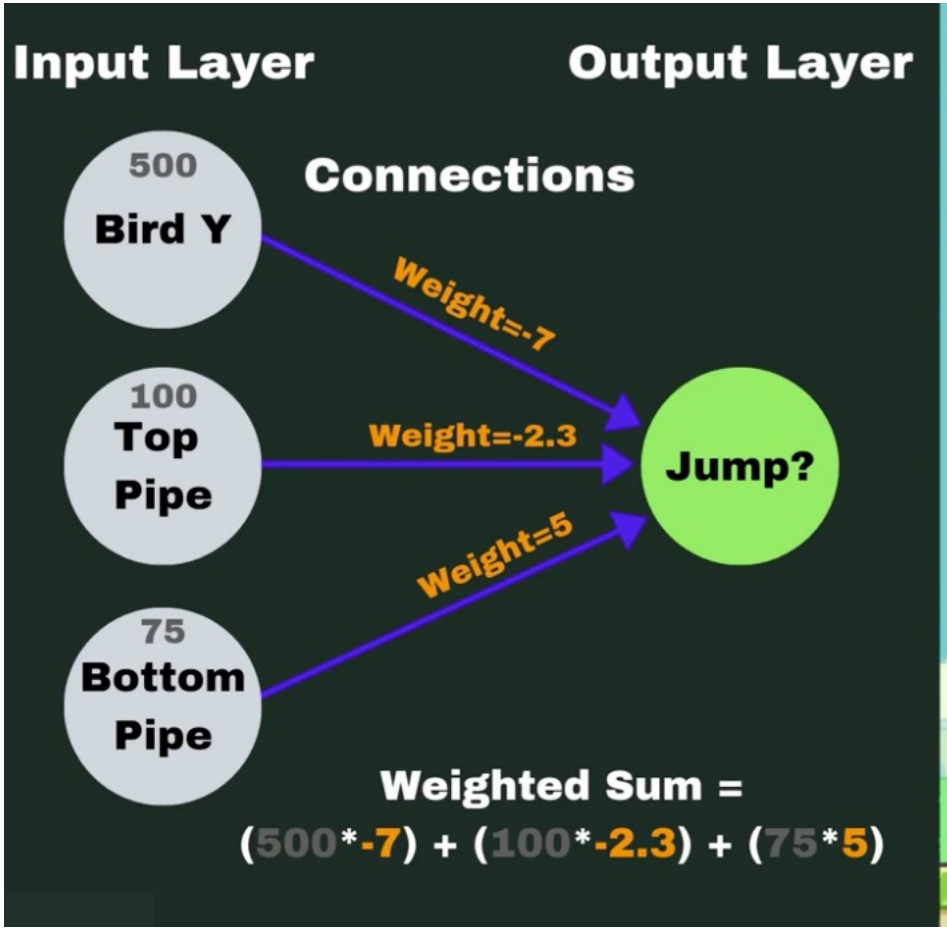
像素鸟游戏

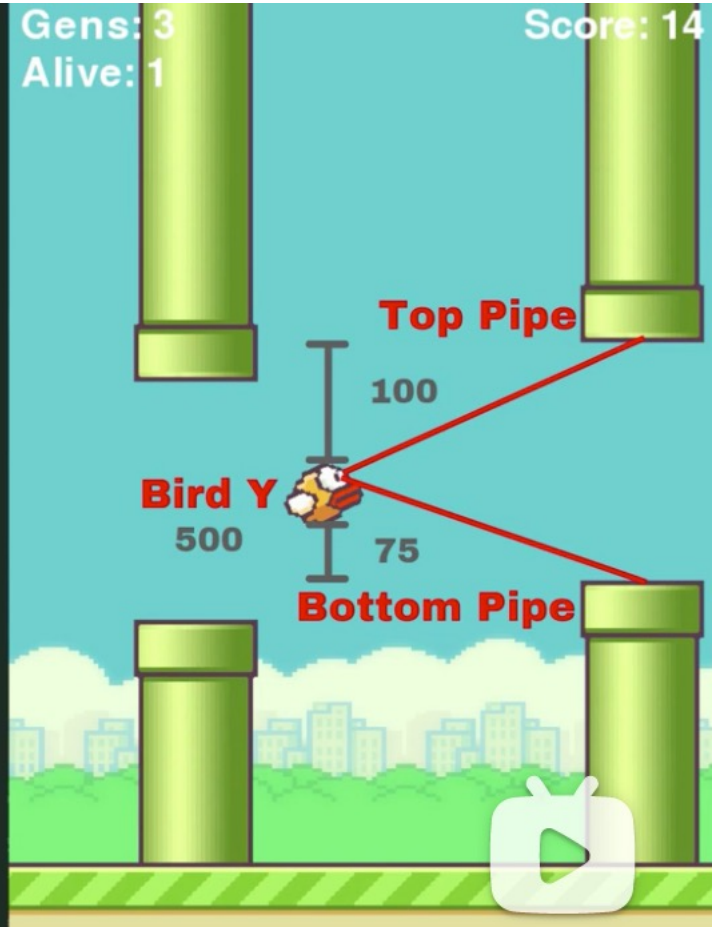


像素鸟游戏

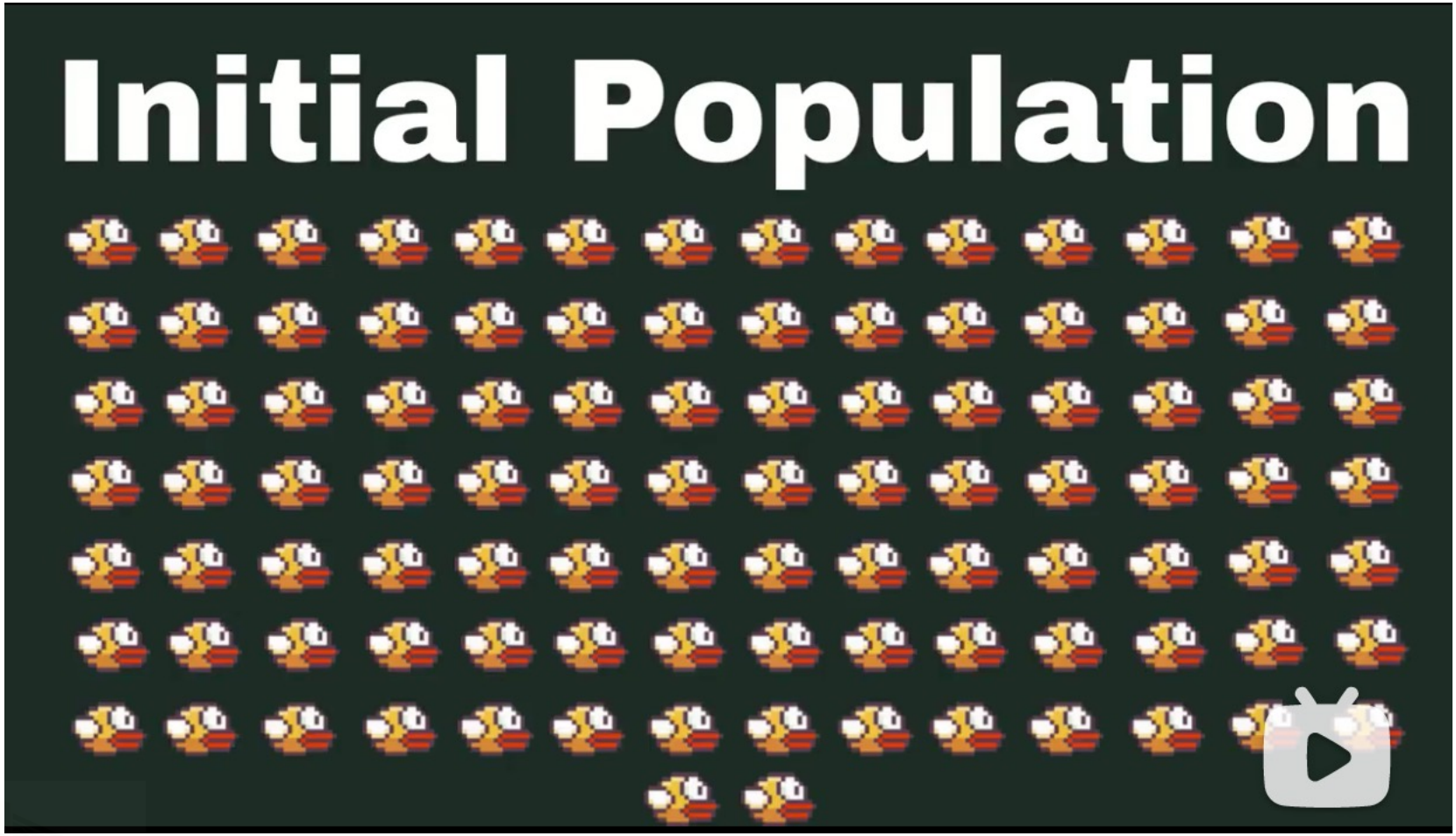

Input Layer
Output Layer
Bird Y
Top Pipe
Bottom Pipe
Jump?
Gens: 3
Alive: 1
Score: 14
Top Pipe
Bird Y
Bottom Pipe

像素鸟游戏

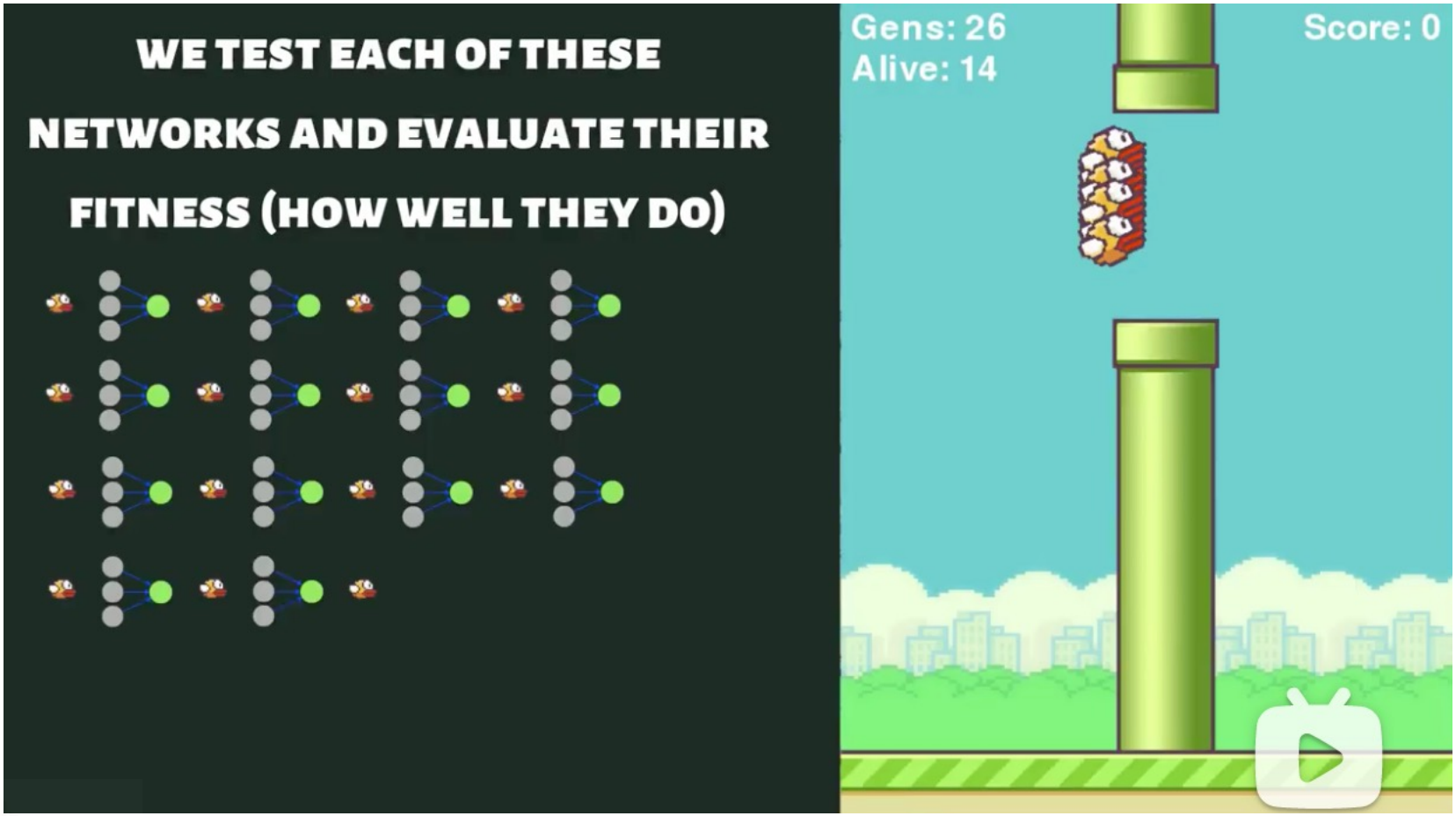

Input Layer
Output Layer
500
Bird Y
Connections
Weight=-7
100
Top
Pipe
Weight=-2.3
Jump?
Weight=5
75
Bottom
Pipe
Weighted Sum =
(500*-7) + (100*-2.3) + (75*5)


Gens: 3
Alive: 1
Score: 14
Top Pipe
100
Bird Y
500
75
Bottom Pipe

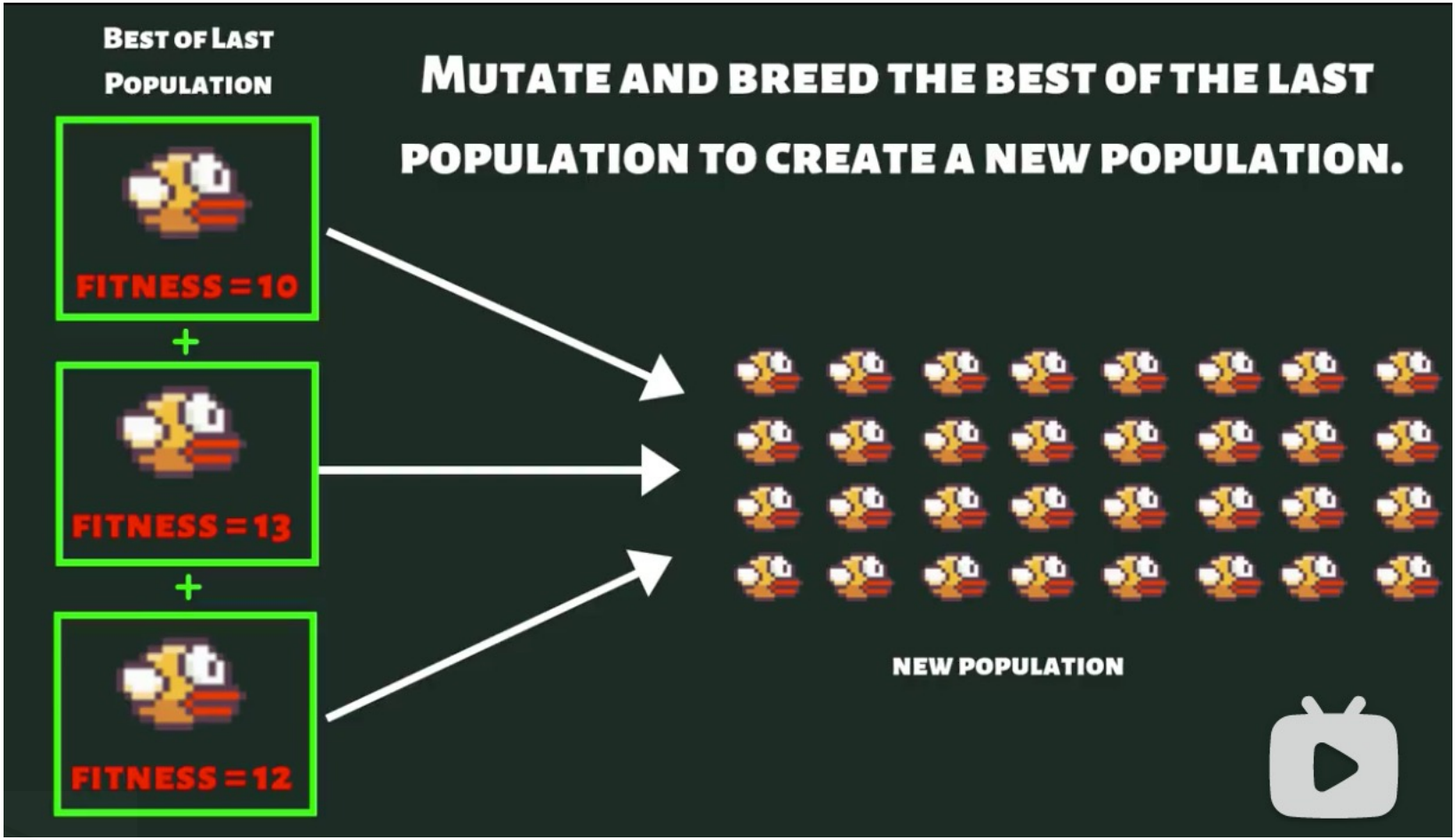
像素鸟游戏



像素鸟游戏


WE TEST EACH OF THESE
NETWORKS AND EVALUATE THEIR
FITNESS (HOW WELL THEY DO)
Gens: 26
Alive: 14
Score: 0

像素鸟游戏


BEST OF LAST POPULATION
MUTATE AND BREED THE BEST OF THE LAST POPULATION TO CREATE A NEW POPULATION.
FITNESS = 10
+
FITNESS = 13
+
FITNESS = 12
NEW POPULATION

像素鸟游戏

Evolving Neural Networks through Augmenting Topologies

3214引用量

Kenneth O. Stanley kstanley@cs.utexas.edu
Department of Computer Sciences, The University of Texas at Austin, Austin, TX 78712, USA

Risto Miikkulainen risto@cs.utexas.edu
Department of Computer Sciences, The University of Texas at Austin, Austin, TX 78712, USA

2002年论文，介绍了如何对神经网络权重（前例中的“weights”）采用遗传算法（基因交叉、基因突变）的方法

内容取自Tech With Tim视频 AI Teaches Itself to Play Flappy Bird - Using NEAT Python!

（实码）遗传算法

Part1 如何从复杂度的角度衡量算法好坏？
Part2 针对一些有特定形式的最优化问题，如何找到最优解？
Part3 针对较复杂的最优化问题（如NP难问题），如何找到较好解？
局部搜索、模拟退火、遗传算法、差分进化算法、蚁群算法、粒子群优化算法、人工蜂群算法
Part4 针对较复杂的多目标最优化问题，如何找到较好解？

遗传算法

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

所有的解需要有基因的表达形式（如01001100...），方便做基因交叉/突变

对于解空间离散的问题（如旅行商问题）可以较容易写出基因表达形式

如何求解解空间连续的问题？

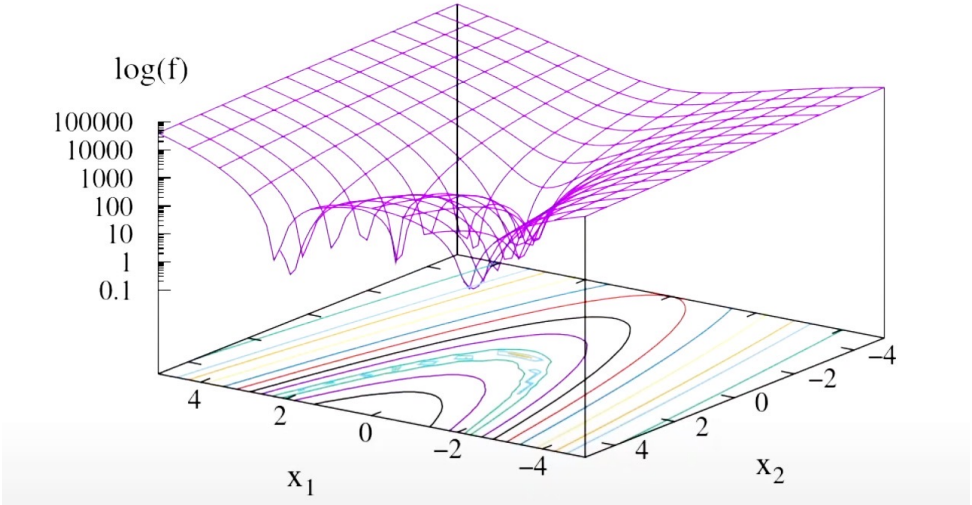
连续解空间

例如，如何用遗传算法求解关于Rosenbrock函数的非凸优化问题

Rosenbrock Function

Minimize $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2,$

bounds $-5 \leq x_1 \leq 5$ and $-5 \leq x_2 \leq 5.$



例子取自Deepak Sharma (IIT Guwahati) Evolutionary Computation for Single and Multi-Objective Optimization

连续解空间

方法一：将连续的解空间离散化

连续解空间

方法一：将连续的解空间离散化

将连续解空间$[x, y]$离散化成由$\{a_1, \ldots, a_L\} \in \{0,1\}^L$表示的解空间

- 定义函数$\Gamma(a_1, \ldots, a_L)$计算离散解$(a_1, \ldots, a_L)$在连续解空间对应的值

$$\Gamma(a_1,\ldots,a_L) = x + \frac{y-x}{2^L-1} \cdot \left(\sum_{j=0}^{L-1} a_{L-j} \cdot 2^j\right) \in [x, y]$$

若是多维变量则每个维度都如此处理

连续解空间

方法一：将连续的解空间离散化

将连续解空间$[x,y]$离散化成由$\{a_1,\dots,a_L\}\in\{0,1\}^L$表示的解空间

- 定义函数$\Gamma(a_1,\dots,a_L)$计算离散解$(a_1,\dots,a_L)$在连续解空间对应的值

$$\Gamma(a_1,\dots,a_L)=x+\frac{y-x}{2^L-1}\cdot\left(\sum_{j=0}^{L-1}a_{L-j}\cdot 2^j\right)\in[x,y]$$

若是多维变量则每个维度都如此处理

- 共 2^L种取值，即L决定了表达精度
- L越大，精度越高，但是基因序列越长、算法收敛时间越长

连续解空间

例如，对连续区间[0,5]做离散化处理。如果L=5

实数 0　　对应基因表达为 00000
实数 5/31　对应基因表达为 00001
实数 10/31 对应基因表达为 00010
实数 15/31 对应基因表达为 00011
…
实数 5　　对应基因表达为 11111

连续解空间

例如，对连续区间[0,5]做离散化处理。如果L=5

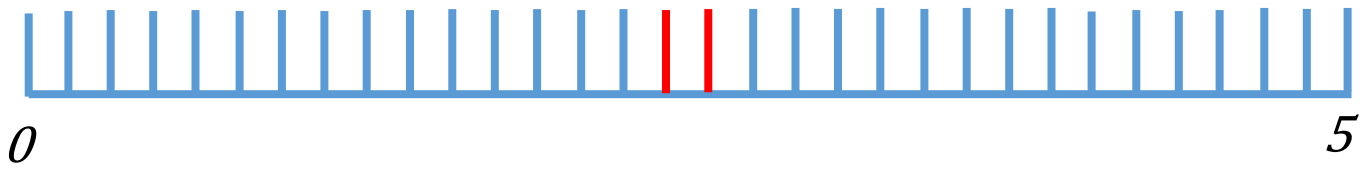
实数 0　　对应基因表达为 00000
实数 5/31　对应基因表达为 00001
实数 10/31 对应基因表达为 00010
实数 15/31 对应基因表达为 00011
…
实数 5　　对应基因表达为 11111

该种方法有什么缺憾？

精度难以保证：长度L的序列只能达到$(5-0)/(2^L-1)$的精度

连续解空间

例如，对连续区间[0,5]做离散化处理。如果L=5



实数 0　　对应基因表达为 00000
实数 5/31　对应基因表达为 00001
实数 10/31 对应基因表达为 00010
实数 15/31 对应基因表达为 00011
…
实数 5　　对应基因表达为 11111

该种方法有什么缺憾？

汉明悬崖(Hamming Cliff)
如相邻的两个解 75/31 与 80/31，
序列差别很大（01111与10000）

连续解空间

方法一：将连续的解空间离散化

缺憾：精度难以保证；汉明悬崖

方法二：直接在连续空间运用遗传算法

称为“用实数编码的遗传算法” (real-coded genetic algorithm)

连续解空间

方法一：将连续的解空间离散化

缺憾：精度难以保证；汉明悬崖

方法二：直接在连续空间运用遗传算法

称为“用实数编码的遗传算法” (real-coded genetic algorithm)

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

哪些步骤会受到影响？

连续解空间

遗传算法

0 初始化当前种群$P(t)$ ⟸ 直接在连续空间随机取值
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

连续解空间

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$ ⟸ 不受影响
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

连续解空间

遗传算法

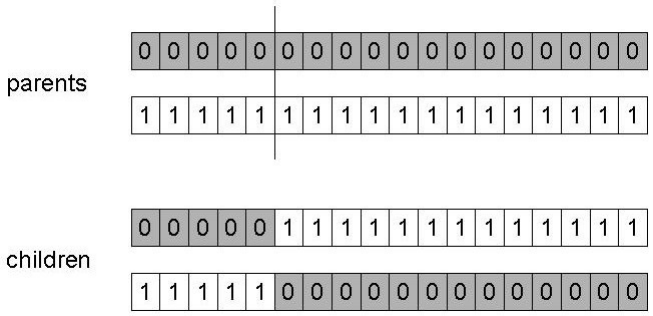
0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$ ⟸ 不受影响
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

连续解空间

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

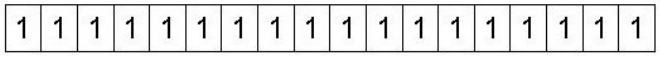
需要重新定义



连续解空间

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变 ⟸ 需要重新定义
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法



连续解空间

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$ ⟸ 不受影响
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

连续解空间

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

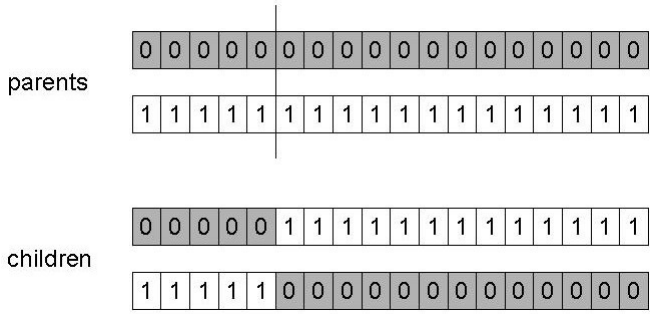
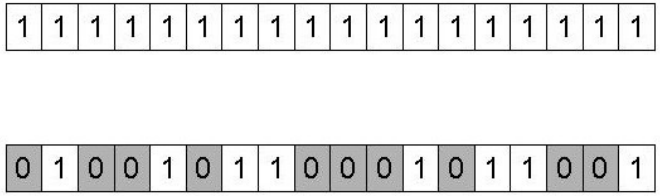
不受影响

连续解空间

遗传算法

0 初始化当前种群$P(t)$
1 评估当前种群$P(t)$
2 for t=1 to ∞:
3 　根据当前种群$P(t)$，确定父代种群$P_P(t)$
4 　根据父代种群$P_P(t)$，通过基因交叉生成子代种群$P_C(t)$
5 　对子代种群$P_C(t)$进行基因突变
6 　评估子代种群$P_C(t)$
7 　根据当前种群$P(t)$和子代种群$P_C(t)$，筛选出新的当前种群$P(t+1)$
8 　检查循环终止条件，若满足则结束算法

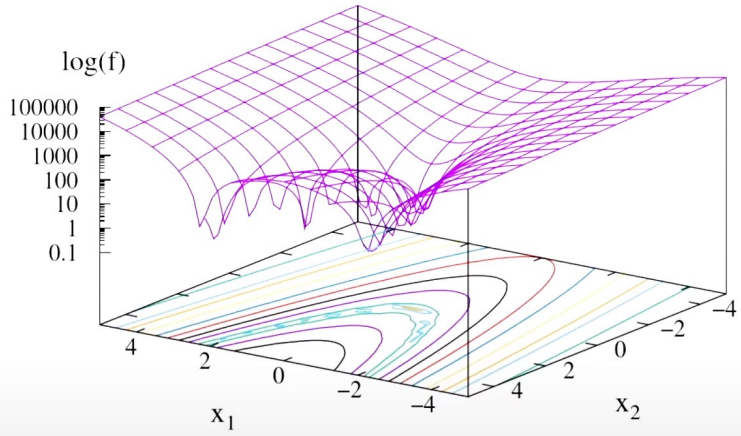
实码遗传算法

	二进制遗传算法 （binary-coded genetic algorithm）	实码遗传算法 （real-coded genetic algorithm）
基因交叉		仿二进制基因交叉 SBX (simulated binary crossover)
基因突变		1. 均匀突变 2. 非均匀突变（如正态分布突变）

实码遗传算法

Rosenbrock Function

Minimize $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$,
bounds $-5 \leq x_1 \leq 5$ and $-5 \leq x_2 \leq 5$.



- Optimum solution is $x^* = (1, 1)^T$ and $f(x^*) = 0$

实码遗传算法

随机生成初始种群（取N=8）

Rosenbrock Function

Minimize $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2,$
bounds $-5 \leq x_1 \leq 5$ and $-5 \leq x_2 \leq 5.$

Initial population

Index	x_1	x_2
1	2.212	3.009
2	−2.289	−2.396
3	−2.393	−4.790
4	−0.639	1.692
5	−3.168	0.706
6	0.215	−2.350
7	−0.742	1.934
8	−4.563	4.791

内容取自Deepak Sharma (IIT Guwahati) Evolutionary Computation for Single and Multi-Objective Optimization

实码遗传算法

评价初始解质量

Rosenbrock Function

Minimize $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2,$
bounds $-5 \leq x_1 \leq 5$ and $-5 \leq x_2 \leq 5.$

Initial population

Index	x_1	x_2	$f(x_1, x_2)$
1	2.212	3.009	357.154
2	−2.289	−2.396	5843.569
3	−2.393	−4.790	11066.800
4	−0.639	1.692	167.414
5	−3.168	0.706	8718.166
6	0.215	−2.350	574.796
7	−0.742	1.934	194.618
8	−4.563	4.791	25731.235

实码遗传算法

选择允许配对繁殖的解

Initial population

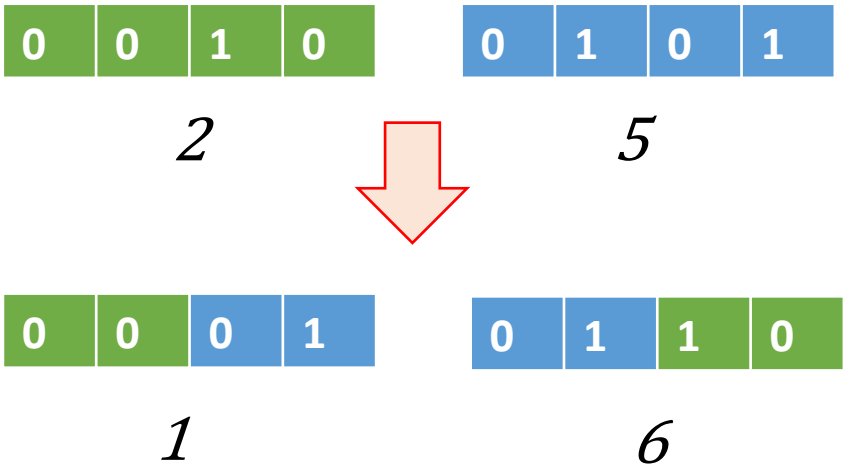
Index	x_1	x_2	$f(x_1, x_2)$
1	2.212	3.009	357.154
2	−2.289	−2.396	5843.569
3	−2.393	−4.790	11066.800
4	−0.639	1.692	167.414
5	−3.168	0.706	8718.166
6	0.215	−2.350	574.796
7	−0.742	1.934	194.618
8	−4.563	4.791	25731.235

Mating Pool

Old index	New index	x_1	x_2	$f(x_1, x_2)$
7	1	−0.742	1.934	194.618
4	2	−0.639	1.692	167.414
3	3	−2.393	−4.790	11066.800
1	4	2.212	3.009	357.154
1	5	2.212	3.009	357.154
2	6	−2.289	−2.396	5843.569
7	7	−0.742	1.934	194.618
4	8	−0.639	1.692	167.414

仿二进制基因交叉SBX

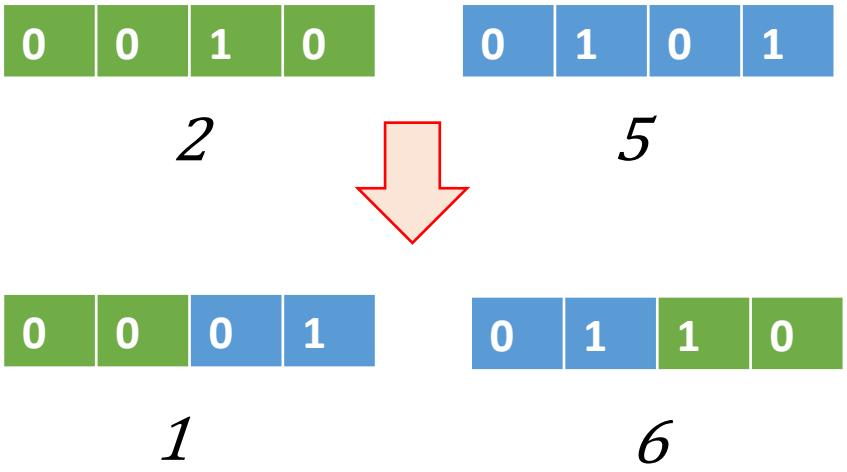
二进制编码中基因交叉的本质是什么？



平均值不变

仿二进制基因交叉SBX

二进制编码中基因交叉的本质是什么？



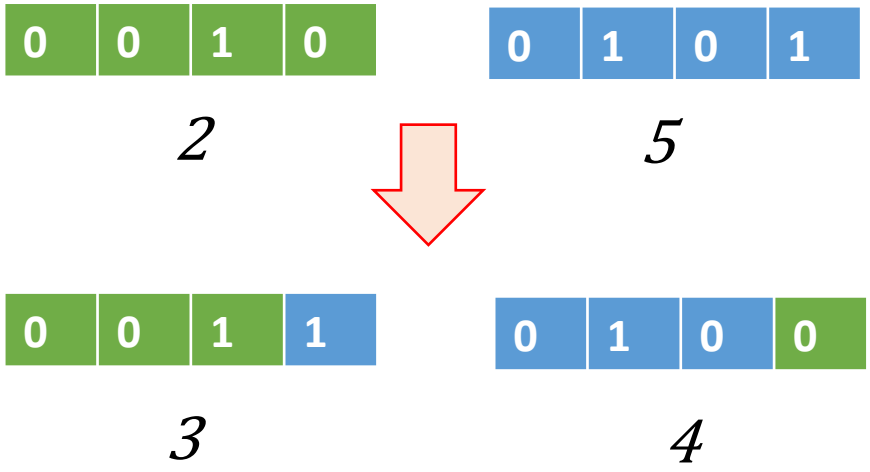
平均值不变

p: parents
o: offspring

子代间距离拉远

仿二进制基因交叉SBX

二进制编码中基因交叉的本质是什么？

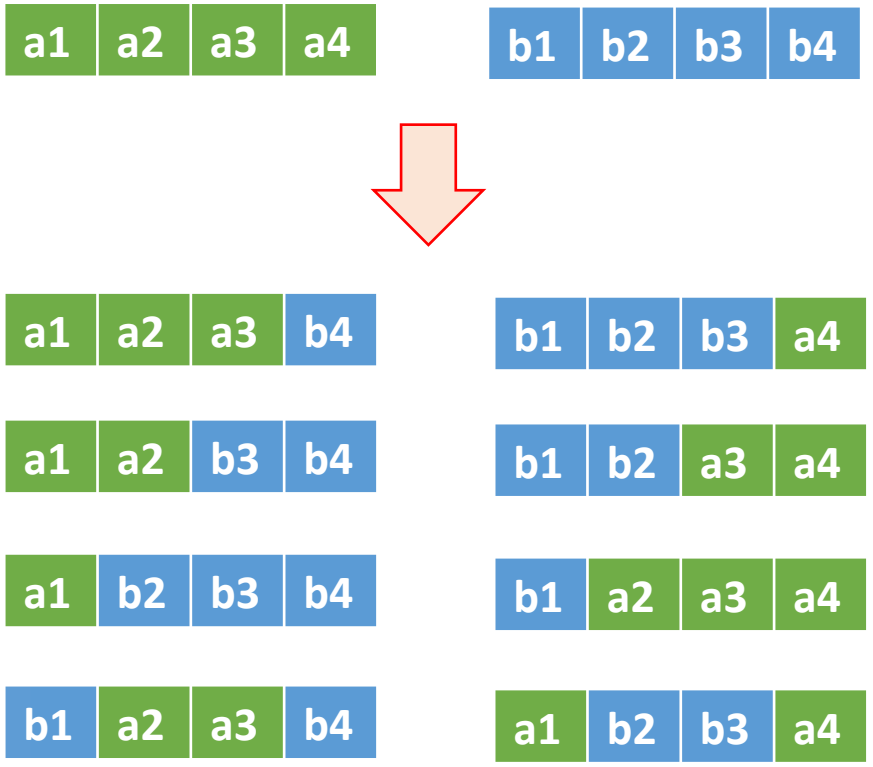


平均值不变

子代间距离拉近

仿二进制基因交叉SBX

二进制编码中基因交叉的本质是什么？

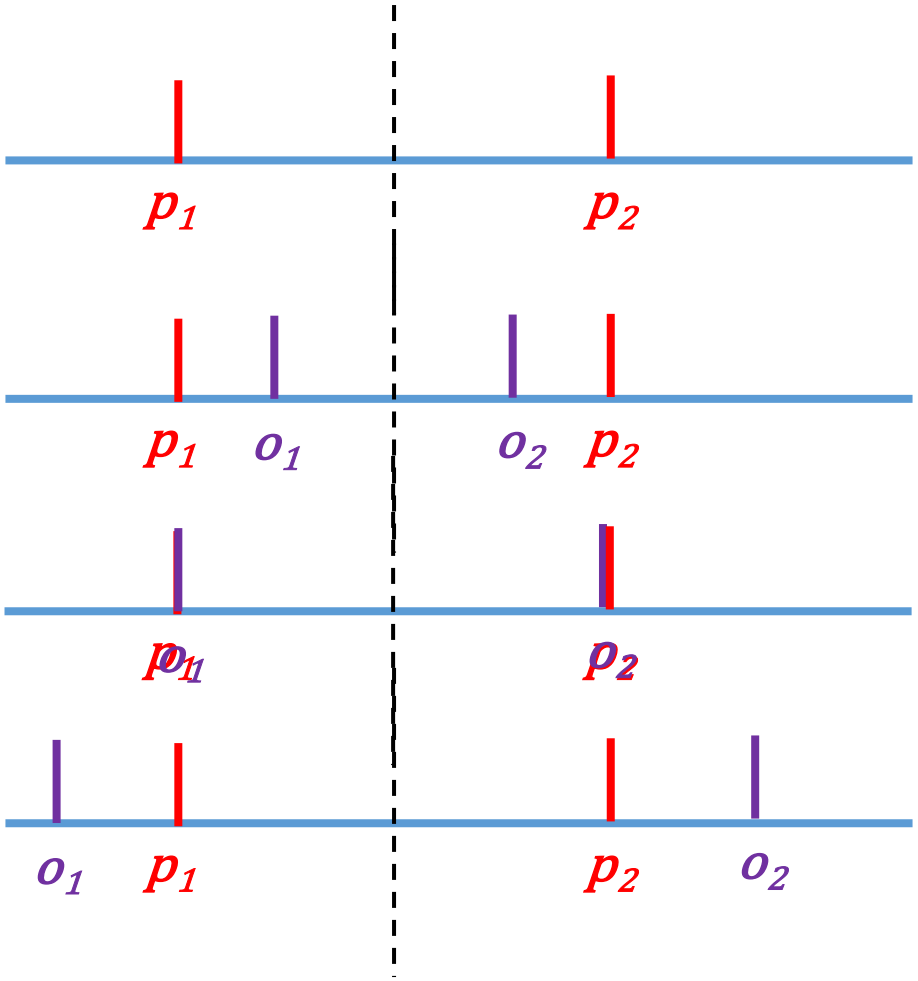


平均值不变

(能否证明？)

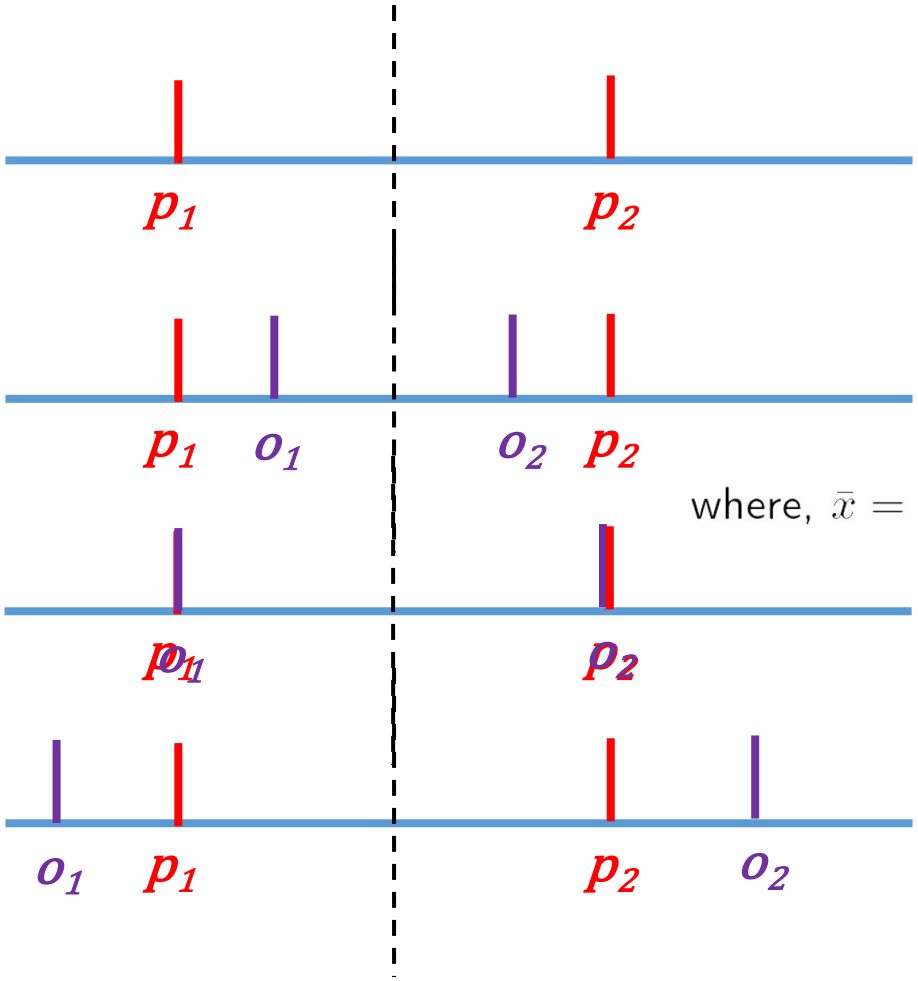
仿二进制基因交叉SBX

二进制编码中基因交叉的本质是什么？



仿二进制基因交叉SBX

二进制编码中基因交叉的本质是什么？



$$o_1 = \bar{x} - \frac{1}{2}\beta(p_2 - p_1)$$
$$o_2 = \bar{x} + \frac{1}{2}\beta(p_2 - p_1)$$

where, $\bar{x} = \frac{1}{2}(p_1 + p_2)$ and $p_2 > p_1$.

子代之间更相似

β<1

子代不变

β=1

子代之间更不同

β>1

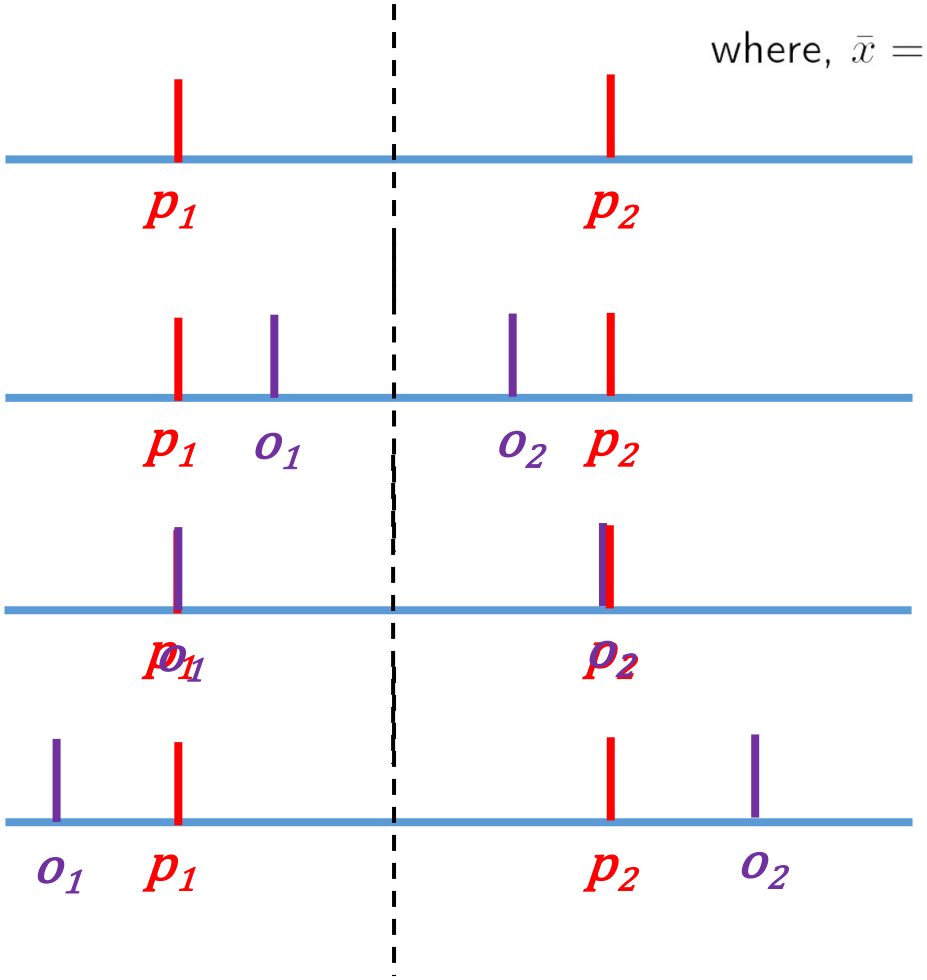
仿二进制基因交叉SBX

$$o_1 = \bar{x} - \frac{1}{2}\beta(p_2 - p_1)$$
$$o_2 = \bar{x} + \frac{1}{2}\beta(p_2 - p_1)$$

在实数域，可以直接用公式计算繁衍的子代（每个维度分别计算）

β怎么取值？

where, $\bar{x} = \frac{1}{2}(p_1 + p_2)$ and $p_2 > p_1$.



子代之间更相似

β<1

子代不变

β=1

子代之间更不同

β>1

仿二进制基因交叉SBX

$$o_1 = \bar{x} - \frac{1}{2}\beta(p_2 - p_1)$$
$$o_2 = \bar{x} + \frac{1}{2}\beta(p_2 - p_1)$$

where, $\bar{x} = \frac{1}{2}(p_1 + p_2)$ and $p_2 > p_1$.

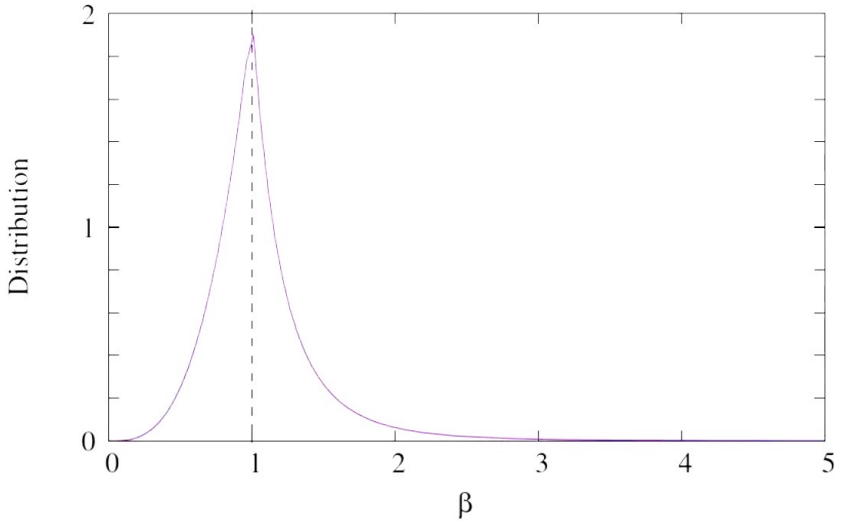
β怎么取值？

- Probability distribution function:

$$p(\beta_i) = \begin{cases} 0.5(\eta_c + 1)\beta_i^{\eta_c}, & \text{if } \beta_i \leq 1 \\ 0.5(\eta_c + 1)\frac{1}{\beta_i^{\eta_c + 2}}, & \text{otherwise.} \end{cases}$$

- η_c is the SBX crossover operator distribution factor that is set by us.

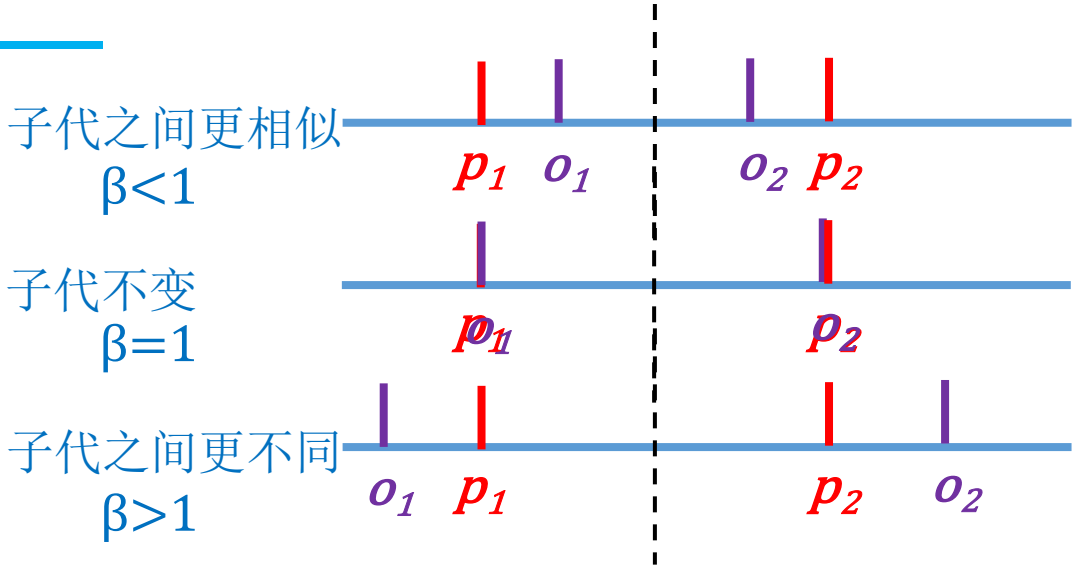
η_c影响分布形状

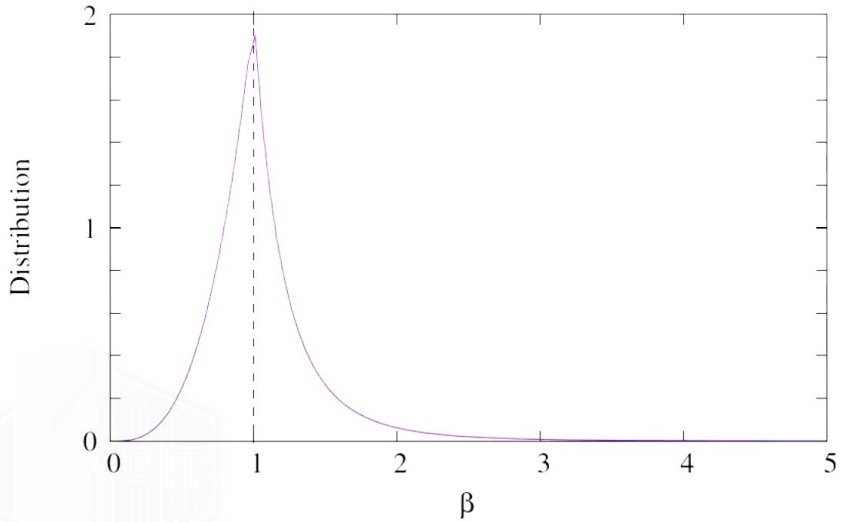


内容取自Deepak Sharma (IIT Guwahati) Evolutionary Computation for Single and Multi-Objective Optimization

仿二进制基因交叉SBX

β怎么取值？





β越接近1，子代与父母越相接近

实码遗传算法

选择允许配对繁殖的解

Initial population

Index	x_1	x_2	$f(x_1, x_2)$
1	2.212	3.009	357.154
2	−2.289	−2.396	5843.569
3	−2.393	−4.790	11066.800
4	−0.639	1.692	167.414
5	−3.168	0.706	8718.166
6	0.215	−2.350	574.796
7	−0.742	1.934	194.618
8	−4.563	4.791	25731.235

Mating Pool

Old index	New index	x_1	x_2	$f(x_1, x_2)$
7	1	−0.742	1.934	194.618
4	2	−0.639	1.692	167.414
3	3	−2.393	−4.790	11066.800
1	4	2.212	3.009	357.154
1	5	2.212	3.009	357.154
2	6	−2.289	−2.396	5843.569
7	7	−0.742	1.934	194.618
4	8	−0.639	1.692	167.414

实码遗传算法

选择允许配对繁殖的解

对第i个维度：

$$
\begin{aligned}
x_i^{(1,t+1)} &= 0.5\left[(x_i^{(1,t)} + x_i^{(2,t)}) - \beta_i(x_i^{(2,t)} - x_i^{(1,t)})\right] \\
x_i^{(2,t+1)} &= 0.5\left[(x_i^{(1,t)} + x_i^{(2,t)}) + \beta_i(x_i^{(2,t)} - x_i^{(1,t)})\right]
\end{aligned}
$$

注意对每个维度都分别取一个新的β

Initial population

Index	x_1	x_2	$f(x_1, x_2)$
1	2.212	3.009	357.154
2	−2.289	−2.396	5843.569
3	−2.393	−4.790	11066.800
4	−0.639	1.692	167.414
5	−3.168	0.706	8718.166
6	0.215	−2.350	574.796
7	−0.742	1.934	194.618
8	−4.563	4.791	25731.235

Mating Pool

Old index	New index	x_1	x_2	$f(x_1, x_2)$
7	1	−0.742	1.934	194.618
4	2	−0.639	1.692	167.414
3	3	−2.393	−4.790	11066.800
1	4	2.212	3.009	357.154
1	5	2.212	3.009	357.154
2	6	−2.289	−2.396	5843.569
7	7	−0.742	1.934	194.618
4	8	−0.639	1.692	167.414

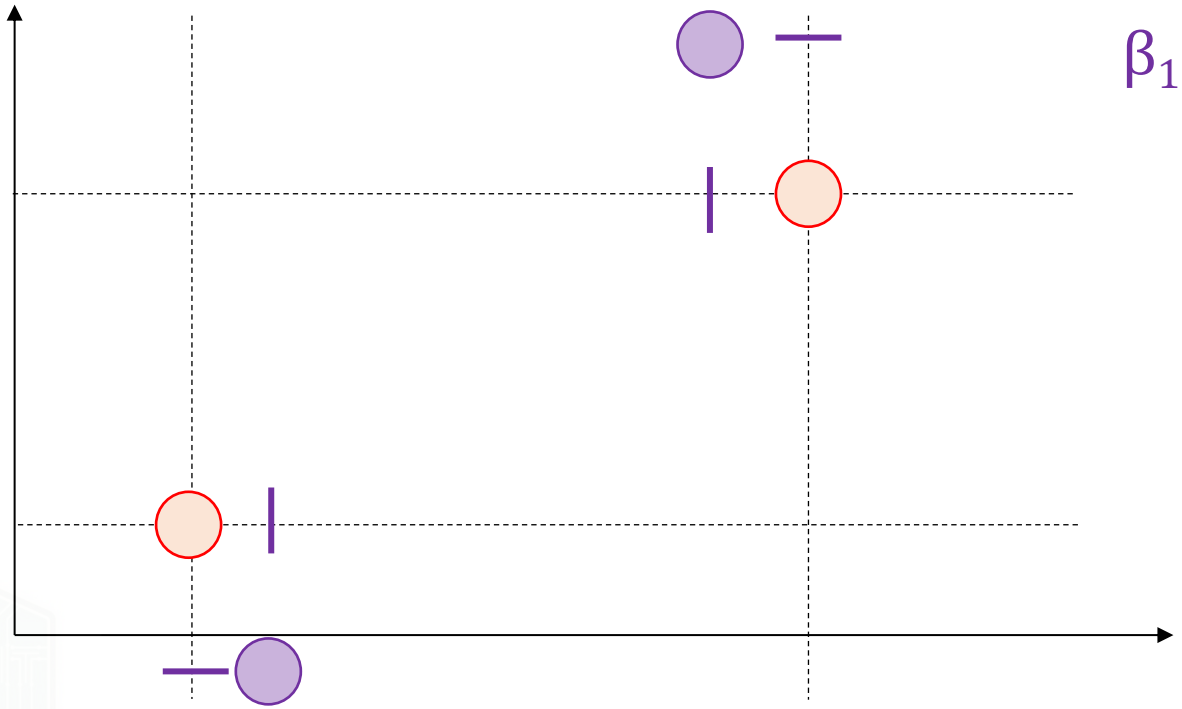
实码遗传算法

选择允许配对繁殖的解

对第*i*个维度：

$$x_i^{(1,t+1)} = 0.5\left[(x_i^{(1,t)} + x_i^{(2,t)}) - \beta_i(x_i^{(2,t)} - x_i^{(1,t)})\right]$$
$$x_i^{(2,t+1)} = 0.5\left[(x_i^{(1,t)} + x_i^{(2,t)}) + \beta_i(x_i^{(2,t)} - x_i^{(1,t)})\right]$$

注意对每个维度都分别取一个新的β



$\beta_1<1, \beta_2>1$

实码遗传算法

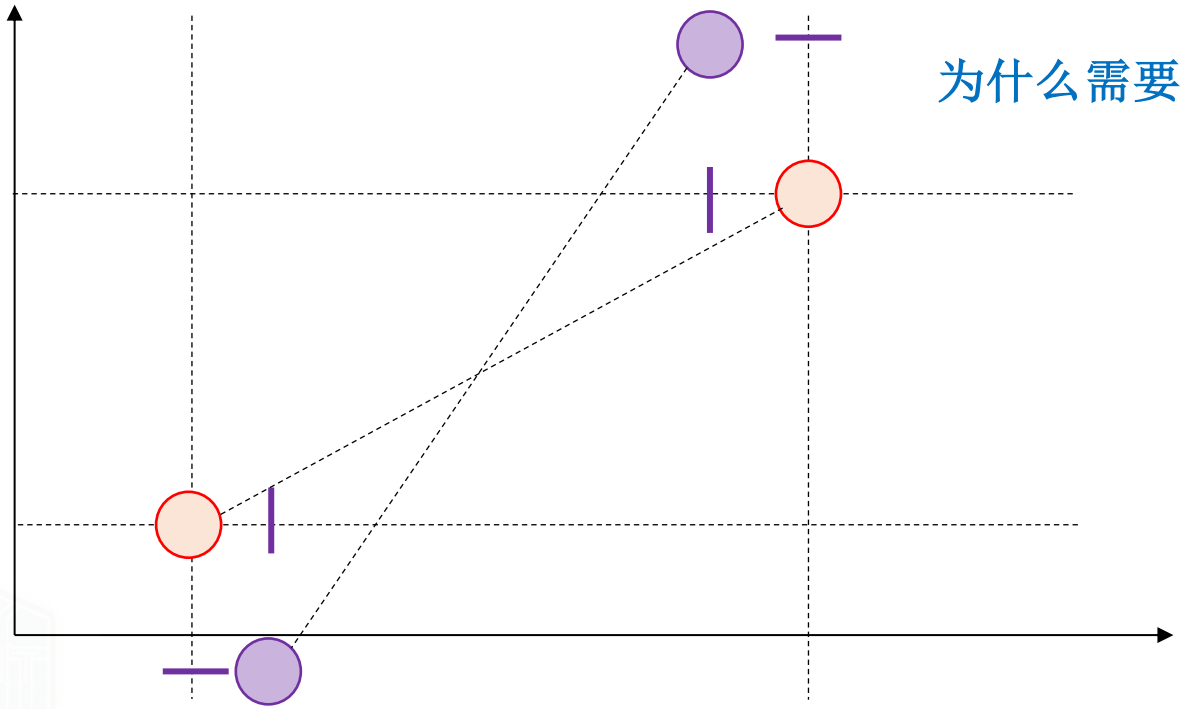
选择允许配对繁殖的解

对第*i*个维度：

$$x_i^{(1,t+1)} = 0.5\left[(x_i^{(1,t)} + x_i^{(2,t)}) - \beta_i(x_i^{(2,t)} - x_i^{(1,t)})\right]$$
$$x_i^{(2,t+1)} = 0.5\left[(x_i^{(1,t)} + x_i^{(2,t)}) + \beta_i(x_i^{(2,t)} - x_i^{(1,t)})\right]$$

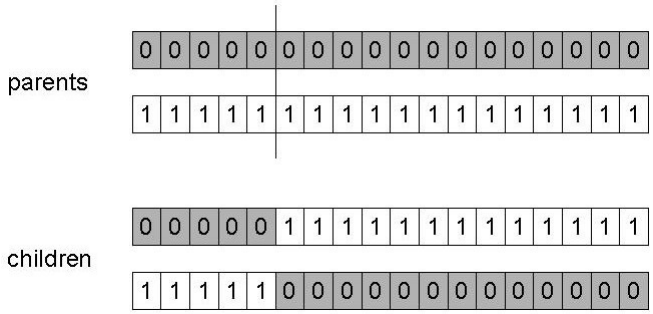
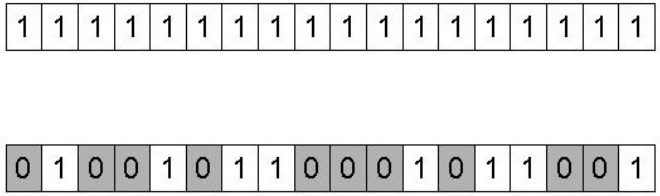
注意对每个维度都分别取一个新的β

交叉前后，均值不变



为什么需要均值不变？

实码遗传算法

	二进制遗传算法 （binary-coded genetic algorithm）	实码遗传算法 （real-coded genetic algorithm）
基因交叉		仿二进制基因交叉 SBX (simulated binary crossover)
基因突变		1. 均匀突变 2. 非均匀突变（如正态分布突变）

实码遗传算法

均匀突变：随机以均匀分布从定义域中取值

非均匀突变：正态分布突变、多项式突变等

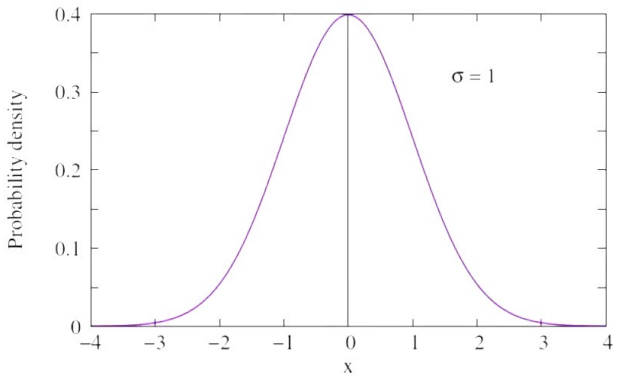
实码遗传算法

正态分布突变：对每个基因都加上服从 $\mathcal{N}(0,\sigma)$ 高斯分布的随机变量，然后截断到定义域。高斯分布的标准差σ控制突变的程度（即$\frac{2}{3}$的突变将在$\pm\sigma$范围内）

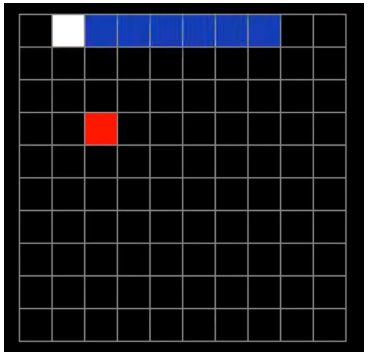
- Simple and popular method is to use a zero-mean Gaussian probability distribution:

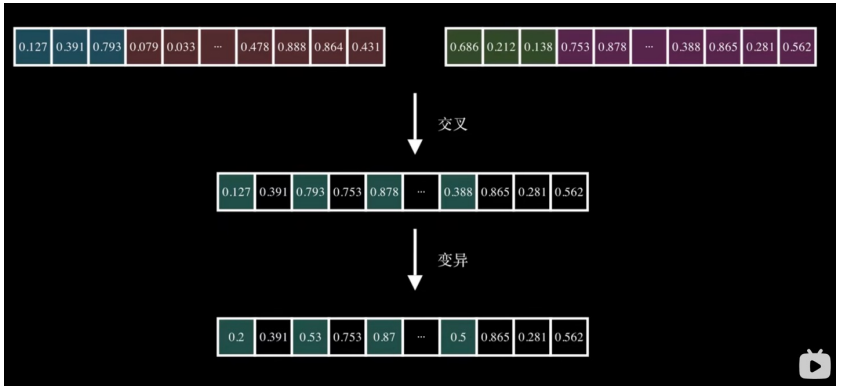
$$y_i^{(1,t)} = x_i^{(1,t)} + N(0,\sigma_i)$$

- σ_i is a fixed user defined parameter. It must be set correctly in a problem.
- Special care must be taken to respect boundary limits on decision variables.



实码遗传算法



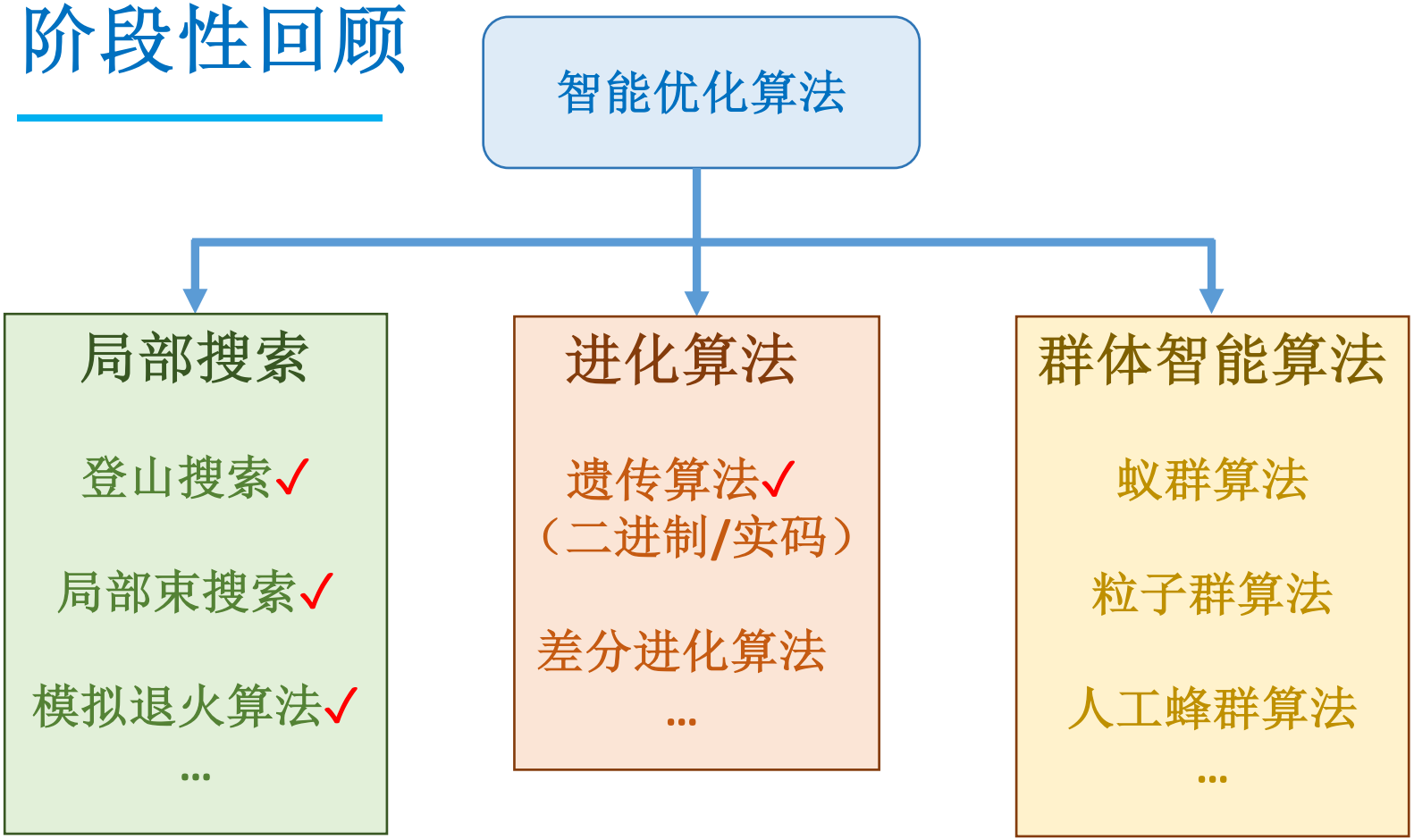


贪吃蛇【遗传算法+神经网络】

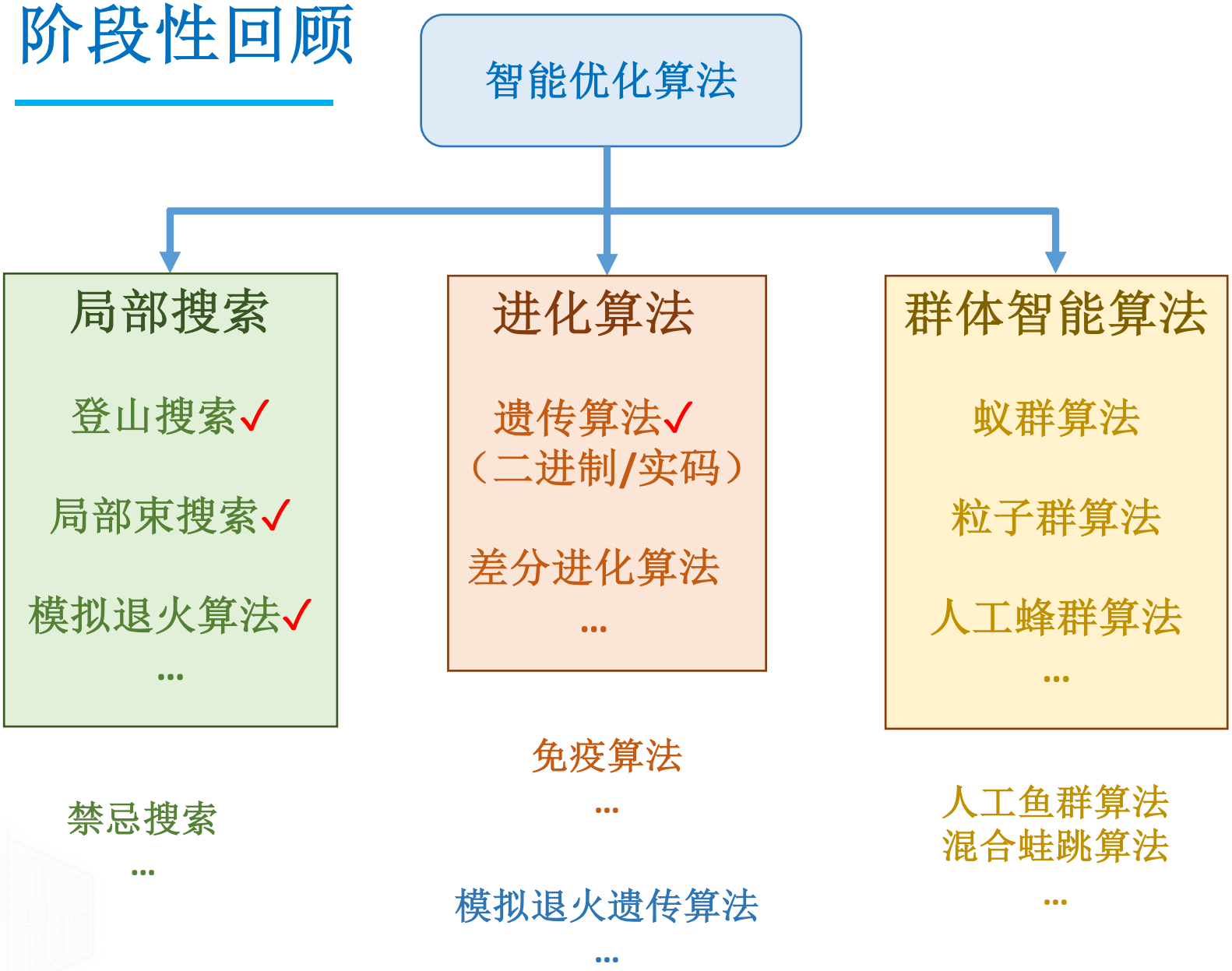
https://www.bilibili.com/video/BV14T4y1v7Cx/?spm_id_from=333.337.search-card.all.click

在该例子中，基因交叉的具体实现有所简化

阶段性回顾


智能优化算法
局部搜索
登山搜索✓
局部束搜索✓
模拟退火算法✓
...
进化算法
遗传算法✓
（二进制/实码）
差分进化算法
...
群体智能算法
蚁群算法
粒子群算法
人工蜂群算法
...

阶段性回顾

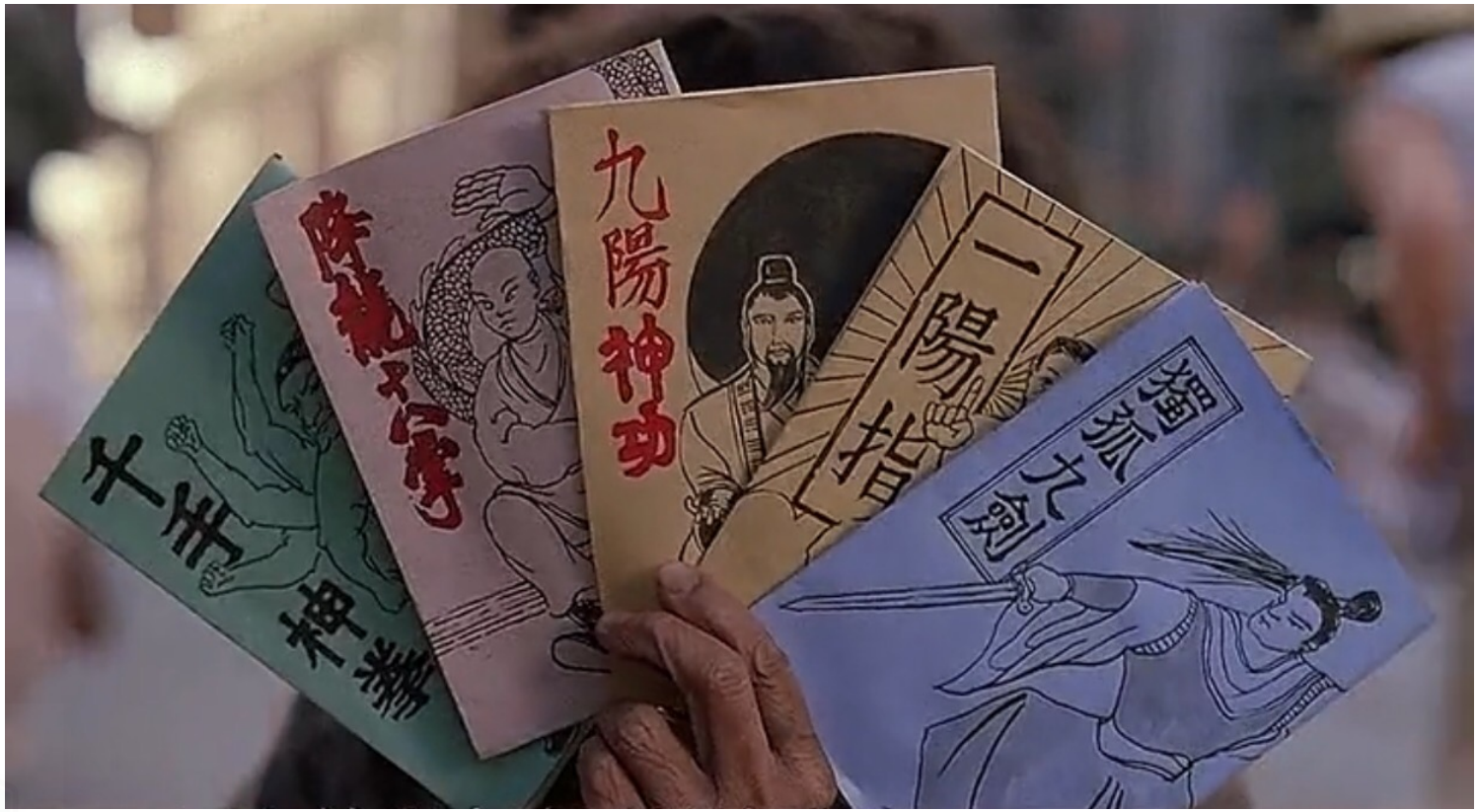

智能优化算法
局部搜索
登山搜索✓
局部束搜索✓
模拟退火算法✓
…
进化算法
遗传算法✓
（二进制/实码）
差分进化算法
…
群体智能算法
蚁群算法
粒子群算法
人工蜂群算法
…
禁忌搜索
…
免疫算法
…
模拟退火遗传算法
…
人工鱼群算法
混合蛙跳算法
…

阶段性回顾

刚刚学几个算法的时候...

阶段性回顾

算法越学越多的时候...



阶段性回顾

不用完全具体记住每一个算法：

- 课堂涵盖的算法多是引用上万的经典算法，算法的设计思想有代表性

  —— 如登山搜索、模拟退火、遗传算法
  —— 如何解决登山搜索陷入局部最优、将遗传算法用于连续决策变量

- 生产实际中被验证有效，可以作为备用工具

- 课程对应学科方向的思维/研究范式

本讲小结

- 模拟退火算法
- 进化算法之遗传算法（二进制/实码）

计算机学院
余皓然

主要参考资料

为汉科技《回火、淬火、正火、退火》视频

Delahaye, et al <Simulated annealing: From basics to applications>

K. A. Dowsland, J. M. Thompson <Simulated Annealing - Handbook of Natural Computing>

Scott Hauck <UW EE 541: Automated Layout of Integrated Circuits>

Thomas Stutzle <Heuristic Optimization>

Frédéric MARQUER <Reproduce image with genetic algorithm> Video

B站“笔记鲨”<10分钟算法: 遗传算法>

A.E. Eiben and J.E. Smith <Introduction to Evolutionary Computing> Slides

Tech With Tim <AI Teaches Itself to Play Flappy Bird - Using NEAT Python!> Video

Deepak Sharma (IIT Guwahati) <Evolutionary Computation for Single and Multi-Objective Optimization> Slides

B站“Int_m1an”<贪吃蛇【遗传算法+神经网络】>

谢谢！

计算机学院
余皓然
2024/4/1