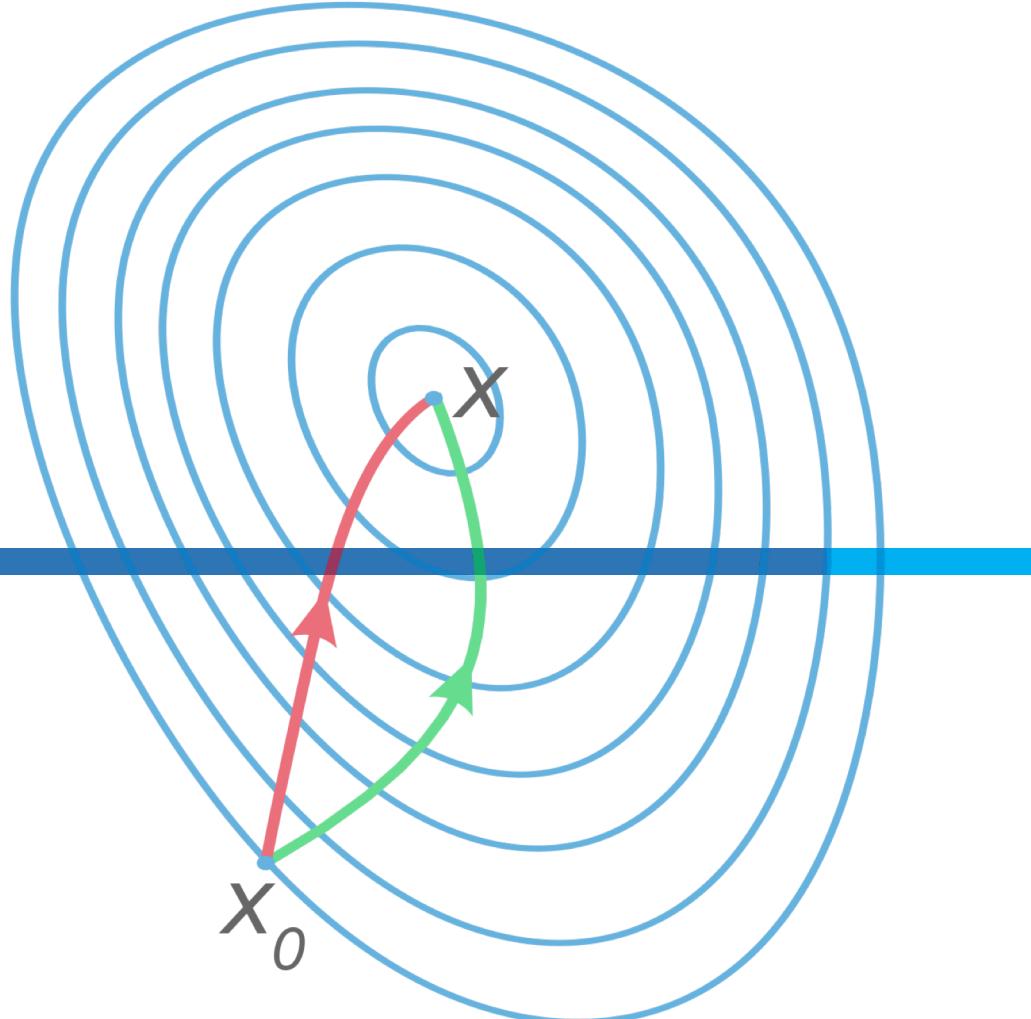


最优化方法

第六周

计算机学院
余皓然

2023/3/30



进化算法之遗传算法

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

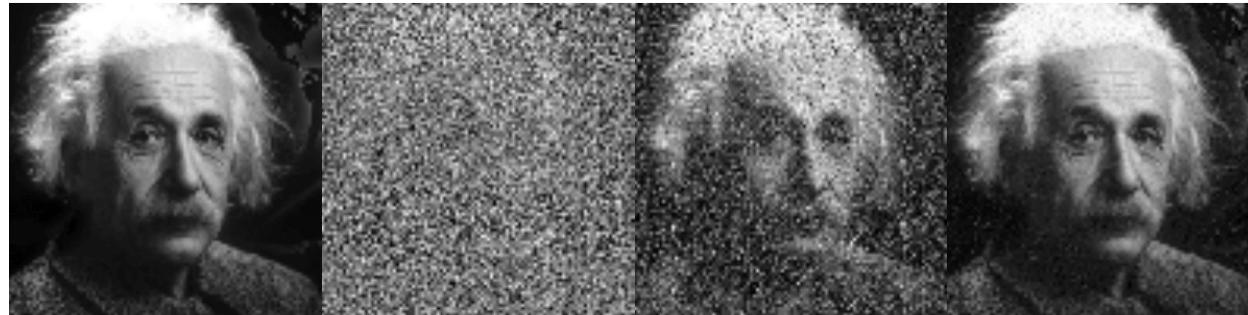
Albert Einstein

Original

2K generations

10K generations

50K generations



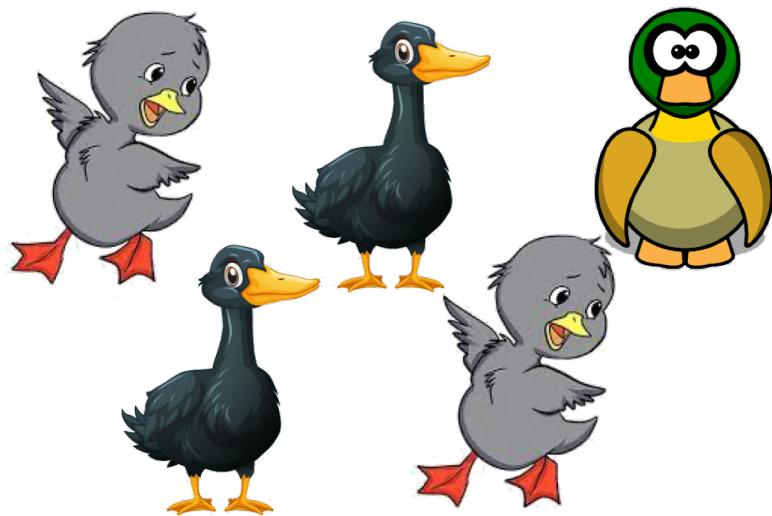
遗传算法酷炫特效——图片再生成 (by Frédéric MARQUER)

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

遗传算法

从前有个鸭子村...

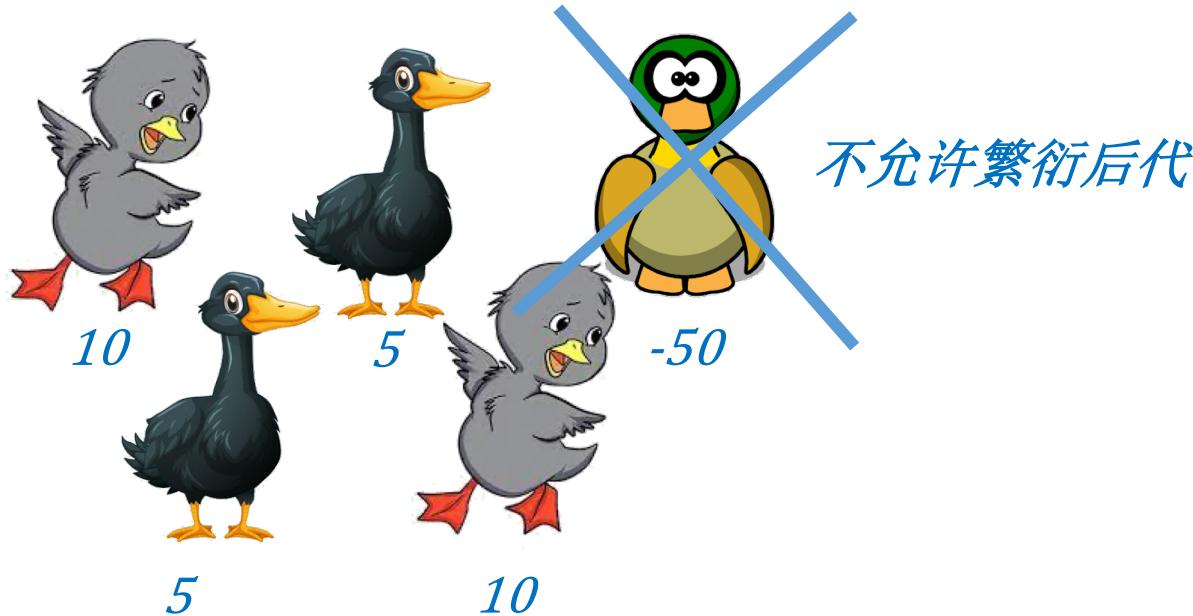
第一年



遗传算法

第一年

评测分数



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

遗传算法



10

1	1	1	1
---	---	---	---



5

0	0	1	0
---	---	---	---

基因交叉

0	0	1	1
---	---	---	---

基因突变

0	1	1	1
---	---	---	---

1	1	1	0
---	---	---	---

1	1	0	0
---	---	---	---

遗传算法



10

1	1	1	1
---	---	---	---



5

0	0	1	0
---	---	---	---



0	1	1	1
---	---	---	---



1	1	0	0
---	---	---	---

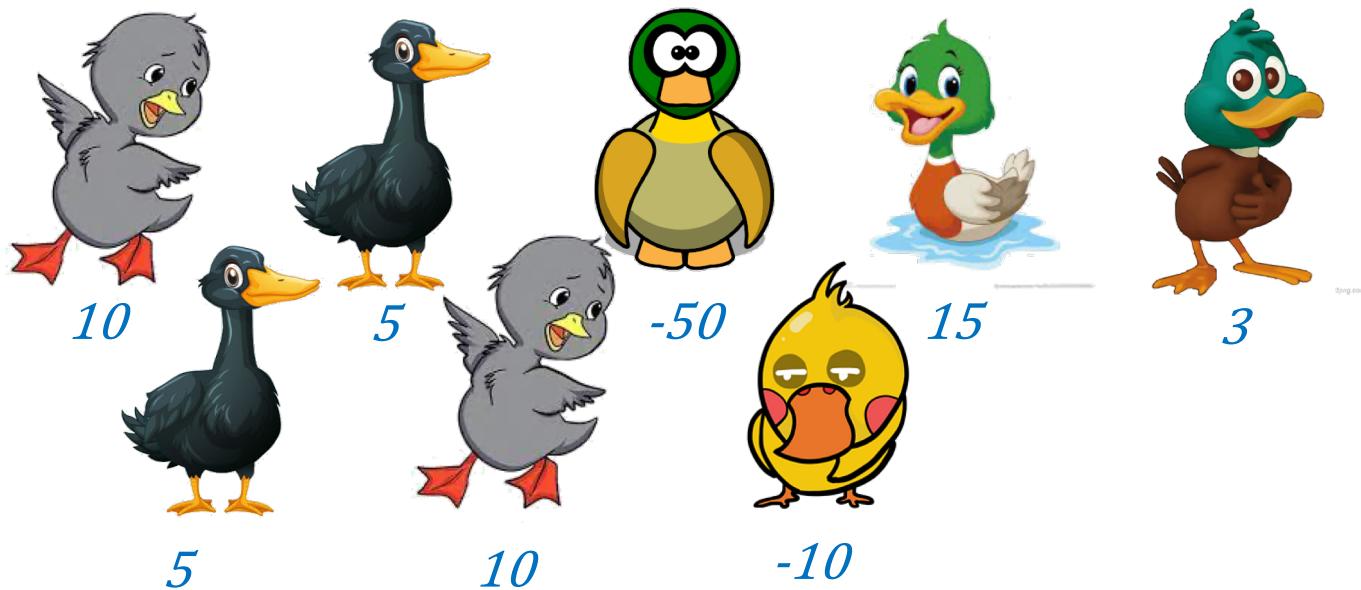


假设表达出分数为15、3

遗传算法

第一年

评测分数



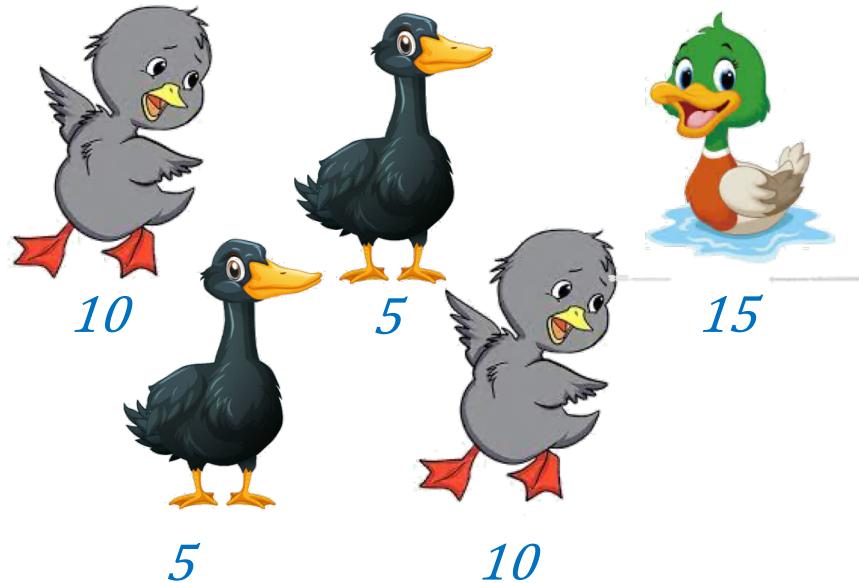
遗传算法

第一年

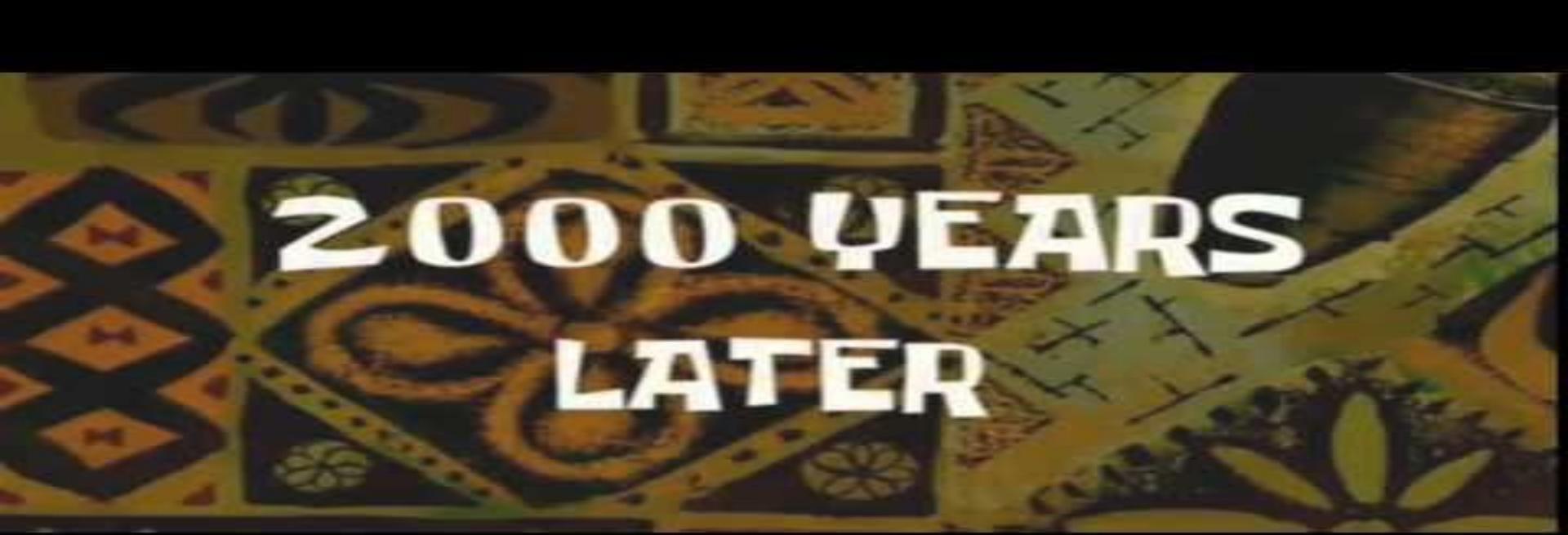


遗传算法

第二年



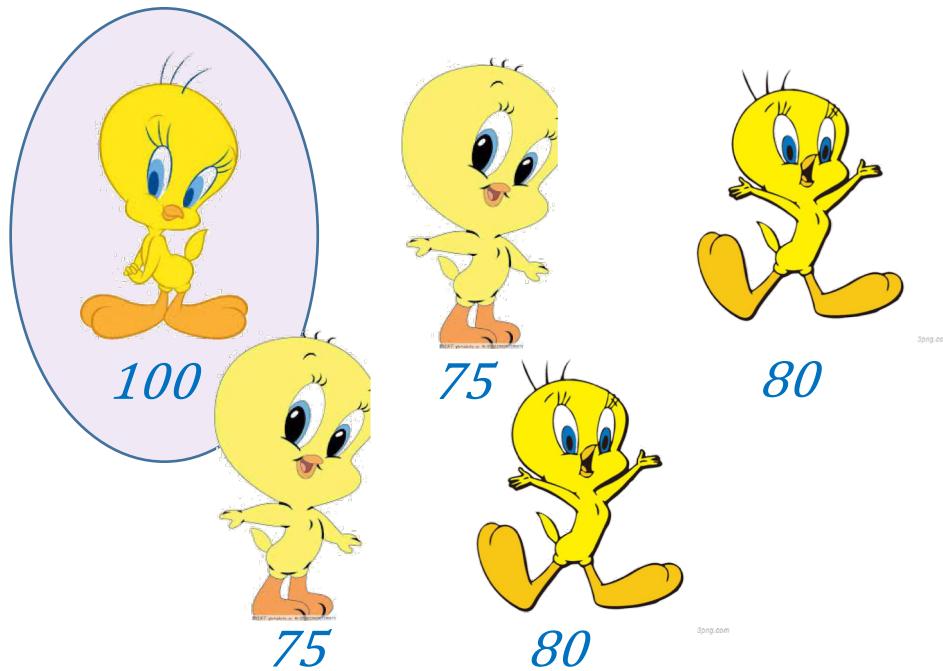
遗传算法



2000 YEARS
LATER

遗传算法

第N年



遗传算法

遗传算法

- 0 初始话当前种群 $P(t)$ 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

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

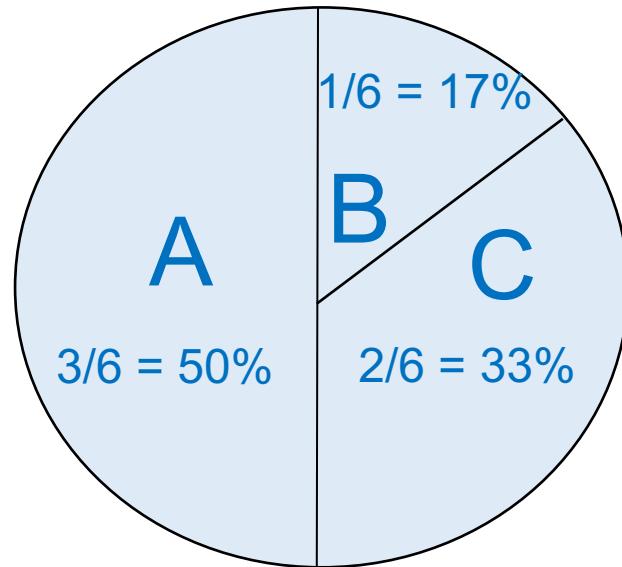
遗传算法

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

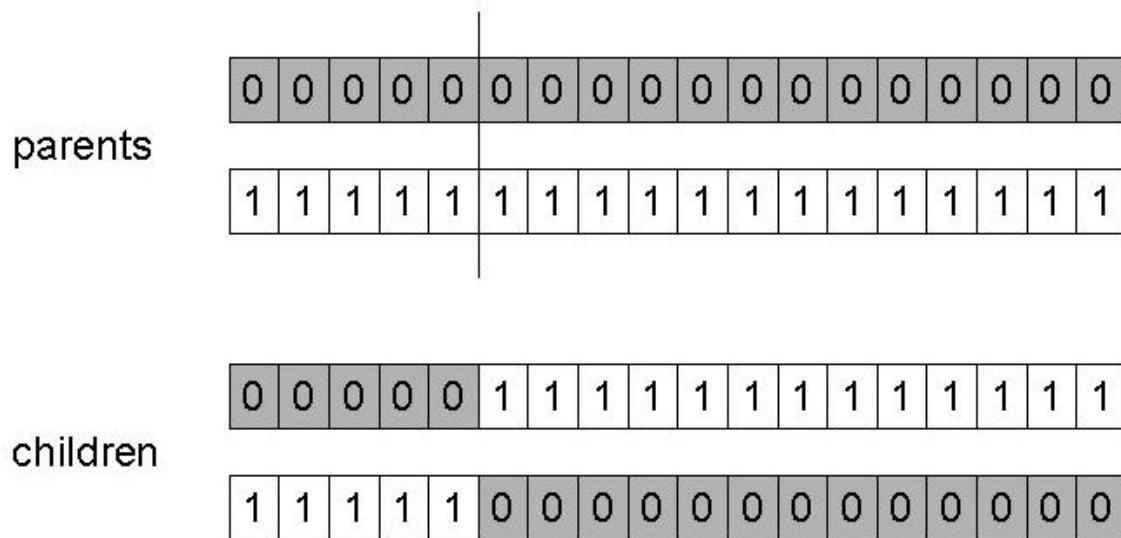
- 0 初始化当前种群 $P(t)$
- 1 评估当前种群 $P(t)$
- 2 for $t=1$ to ∞ :
- 3 根据当前种群 $P(t)$ ，确定父代种群 $P_P(t)$
- 4 根据父代种群 $P_P(t)$ ，通过基因交叉生成子代种群 $P_C(t)$
- 5 对子代种群 $P_C(t)$ 进行基因突变
- 6 评估子代种群 $P_C(t)$
- 7 根据当前种群 $P(t)$ 和子代种群 $P_C(t)$ ，筛选出新的当前种群 $P(t + 1)$
- 8 检查循环终止条件，若满足则结束算法

先配对（顺序/随机）
再进行基因交叉

遗传算法

基因交叉

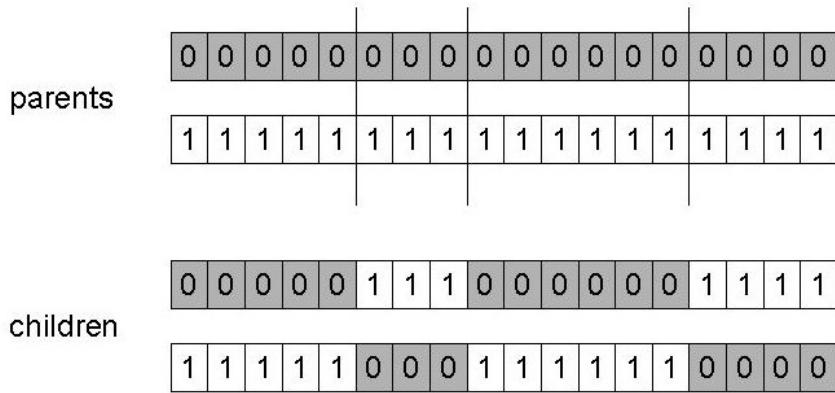
随机选取一个点， 将基因序列分成两段， 交叉生成后代基因序列



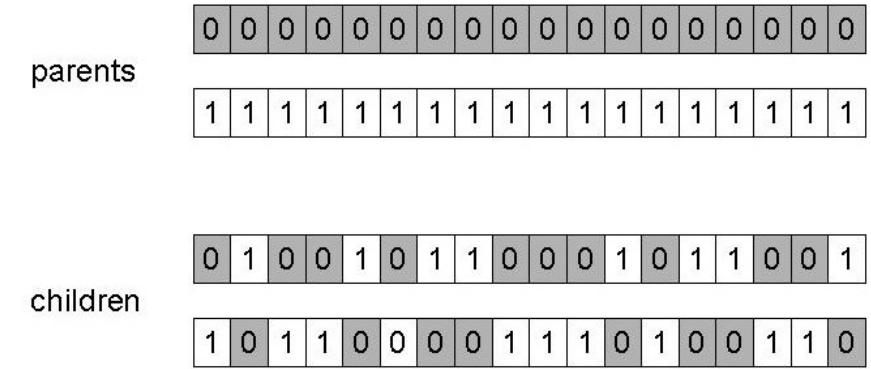
遗传算法

基因交叉（其它版本）

多点交叉



均匀交叉



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

parent

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

child

0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

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 \leftrightarrow 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

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 $01101 \leftrightarrow 13$
- 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
---	---	---	---	---	---	---	---	---

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
---	---	---	---	---	---	---	---	---

基因交叉

1	8	3	2	1	6	9	3	5
---	---	---	---	---	---	---	---	---

不成完整路径

4	2	7	8	6	5	4	9	7
---	---	---	---	---	---	---	---	---

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？

1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---

9	3	7	8	2	6	5	1	4
---	---	---	---	---	---	---	---	---

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字

1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---

1、2、3、8、9未安置



			4	5	6	7		
--	--	--	---	---	---	---	--	--

9	3	7	8	2	6	5	1	4
---	---	---	---	---	---	---	---	---

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

1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---

1、2、3、8、9未安置



3	8	2	4	5	6	7	1	9
---	---	---	---	---	---	---	---	---

9	3	7	8	2	6	5	1	4
---	---	---	---	---	---	---	---	---

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

1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---



1	5	3	4	2	6	7	8	9
---	---	---	---	---	---	---	---	---

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

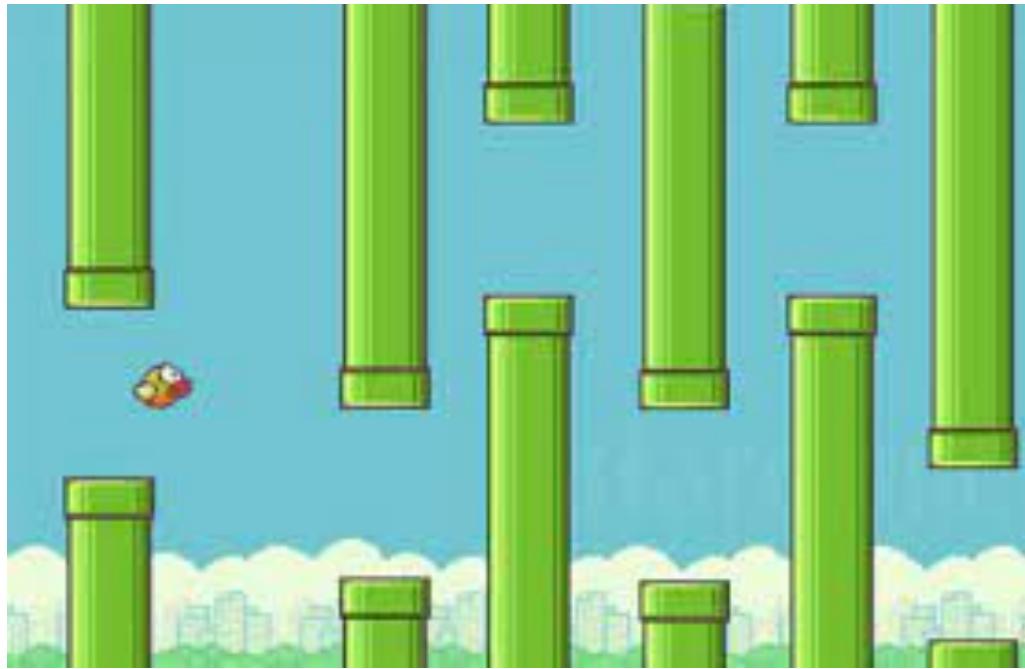
1	2	3	4	5	6	7	8	9
---	---	---	---	---	---	---	---	---



1	2	5	3	4	6	7	8	9
---	---	---	---	---	---	---	---	---

保护多对城市之间的相邻关系

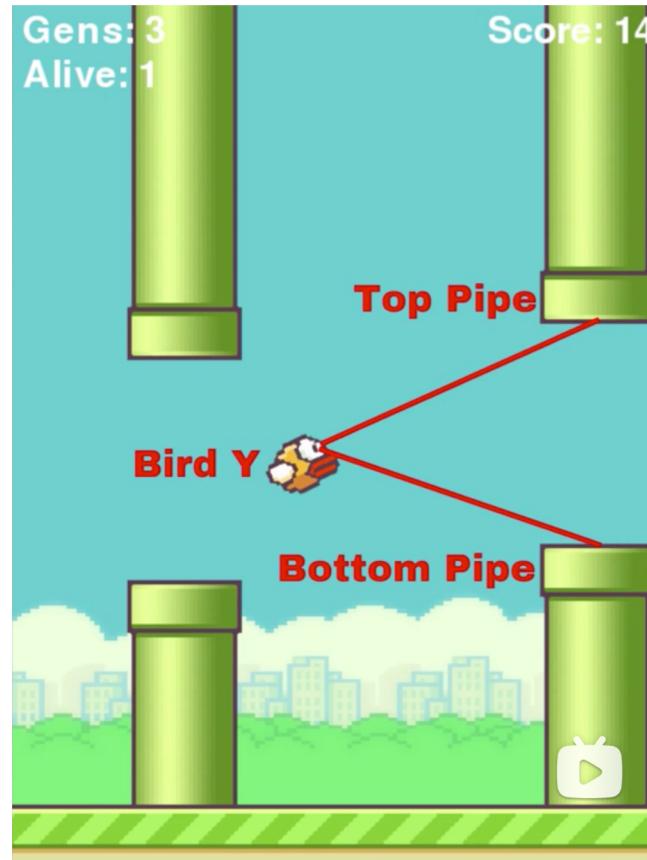
像素鸟游戏



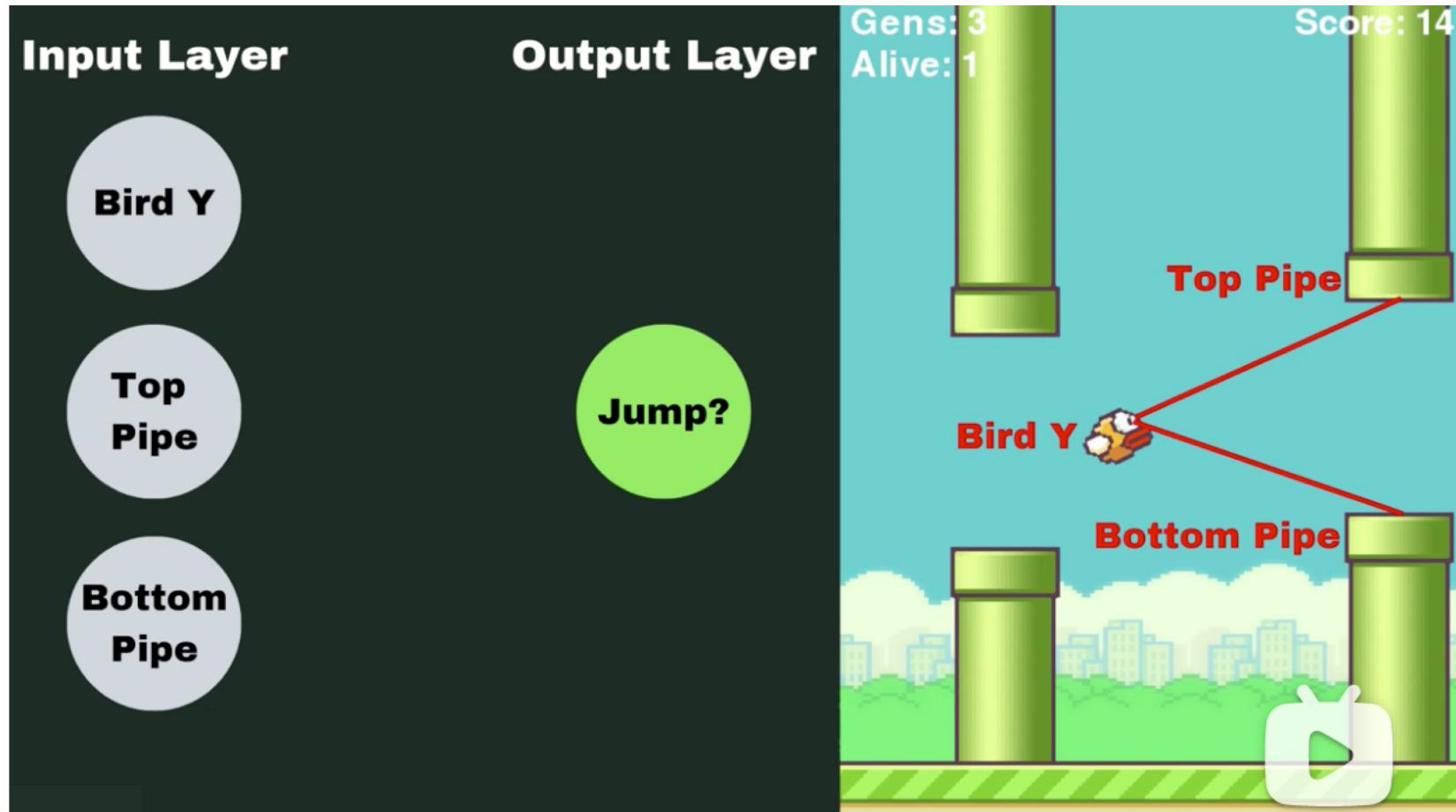
Tech With Tim -- AI Teaches Itself to Play Flappy Bird - Using NEAT Python!

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

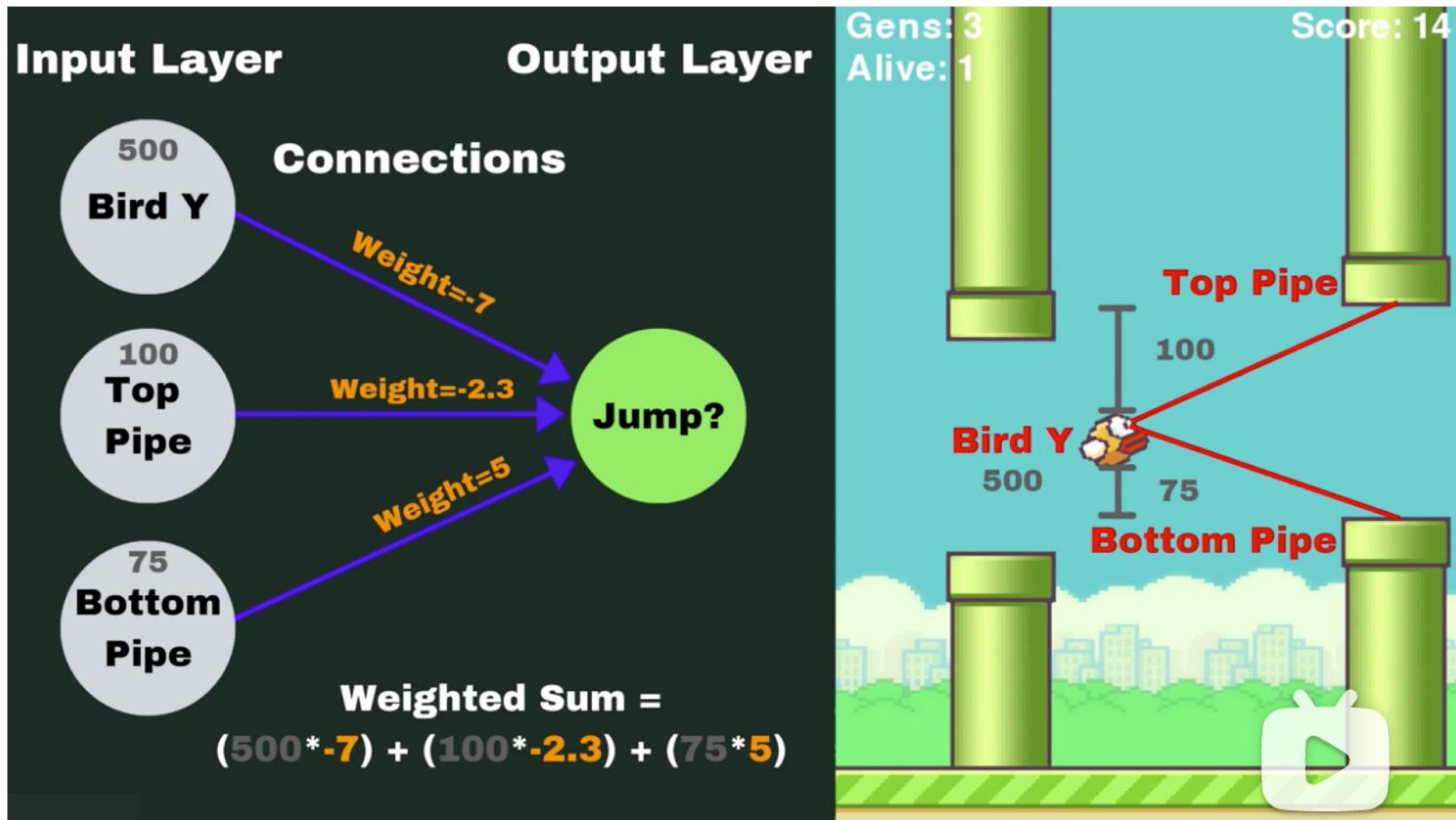
像素鸟游戏



像素鸟游戏

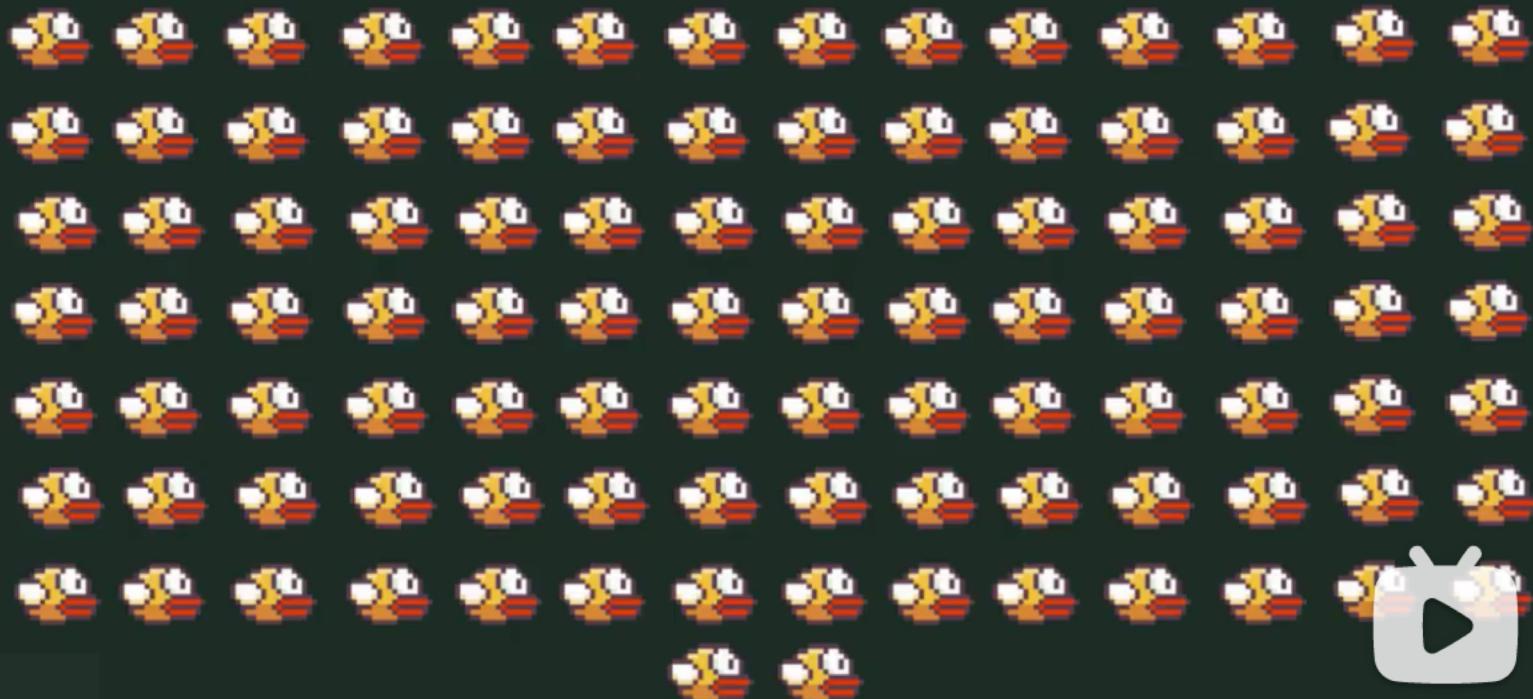


像素鸟游戏



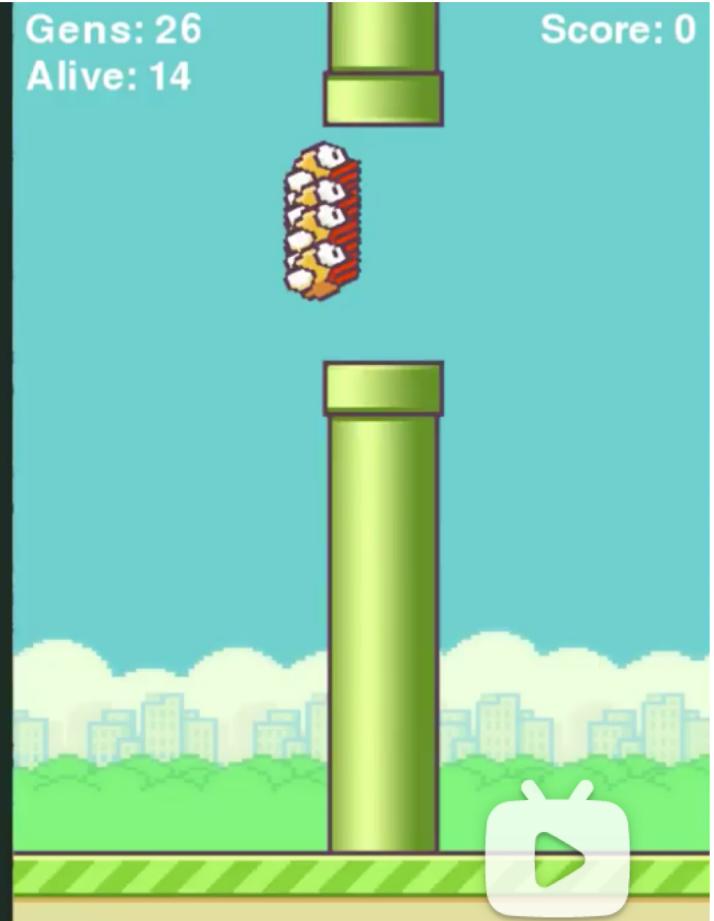
像素鸟游戏

Initial Population

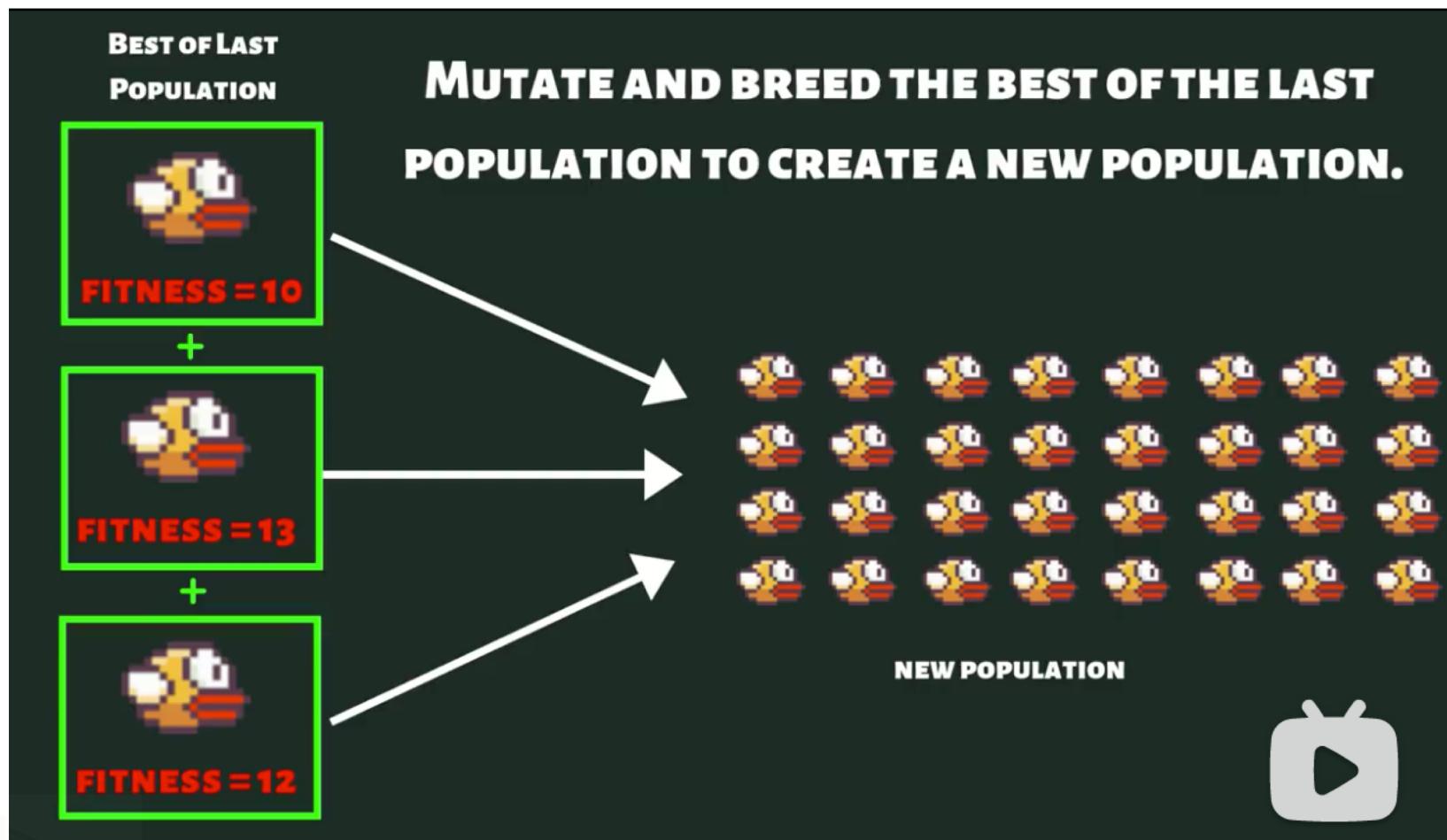


像素鸟游戏

WE TEST EACH OF THESE
NETWORKS AND EVALUATE THEIR
FITNESS (HOW WELL THEY DO)



像素鸟游戏



像素鸟游戏

Evolving Neural Networks through Augmenting Topologies

3214引用量

Kenneth O. Stanley

Department of Computer Sciences, The University of Texas at Austin, Austin, TX
78712, USA

kstanley@cs.utexas.edu

Risto Miikkulainen

Department of Computer Sciences, The University of Texas at Austin, Austin, TX
78712, USA

risto@cs.utexas.edu

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

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

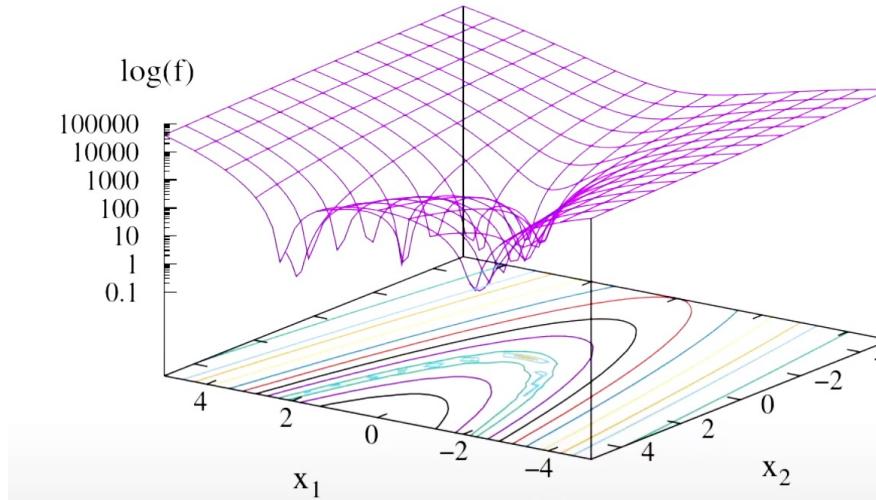
如何求解解空间连续的问题？

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 $\{a_1, \dots, a_L\} \in \{0,1\}^L$ 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

$$\Gamma(a_1, \dots, a_L) = x + \frac{y - x}{2^L - 1} \cdot \left(\sum_{j=0}^{L-1} a_{L-j} \cdot 2^j \right) \in [x, y] \quad \text{若是多维变量则每个维度都如此处理}$$

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

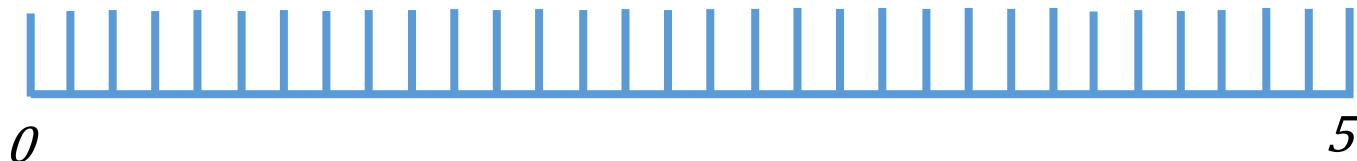
- 定义函数 $\Gamma(a_1, \dots, a_L)$ 计算离散解 (a_1, \dots, a_L) 在连续解空间对应的值

$$\Gamma(a_1, \dots, a_L) = x + \frac{y - x}{2^L - 1} \cdot \left(\sum_{j=0}^{L-1} a_{L-j} \cdot 2^j \right) \in [x, y] \quad \text{若是多维变量则每个维度都如此处理}$$

- 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

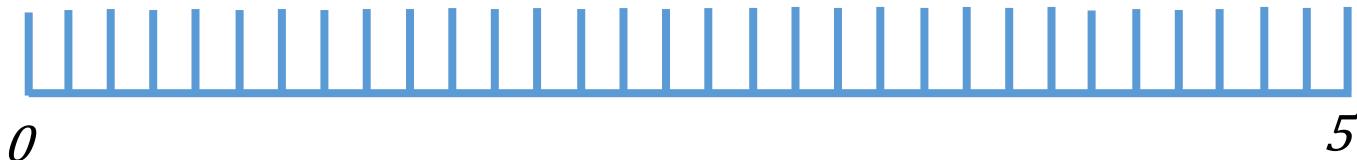
实数 15/31 对应基因表达为 00011

...

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

...

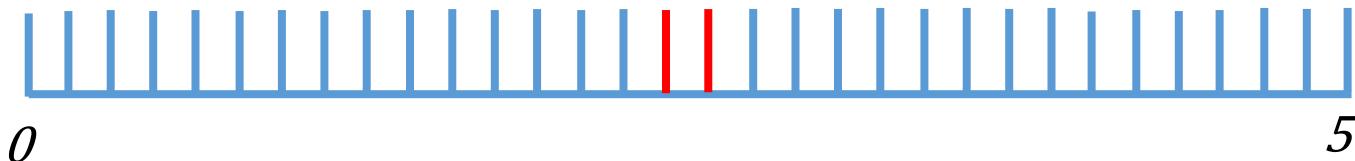
实数 5 对应基因表达为 11111

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

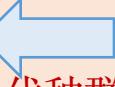
连续解空间

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

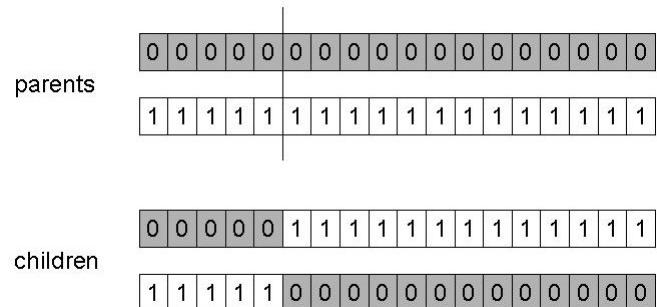
- 0 初始化当前种群 $P(t)$
- 1 评估当前种群 $P(t)$
- 2 for $t=1$ to ∞ :
- 3 根据当前种群 $P(t)$, 确定父代种群 $P_P(t)$  不受影响
- 4 根据父代种群 $P_P(t)$, 通过基因交叉生成子代种群 $P_C(t)$
- 5 对子代种群 $P_C(t)$ 进行基因突变
- 6 评估子代种群 $P_C(t)$
- 7 根据当前种群 $P(t)$ 和子代种群 $P_C(t)$, 筛选出新的当前种群 $P(t + 1)$
- 8 检查循环终止条件, 若满足则结束算法

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

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

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

基因交叉

二进制遗传算法
(binary-coded genetic algorithm)

实码遗传算法
(real-coded genetic algorithm)

parents	<table border="1"><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																						
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1																						
children	<table border="1"><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1																						
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																						

仿二进制基因交叉
SBX (simulated binary crossover)

基因突变

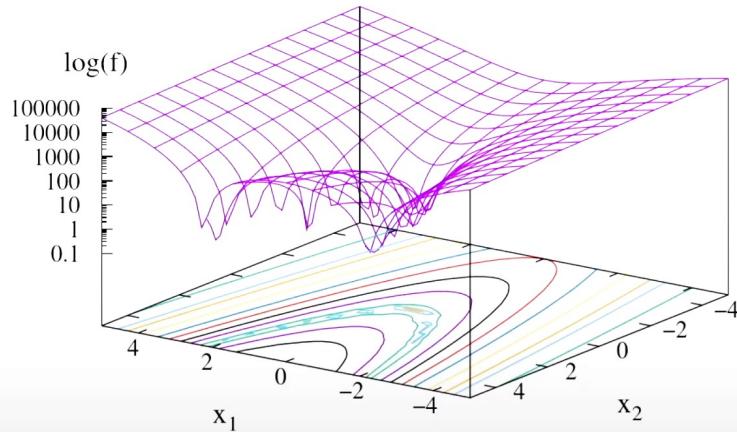
<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
<table border="1"><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr></table>	0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1	0	1
0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1	0	1	

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

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

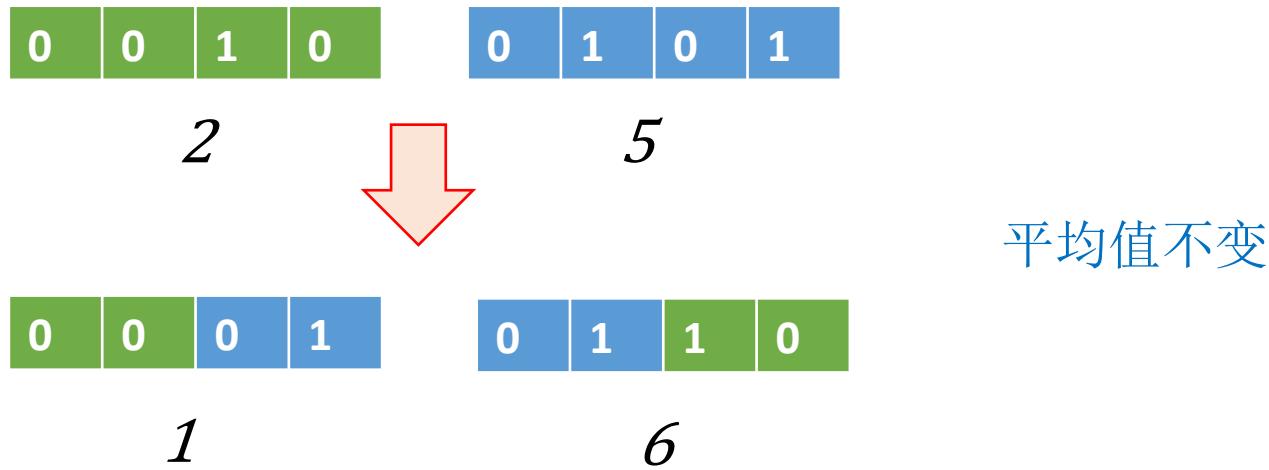
选择允许配对繁殖的解

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？



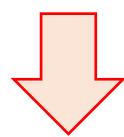
仿二进制基因交叉SBX

二进制编码中基因交叉的本质是什么？

0 0 1 0 0 1 0 1

2

5



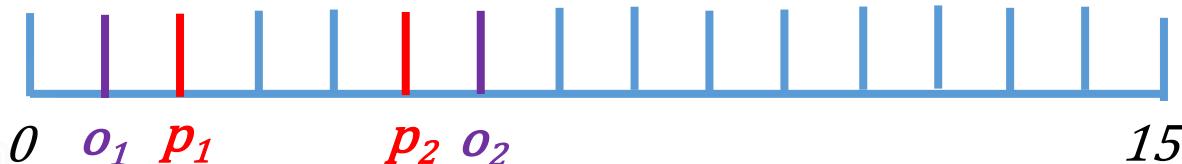
0 0 0 1

0 1 1 0

1

6

平均值不变



子代间距离拉远

*p: parents
o: offspring*

仿二进制基因交叉SBX

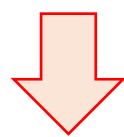
二进制编码中基因交叉的本质是什么？



2



5



3



4

平均值不变



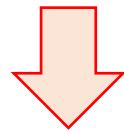
子代间距离拉近

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？

a1	a2	a3	a4
----	----	----	----

b1	b2	b3	b4
----	----	----	----



a1	a2	a3	b4
----	----	----	----

b1	b2	b3	a4
----	----	----	----

平均值不变

(能否证明？)

a1	a2	b3	b4
----	----	----	----

b1	b2	a3	a4
----	----	----	----

a1	b2	b3	b4
----	----	----	----

b1	a2	a3	a4
----	----	----	----

b1	a2	a3	b4
----	----	----	----

a1	b2	b3	a4
----	----	----	----

.....

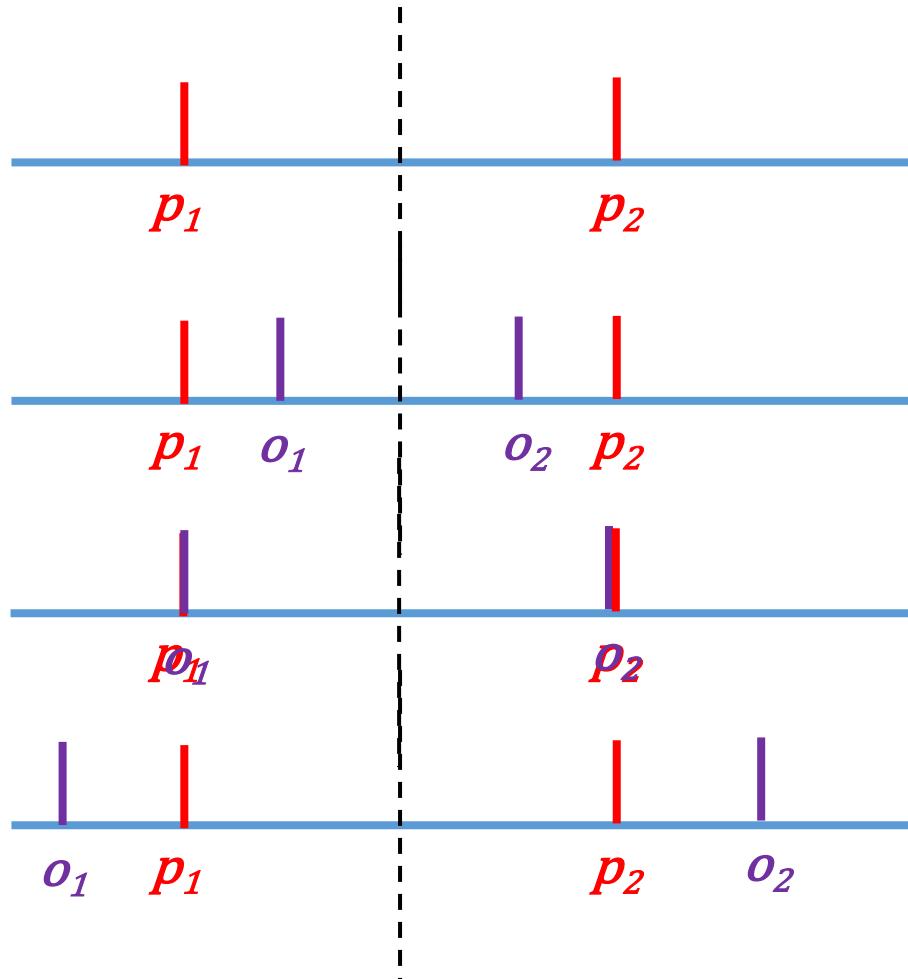
仿二进制基因交叉SBX

二进制编码中基因交叉的本质是什么？

子代之间更相似

子代不变

子代之间更不同



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？

子代之间更相似

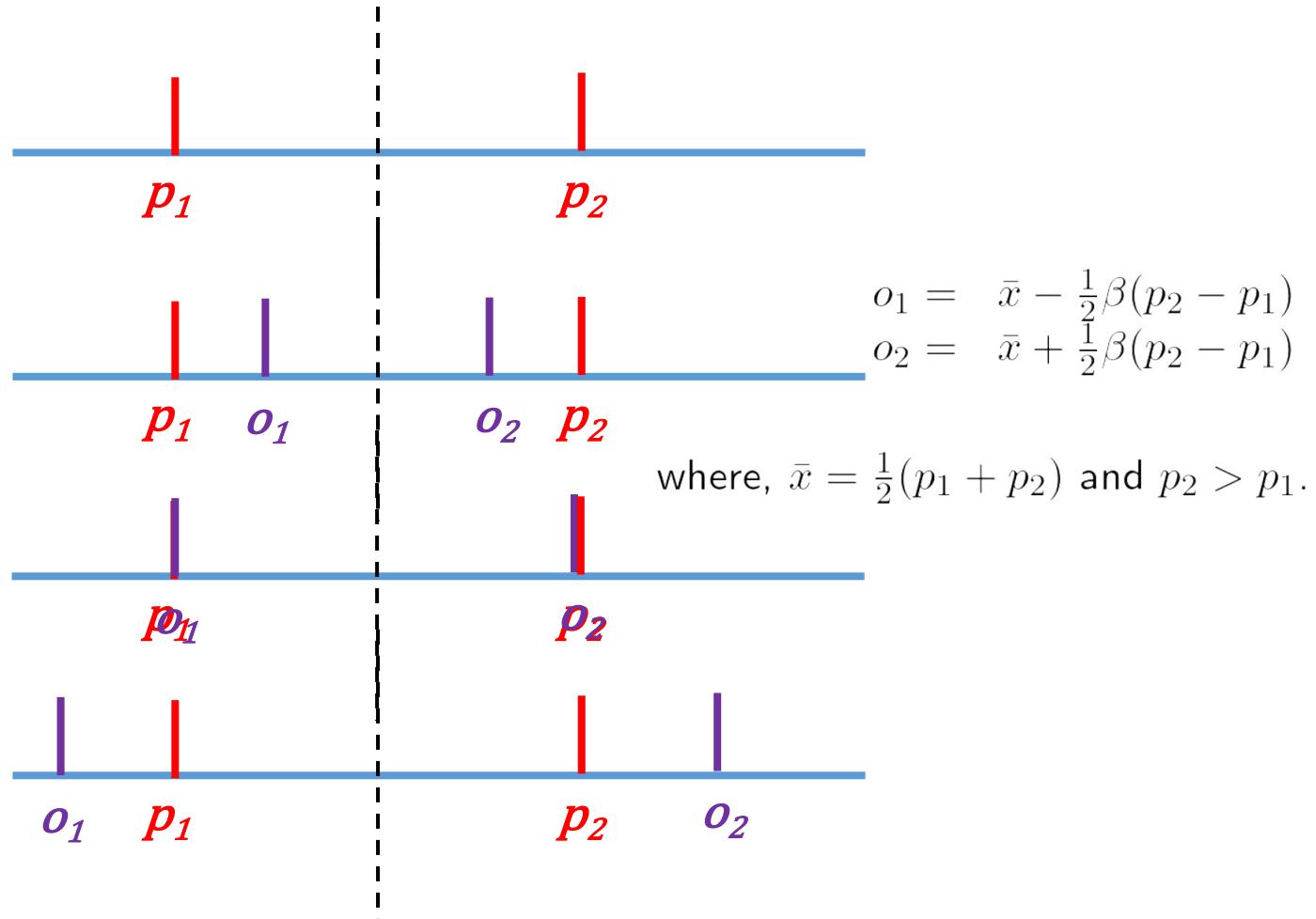
$$\beta < 1$$

子代不变

$$\beta = 1$$

子代之间更不同

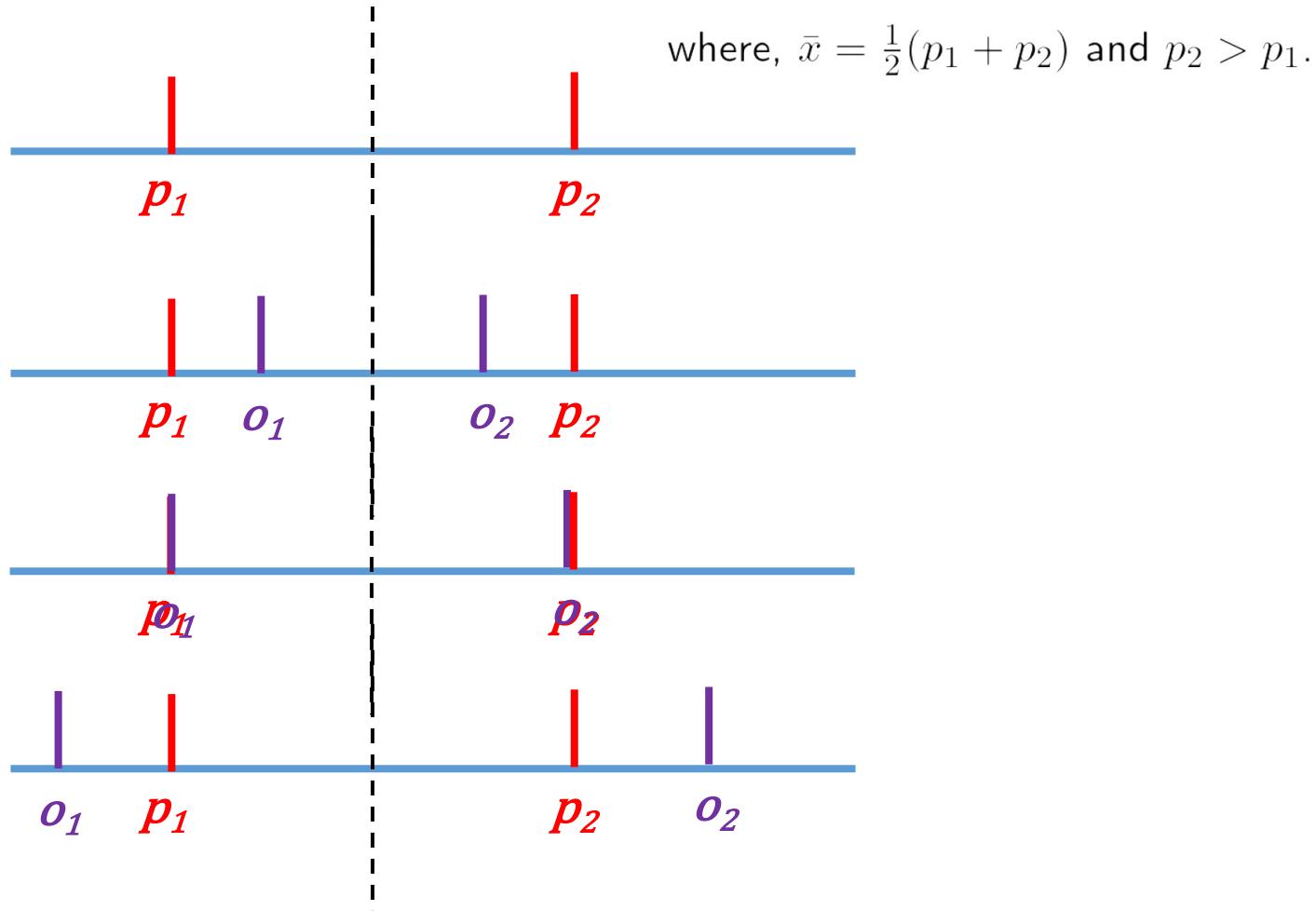
$$\beta > 1$$



仿二进制基因交叉SBX

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？



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 $p_2 > p_1$.

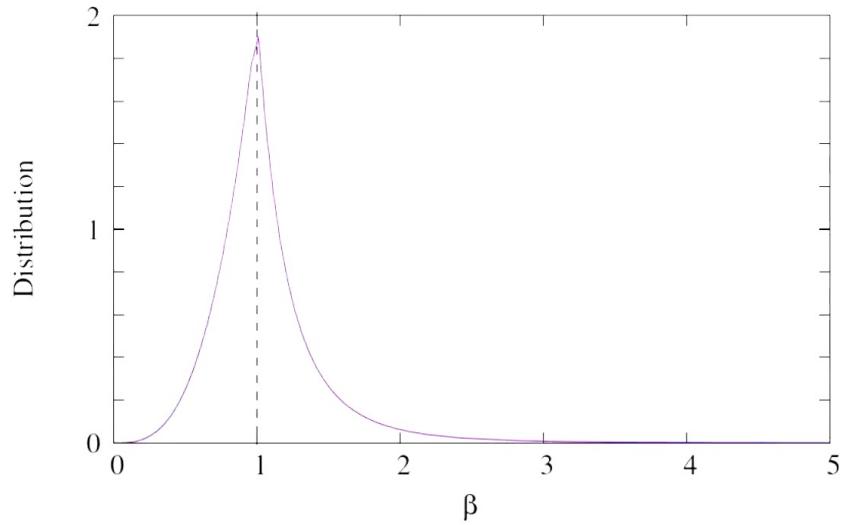
β怎么取值？

- Probability distribution function:

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

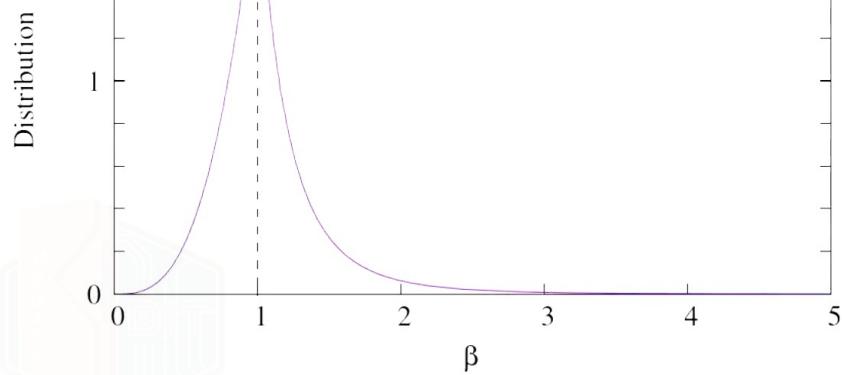
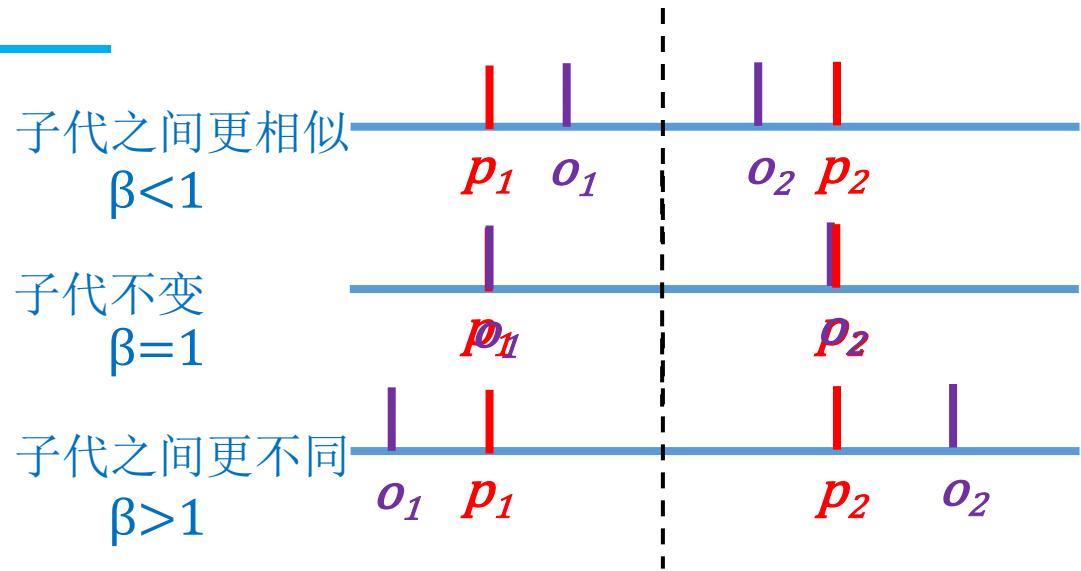
- η_c is the SBX crossover operator distribution factor that is set by us.

η_c 影响分布形状



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：

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

注意对每个维度都分别取一个新的 β

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	-2.350	5843.569
	7	7	-0.742	1.934	194.618
	4	8	-0.639	1.692	167.414

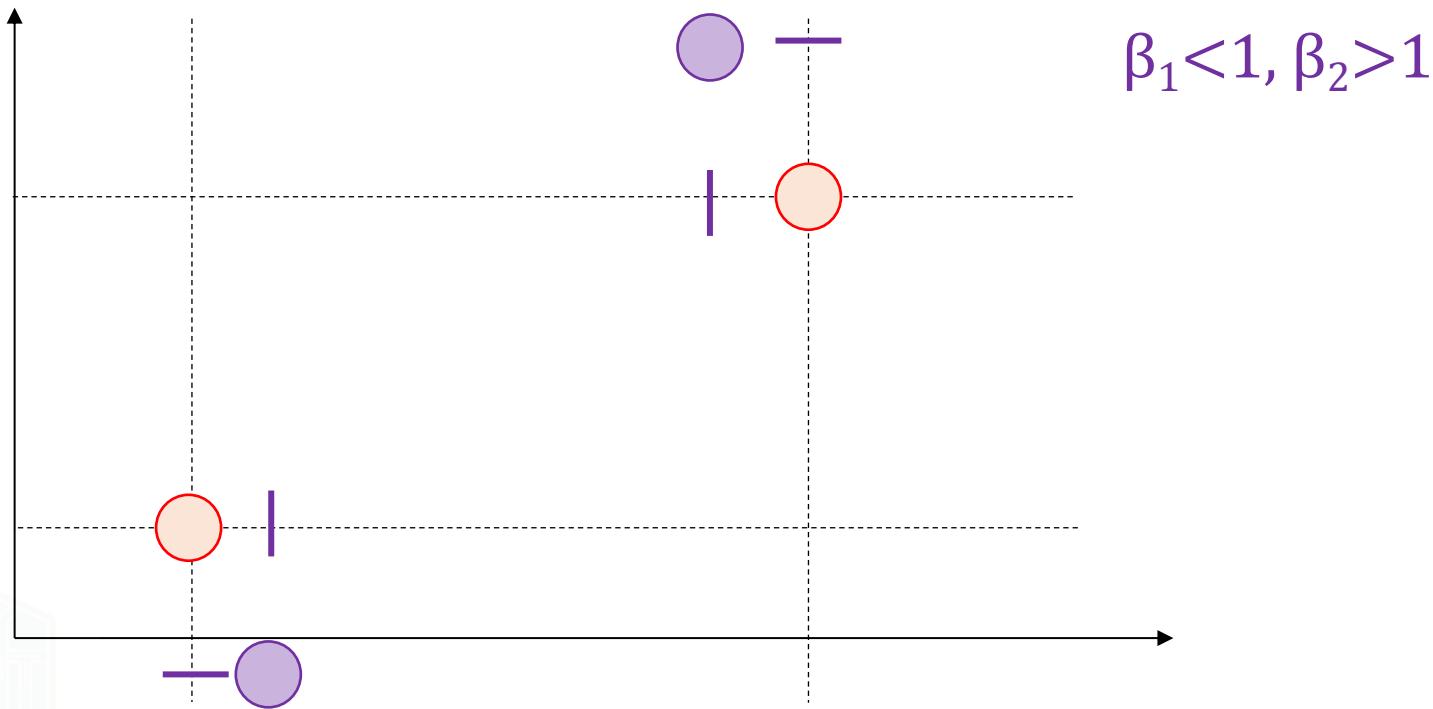
实码遗传算法

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



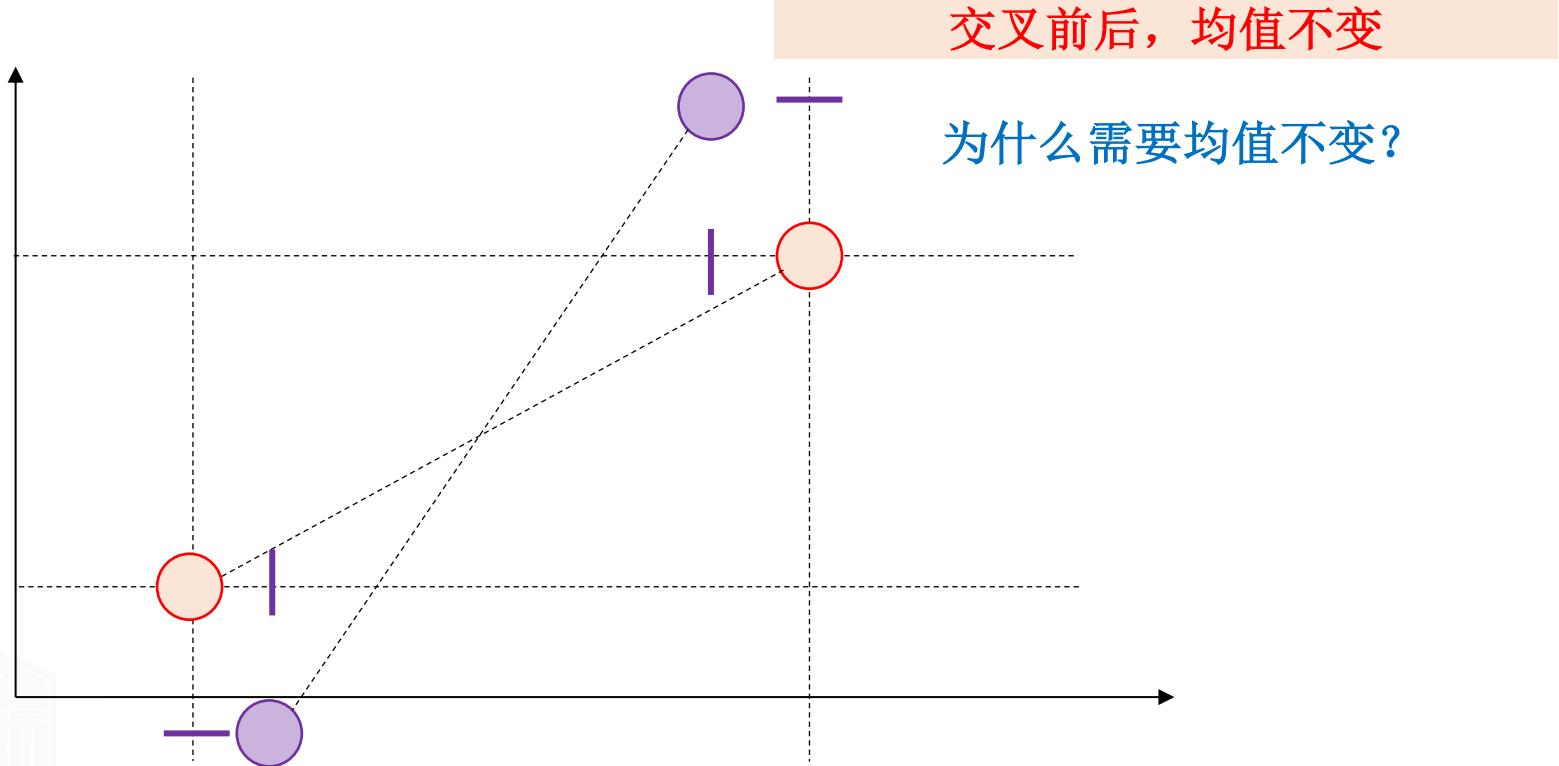
实码遗传算法

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



实码遗传算法

基因交叉

二进制遗传算法
(binary-coded genetic algorithm)

parents

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

children

0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

基因突变

实码遗传算法
(real-coded genetic algorithm)

仿二进制基因交叉

SBX (simulated binary crossover)

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

1. 均匀突变

2. 非均匀突变（如正态分布突变）

实码遗传算法

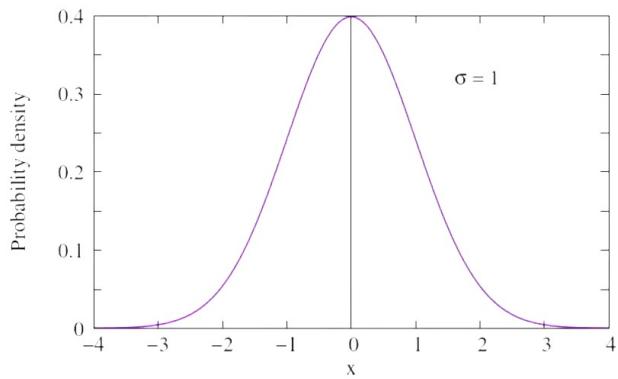
均匀突变：随机以均匀分布从定义域中取值

非均匀突变：正态分布突变、多项式突变等

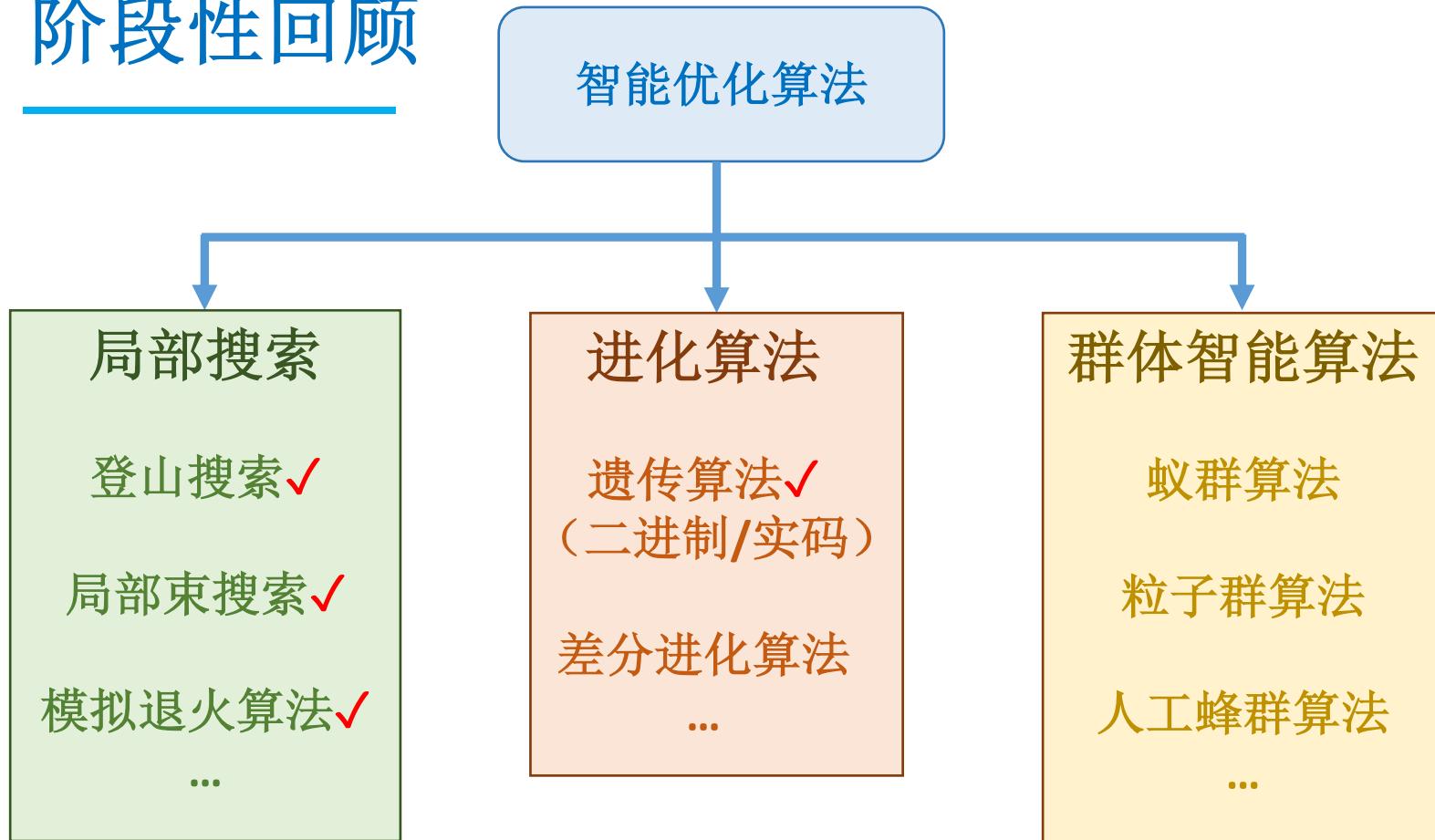
实码遗传算法

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）

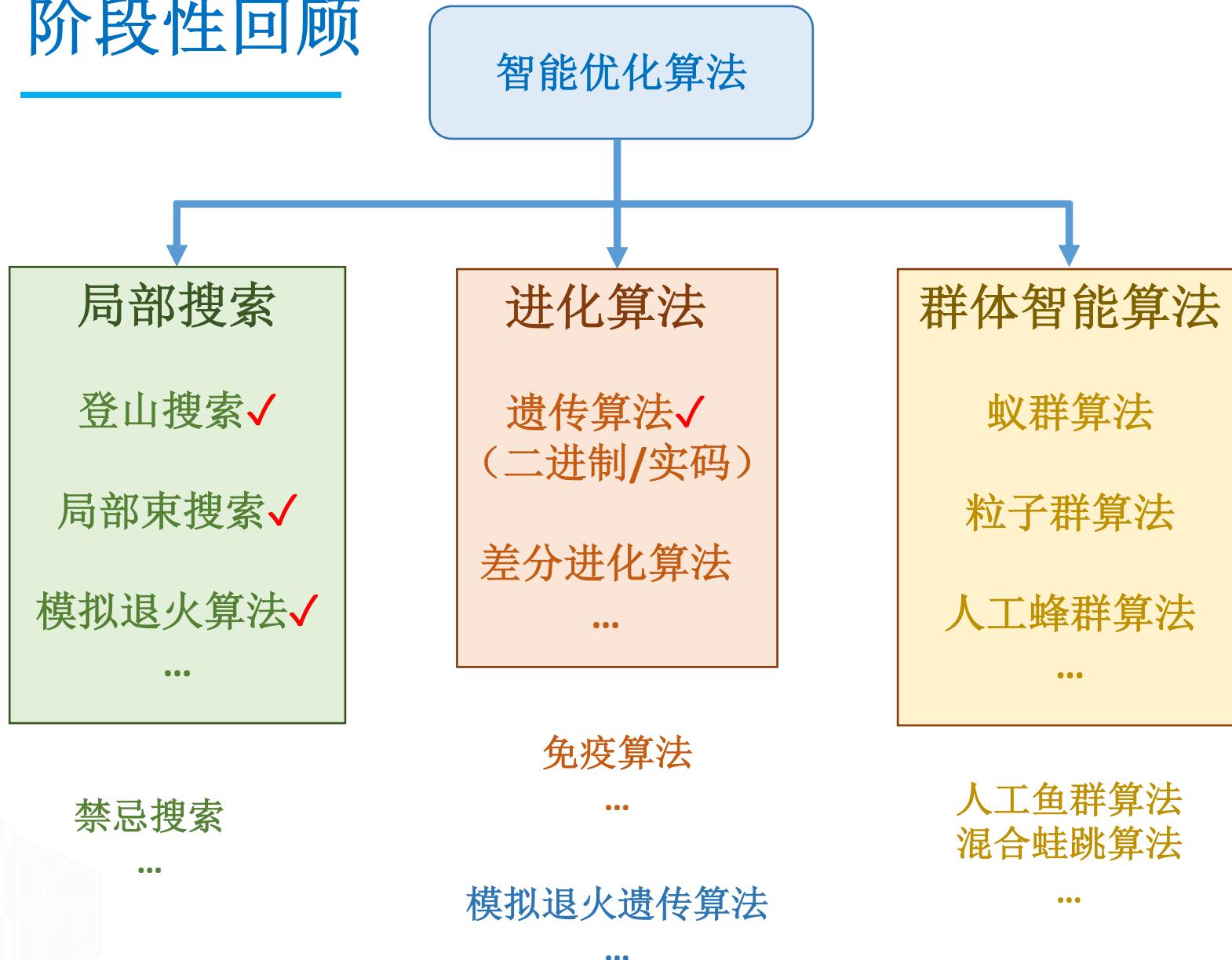
- Simple and popular method is to use a zero-mean Gaussian probability distribution:
$$y_i^{(1,t)} = x_i^{(1,t)} + N(0, \sigma_i)$$
- σ_i is a fixed user defined parameter. It must be set correctly in a problem.
- Special care must be taken to respect boundary limits on decision variables.



阶段性回顾



阶段性回顾



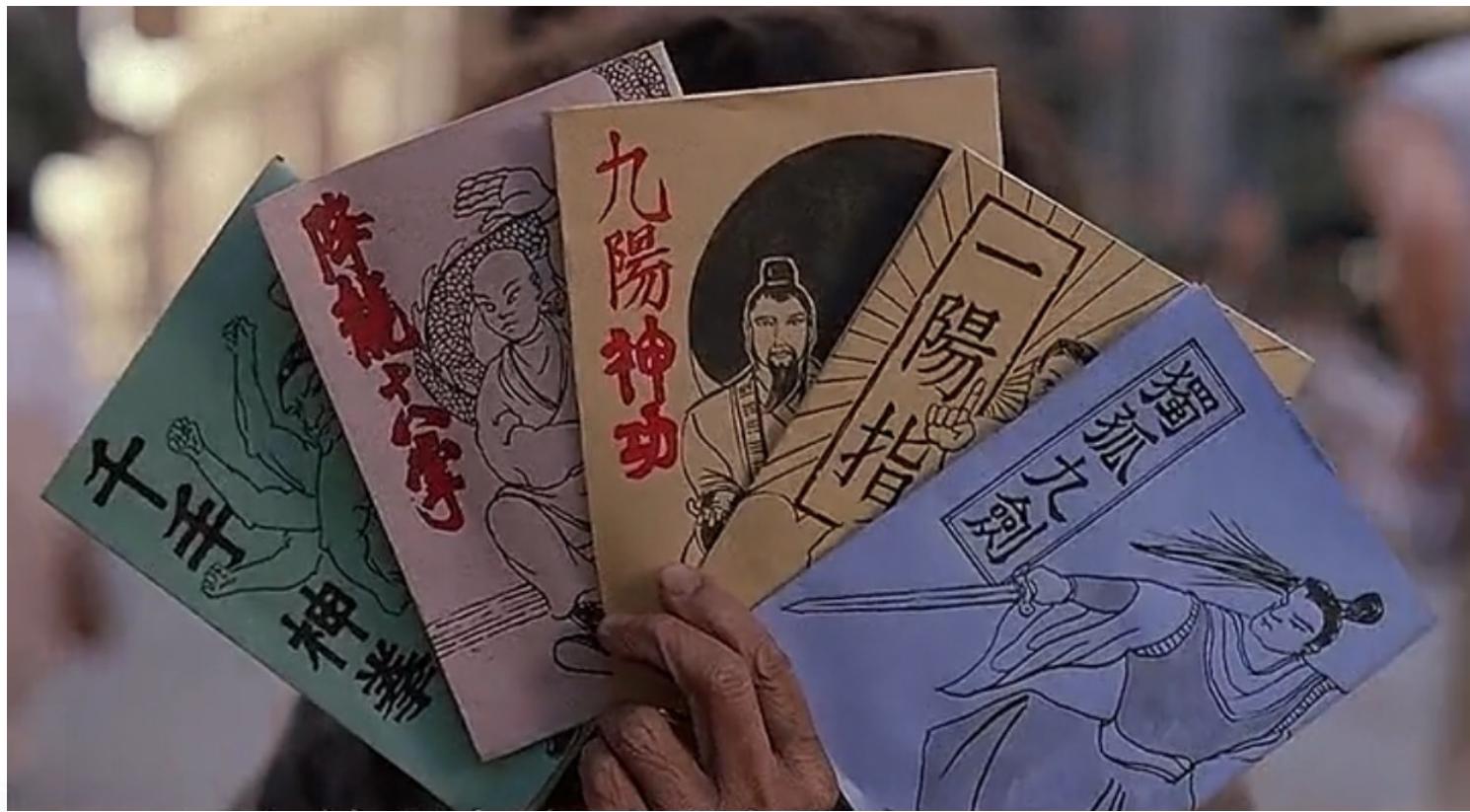
阶段性回顾

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

进化算法之差分进化算法

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？

基本介绍

差分进化（Differential Evolution, DE）

主要用于解决连续决策变量的问题

Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces

R Storn, K Price - Journal of global optimization, 1997 - Springer

A new heuristic approach for minimizing possibly nonlinear and non-differentiable continuous spacefunctions is presented. By means of an extensivetestbed it is demonstrated that the new methodconverges faster and with more certainty than manyother acclaimed ...

☆ 99 Cited by 25939 Related articles All 47 versions »

差分进化

Algorithm 1: basic DE algorithm

```
01: Generate a uniformly distributed random initial population including  $NP$  solutions that contain  
     $D$  variables according to  $X_{i,j}^0 = X_j^{\min} + \text{rand}(0, 1) \cdot (X_j^{\max} - X_j^{\min})$  ( $i \in [1, NP], j \in [1, D]$ )  
02: while termination condition is not satisfied  
03:   for  $i=1$  to  $NP$   
04:     Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
05:      $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation  
06:      $j_{rand} = \text{randind}(1, D)$  //crossover  
07:     for  $i=1$  to  $D$   
08:       if  $\text{rand}(0, 1) \leq CR$  or  $j = j_{rand}$   
09:          $U_{i,j}^G = V_{i,j}^G$   
10:       else  
11:          $U_{i,j}^G = X_{i,j}^G$   
12:       end if  
13:     end for //end crossover  
14:     if  $f(U_i^G) \leq f(X_i^G)$  //selection  
15:        $X_i^{G+1} = U_i^G$   
16:     else  
17:        $X_i^{G+1} = X_i^G$   
18:     end if //end selection  
19:   end for  
20: end while
```

差分进化

突变 和 交叉 的顺序和遗传算法相反

Algorithm 1: basic DE algorithm

```
01: Generate a uniformly distributed random initial population including  $NP$  solutions that contain  
D variables according to  $X_{i,j}^0 = X_j^{\min} + \text{rand}(0, 1) \cdot (X_j^{\max} - X_j^{\min})$  ( $i \in [1, NP], j \in [1, D]$ )  
02: while termination condition is not satisfied  
03:   for  $i=1$  to  $NP$   
04:     Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
05:      $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation  
06:      $j_{rand} = \text{randind}(1, D)$  //crossover  
07:     for  $i=1$  to  $D$   
08:       if  $\text{rand}(0, 1) \leq CR$  or  $j = j_{rand}$   
09:          $U_{i,j}^G = V_{i,j}^G$   
10:       else  
11:          $U_{i,j}^G = X_{i,j}^G$   
12:       end if  
13:     end for //end crossover  
14:     if  $f(U_i^G) \leq f(X_i^G)$  //selection  
15:        $X_i^{G+1} = U_i^G$   
16:     else  
17:        $X_i^{G+1} = X_i^G$   
18:     end if //end selection  
19:   end for  
20: end while
```

种群初始化

突变

交叉

选择下一代

差分进化

Algorithm 1: basic DE algorithm

```
01: Generate a uniformly distributed random initial population including  $NP$  solutions that contain  
D variables according to  $X_{i,j}^0 = X_j^{\min} + \text{rand}(0, 1) \cdot (X_j^{\max} - X_j^{\min})$  ( $i \in [1, NP], j \in [1, D]$ )  
02: while termination condition is not satisfied  
03:   for  $i=1$  to  $NP$   
04:     Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
05:      $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation  
06:      $j_{rand} = \text{randind}(1, D)$  //crossover  
07:     for  $i=1$  to  $D$   
08:       if  $\text{rand}(0,1) \leq CR$  or  $j = j_{rand}$   
09:          $U_{i,j}^G = V_{i,j}^G$   
10:       else  
11:          $U_{i,j}^G = X_{i,j}^G$   
12:       end if  
13:     end for //end crossover  
14:     if  $f(U_i^G) \leq f(X_i^G)$  //selection  
15:        $X_i^{G+1} = U_i^G$   
16:     else  
17:        $X_i^{G+1} = X_i^G$   
18:     end if //end selection  
19:   end for  
20: end while
```

种群初始化

- (1) 每一代种群中个体数目确定，即 NP
(2) 每一个解是 D 维向量，包含 D 个元素

差分进化

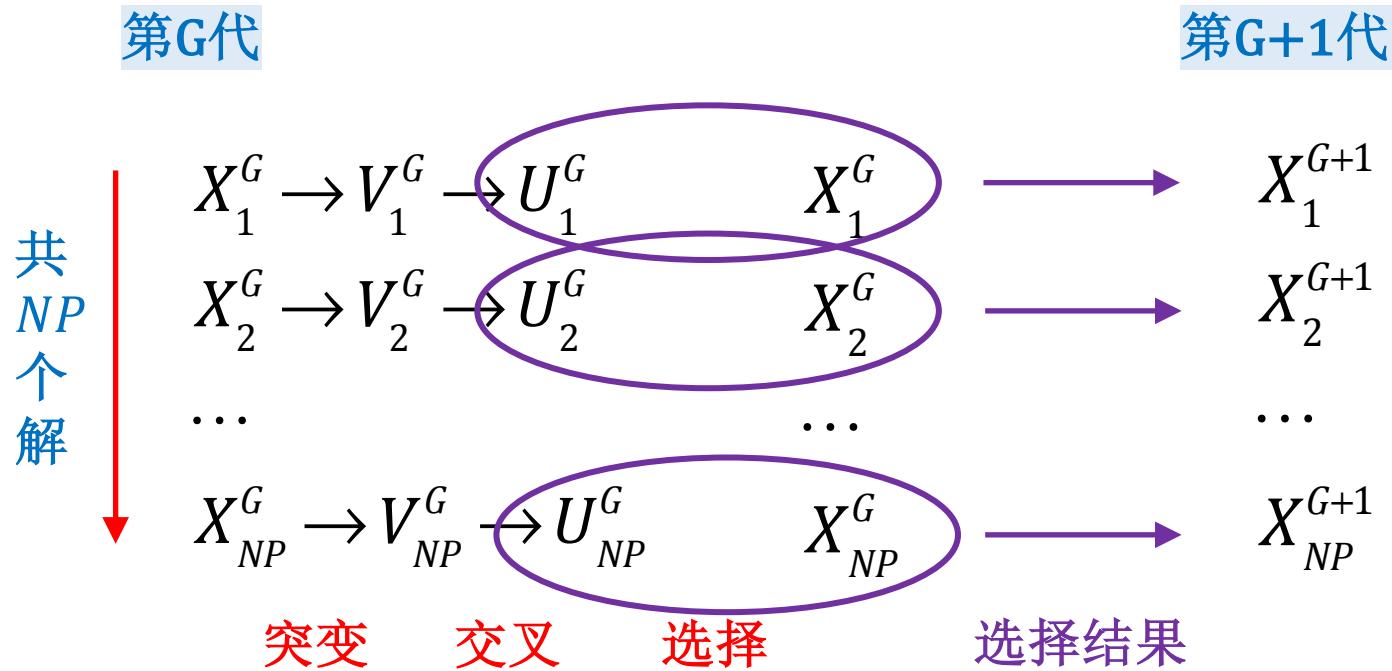
Algorithm 1: basic DE algorithm

```
01: Generate a uniformly distributed random initial population including  $NP$  solutions that contain  
D variables according to  $X_{i,j}^0 = X_j^{\min} + \text{rand}(0, 1) \cdot (X_j^{\max} - X_j^{\min})$  ( $i \in [1, NP], j \in [1, D]$ )  
02: while termination condition is not satisfied  
03:   for  $i=1$  to  $NP$   
04:     Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
05:      $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation  
06:      $j_{rand} = \text{randind}(1, D)$  //crossover  
07:     for  $i=1$  to  $D$   
08:       if  $\text{rand}(0, 1) \leq CR$  or  $j = j_{rand}$   
09:          $U_{i,j}^G = V_{i,j}^G$   
10:       else  
11:          $U_{i,j}^G = X_{i,j}^G$   
12:       end if  
13:     end for //end crossover  
14:     if  $f(U_i^G) \leq f(X_i^G)$  //selection  
15:        $X_i^{G+1} = U_i^G$   
16:     else  
17:        $X_i^{G+1} = X_i^G$   
18:     end if //end selection  
19:   end for  
20: end while
```

每一代 NP 个解分别“进化”

差分进化

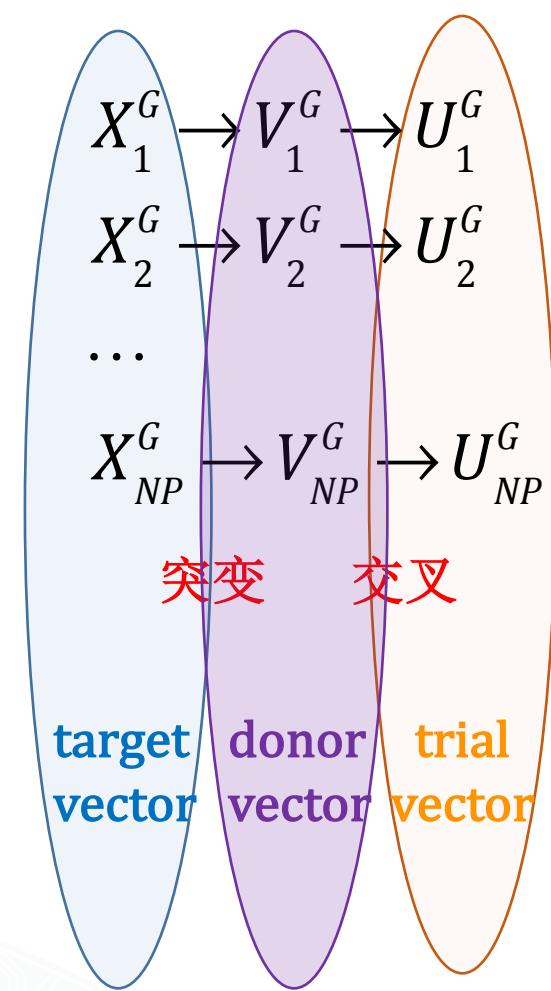
每一代 NP 个解分别“进化”



注意以上都是向量表达

在遗传算法中，繁殖（基因交叉）是以对为单位进行

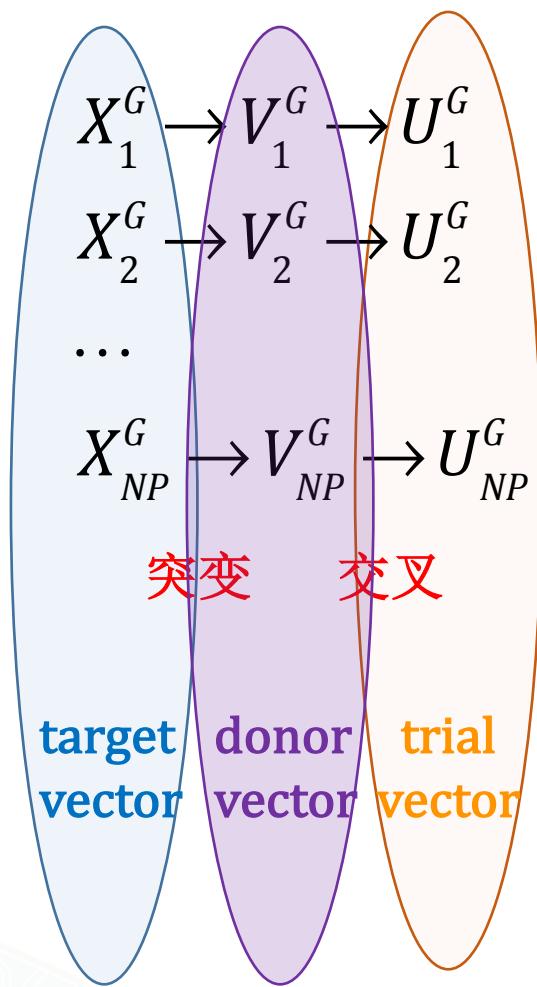
差分进化



突变:

为靶向量(target vector)确定
捐赠向量(donor vector)

差分进化



突变:

为靶向量(target vector)确定
捐赠向量(donor vector)

Algorithm 1: basic DE algorithm

```
01: Generate a uniformly distributed random initial population including  $NP$  solutions that contain  
D variables according to  $X_{i,j}^0 = X_j^{\min} + \text{rand}(0, 1) \cdot (X_j^{\max} - X_j^{\min})$  ( $i \in [1, NP], j \in [1, D]$ )  
02: while termination condition is not satisfied  
03:   for  $i=1$  to  $NP$   
04:     Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
05:      $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation  
06:      $j_{\text{rand}} = \text{randind}(1, D)$  //crossover  
07:     for  $i=1$  to  $D$   
08:       if  $\text{rand}(0, 1) \leq CR$  or  $j = j_{\text{rand}}$   
09:          $U_{i,j}^G = V_{i,j}^G$   
10:       else  
11:          $U_{i,j}^G = X_{i,j}^G$   
12:       end if  
13:     end for                                //end crossover  
14:     if  $f(U_i^G) \leq f(X_i^G)$                 //selection  
15:        $X_i^{G+1} = U_i^G$   
16:     else  
17:        $X_i^{G+1} = X_i^G$   
18:     end if                                //end selection  
19:   end for  
20: end while
```

突变

```
Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
 $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation
```

比如为 $i = 1$ 靶向量找捐赠向量

$$X_1^G, X_2^G, X_3^G, X_4^G, \dots, X_{NP}^G$$

不能选

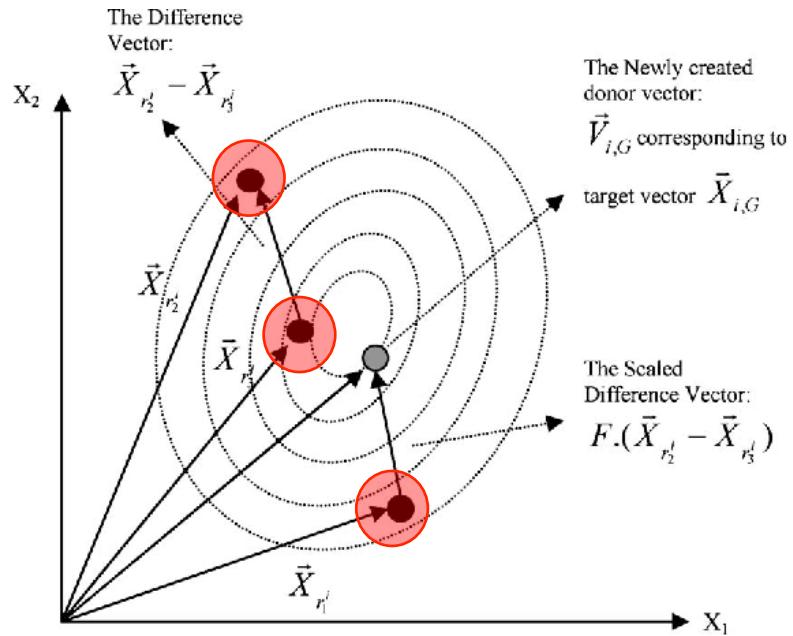
选三个不同的向量

$F \in [0,2]$ 是自定义的系数

差分进化算法需要 NP 至少为 4

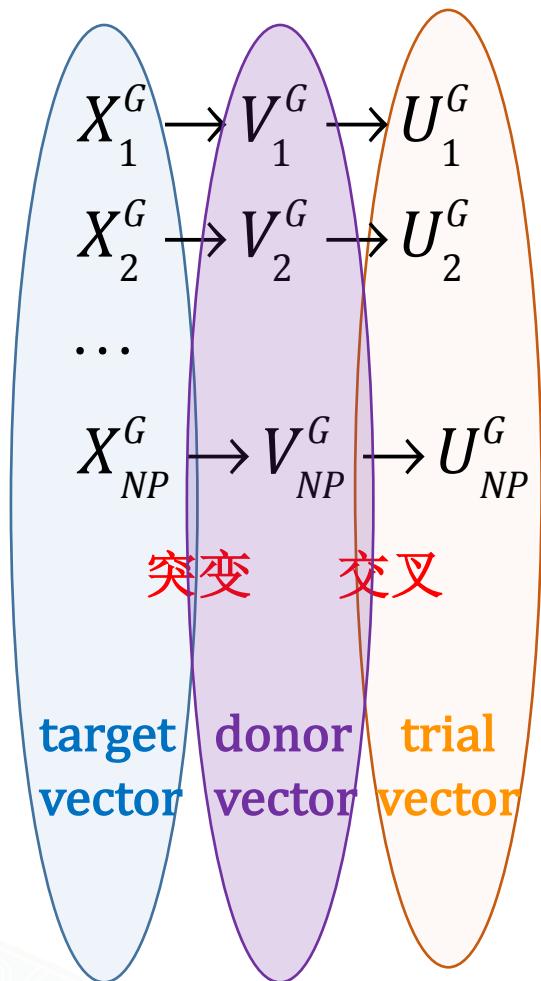
突变

```
Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
 $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation
```



g. 2. Illustrating a simple DE mutation scheme in 2-D parametric space.

差分进化



交叉: 根据靶向量(target vector)和捐赠向量(donor vector)确定试验向量(trial vector)

Algorithm 1: basic DE algorithm

```
01: Generate a uniformly distributed random initial population including  $NP$  solutions that contain  
02:  $D$  variables according to  $X_{i,j}^0 = X_j^{\min} + \text{rand}(0, 1) \cdot (X_j^{\max} - X_j^{\min})$  ( $i \in [1, NP]$ ,  $j \in [1, D]$ )  
03: while termination condition is not satisfied  
04:   for  $i=1$  to  $NP$   
05:     Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
06:      $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation  
07:      $j_{rand} = \text{randind}(1, D)$  //crossover  
08:     for  $i=1$  to  $D$   
09:       if  $\text{rand}(0, 1) \leq CR$  or  $j = j_{rand}$   
10:          $U_{i,j}^G = V_{i,j}^G$   
11:       else  
12:          $U_{i,j}^G = X_{i,j}^G$   
13:       end if  
14:     end for //end crossover  
15:     if  $f(U_i^G) \leq f(X_i^G)$  //selection  
16:        $X_i^{G+1} = U_i^G$   
17:     else  
18:        $X_i^{G+1} = X_i^G$   
19:     end if //end selection  
20:   end for  
21: end while
```

差分进化

```
 $j_{rand} = randind(1, D)$  //crossover  
for  $i=1$  to  $D$   
    if  $rand(0,1) \leq CR$  or  $j = j_{rand}$   
        试验向量第j个元素      $U_{i,j}^G = V_{i,j}^G$  捐赠向量第j个元素  
    else                      $U_{i,j}^G = X_{i,j}^G$  靶向量第j个元素  
    end if  
end for                    //end crossover  
确定 从捐赠向量取值的概率CR以及 $j_{rand}$ 
```

差分进化

```
 $j_{rand} = randind(1, D)$  //crossover  
for  $i=1$  to  $D$   
    if  $rand(0,1) \leq CR$  or  $j = j_{rand}$   
         $U_{i,j}^G = V_{i,j}^G$  捐赠向量第 $j$ 个元素  
    else  
         $U_{i,j}^G = X_{i,j}^G$  靶向量第 $j$ 个元素  
    end if  
end for //end crossover
```

确定 从捐赠向量取值的概率 CR 以及 j_{rand}

靶向量

2.2
3.1
0.4
2.1
0.5
2.1
3.5
4.1

比如取 $CR = 0.7$

$j_{rand} = 2$

试验向量

2.2
2.1
3.5
4.1

j_{rand} 的目的是确保至少一个元素取自捐赠向量，及确保试验向量和靶向量不同

差分进化

```
jrand = randind(1, D) //crossover  
for i=1 to D  
    if rand(0,1) ≤ CR or j = jrand  
        UGi,j = VGi,j 捐赠向量第j个元素  
    else  
        UGi,j = XGi,j 靶向量第j个元素  
    end if  
end for //end crossover
```

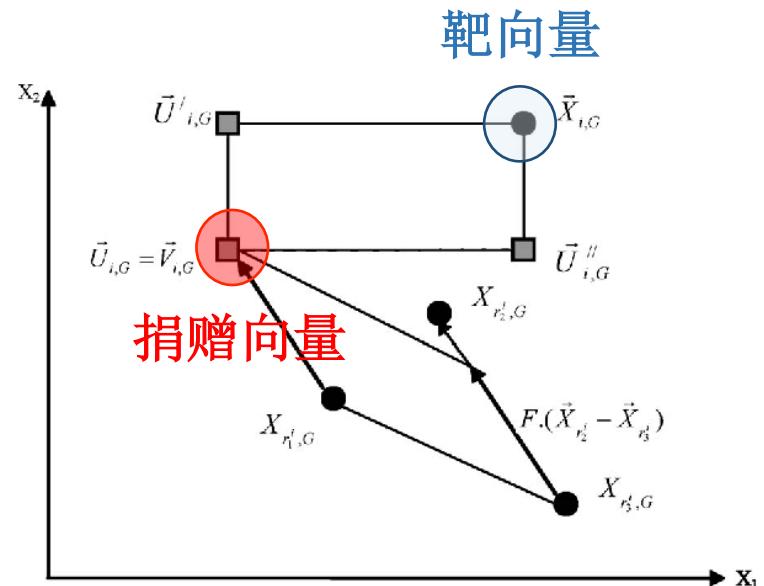


Fig. 3. Different possible trial vectors formed due to uniform/binomial crossover between the target and the mutant vectors in 2-D search space.

差分进化

Algorithm 1: basic DE algorithm

```
01: Generate a uniformly distributed random initial population including  $NP$  solutions that contain  
D variables according to  $X_{i,j}^0 = X_j^{\min} + \text{rand}(0, 1) \cdot (X_j^{\max} - X_j^{\min})$  ( $i \in [1, NP], j \in [1, D]$ )  
02: while termination condition is not satisfied  
03:   for  $i=1$  to  $NP$   
04:     Generate three random indexes  $r1, r2$  and  $r3$  with  $r1 \neq r2 \neq r3 \neq i$  //mutation  
05:      $V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G)$  //end mutation  
06:      $j_{rand} = \text{randind}(1, D)$  //crossover  
07:     for  $i=1$  to  $D$   
08:       if  $\text{rand}(0,1) \leq CR$  or  $j = j_{rand}$   
09:          $U_{i,j}^G = V_{i,j}^G$   
10:       else  
11:          $U_{i,j}^G = X_{i,j}^G$   
12:       end if  
13:     end for //end crossover  
14:     if  $f(U_i^G) \leq f(X_i^G)$  //selection  
15:        $X_i^{G+1} = U_i^G$   
16:     else  
17:        $X_i^{G+1} = X_i^G$   
18:     end if //end selection  
19:   end for  
20: end while
```

种群初始化

突变捐赠向量

交叉试验向量

对比靶向量和试验向量 选择下一代

本讲小结



进化算法之遗传算法（二进制遗传、实码遗传）



进化算法之差分进化算法



主要参考资料

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

Prakash Kotecha (IIT Guwahati) <Computer Aided Applied Single Objective Optimization> Slides

Li et al. <A novel hybrid differential evolution algorithm with modified CoDE and JADE> Paper

S. Das and P. Suganthan <Differential Evolution: A Survey of the State-of-the-Art> Paper

Kelly Fleetwood <An Introduction to Differential Evolution> Slides

谢谢！

