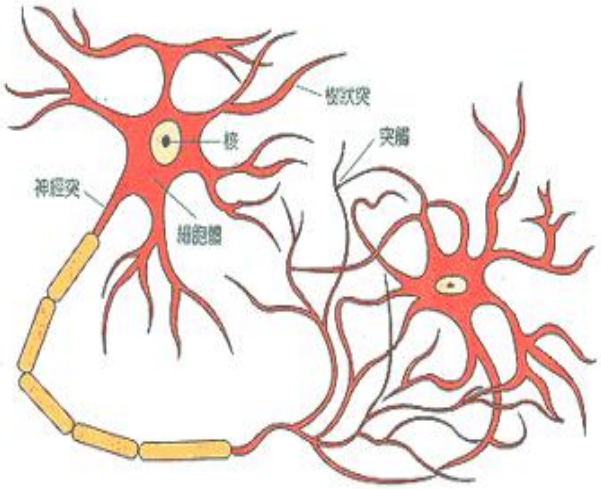
第 6 章　智能计算及其应用

智能计算，也称计算智能，主要是以生物进化的观点认识和模拟智能。

第6章 智能计算及其应用

- 向大自然学习
  - 遗传算法(GA)
    - 物竞天择，设计染色体编码，根据适应值函数进行染色体选择、交叉和变异操作，优化求解
  - 人工神经网络(ANN)
    - 模仿生物神经元，透过神经元的信息传递、训练学习、联想，优化求解
  - 模拟退火算法(SA)
    - 模模仿金属物质退火过程



第6章 智能计算及其应用

- 受自然界和生物界规律的启迪，人们根据其原理模仿设计了许多求解问题的算法，包括人工神经网络、模糊逻辑、遗传算法、DNA计算、模拟退火算法、免疫算法、量子计算、粒子群优化算法、蚁群算法、人工蜂群算法、人工鱼群算法以及细菌群体优化算法等，这些算法称为智能计算也称为计算智能(computational intelligence, CI)。

第6章 智能计算及其应用

- 6.1 进化算法的产生与发展
- 6.2 基本遗传算法
- 6.3 遗传算法的改进算法
- 6.4 遗传算法的应用
- 6.5 群智能算法产生的背景
- 6.6 粒子群优化算法及其应用
- 6.7 蚁群算法及其应用

- 掌握遗传算法的基本概念和基本方法，了解一些遗传算法的改进算法，了解遗传算法的应用实例；
- 掌握粒子群优化算法等群智能算法的原理、步骤及应用，理解群智能算法与遗传算法的区别。

第6章 智能计算及其应用

- 智能优化方法通常包括进化计算和群智能等两大类方法，是一种典型的元启发式随机优化方法，已经广泛应用于组合优化、机器学习、智能控制、模式识别、规划设计、网络安全等领域，是21世纪有关智能计算中的重要技术之一。

•元启发方法（meta-heuristics）：以遗传算法、粒子群优化等为代表

•自然启发算法：蛙跳算法、入侵杂草优化、智能水滴算法、生物地理学优化、布谷鸟搜索、蝙蝠算法，以及鱼群算法、烟花算法、拟物拟人算法等。

6.1 进化算法的产生与发展

- 进化计算(Evolutionary Computation，EC)是20世纪60年代开始发展、90 年代兴起的一门模拟生物进化与遗传规律的计算学科。已发展成为与人工神经网络、模糊逻辑相并列的智能计算支撑技术。

- 遗传算法(Genetic algorithms, GA)：二十世纪六十年代初，美国Michigan 大学的J. H. Holland 教授提出，强调染色体的操作。
- 进化策略(Evolution strategies, ES)：1965年，由Rechenberg和Schwefel独立提出，强调个体级的行为变化。
- 进化规划(Evolutionary programming, EP)：由Fogel在20世纪60年代提出，强调种群级上的行为变化。
- 遗传编程 (Genetic Programming，GP)：1992年，Stanford 大学的J.R.Koza提出，采用层式编码结构（树形结构）。
- 基因表达式编程(Gene Expression Programming , GEP)：2001年, 葡萄牙人Candida Ferreira 提出。

第6章 智能计算及其应用

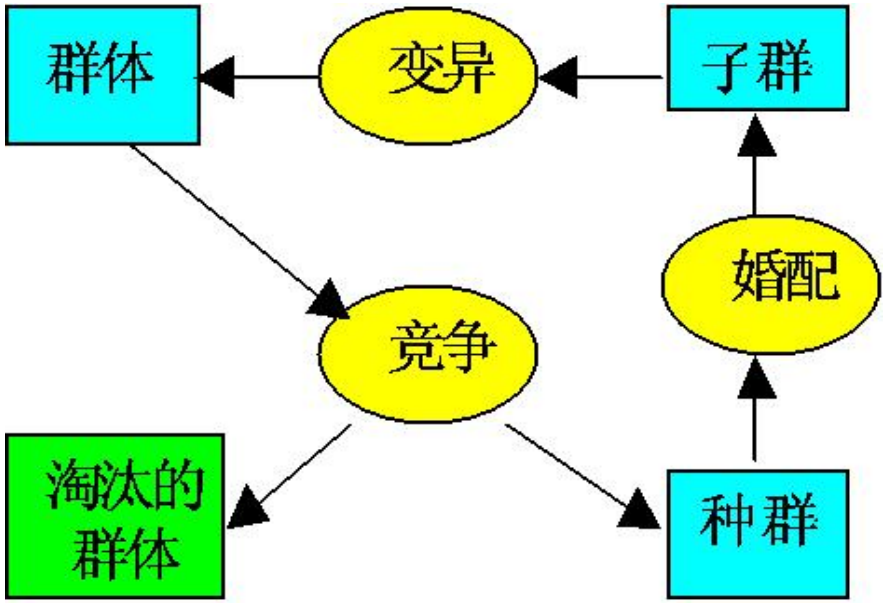
- 6.1 进化算法的产生与发展
- 6.2 基本遗传算法
- 6.3 遗传算法的改进算法
- 6.4 遗传算法的应用
- 6.5 群智能算法产生的背景
- 6.6 粒子群优化算法及其应用
- 6.7 蚁群算法及其应用

- 进化算法是一个“算法簇”，包括遗传算法(GA)、遗传规划、进化策略和进化规划等。
- 进化算法的基本框架是遗传算法所描述的框架。

6.2 基本遗传算法

□ 遗传算法 (Genetic Algorithms, GA):一类借鉴生物界自然选择和基因遗传学原理的启发式随机搜索算法。

- 自然选择：生物在自然界的生存环境中适者生存，不适者被淘汰的过程。
- 遗传：子代总是与亲代（父代）相似。



6.2 基本遗传算法

□ 遗传算法 (Genetic Algorithms, GA):一类借鉴生物界自然选择和基因遗传学原理的启发式随机搜索算法。

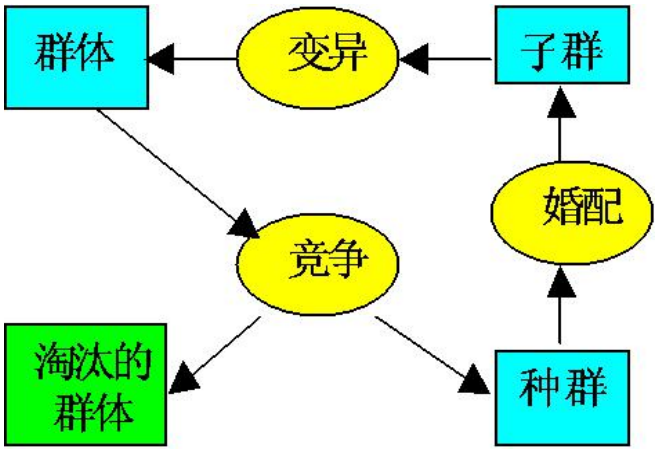
- 自然选择：生物在自然界的生存环境中适者生存，不适者被淘汰的过程。
- 遗传：子代总是与亲代（父代）相似。

- 染色体(chromosome)：生物的遗传物质的主要载体;
- 基因(gene)：扩展生物性状的遗传物质的功能单元和结构单位;
- 基因座(locus)：染色体中基因的位置;
- 等位基因(alleles)：基因所取的值。

6.2 基本遗传算法

□ 遗传算法 (Genetic Algorithms, GA):一类借鉴生物界自然选择和基因遗传学原理的启发式随机搜索算法。

- 实现：在计算机上模拟生物的进化过程和基因的操作（繁殖、变异、重组）。
- 研究目的：
- 抽象和严谨地解释自然界的适应过程；
- 将自然生物系统的重要机理运用到人工系统的设计中。



- Matlab的GA工具箱gatool
- 谢菲尔德大学的Matlab遗传算法库Gatbx以及其升级版GEATbx
- python-GA的库：deap、gaft、Geatpy等

6.2 基本遗传算法

- 6.2.1 遗传算法的产生与发展
- 6.2.2 遗传算法的基本操作
- 6.2.3 遗传算法的设计与实现

- 重点：
  - 选择操作、交叉操作、变异操作
  - 遗传算法的基本流程
- 难点：
  - 不同的编码方式特点，如何确定适应度函数，不同的交叉操作、变异操作、选择操作之间的区别

6.2.1 遗传算法的产生与发展

❑遗传算法的发展概况

(John Holland, 1929.2.2－)

- 起源（20世纪40年代）：从生物学的角度进行生物的进化过程模拟、遗传过程等研究工作。
- 产生（20世纪60年代）：

• 1962年，Fraser提出自然遗传算法。

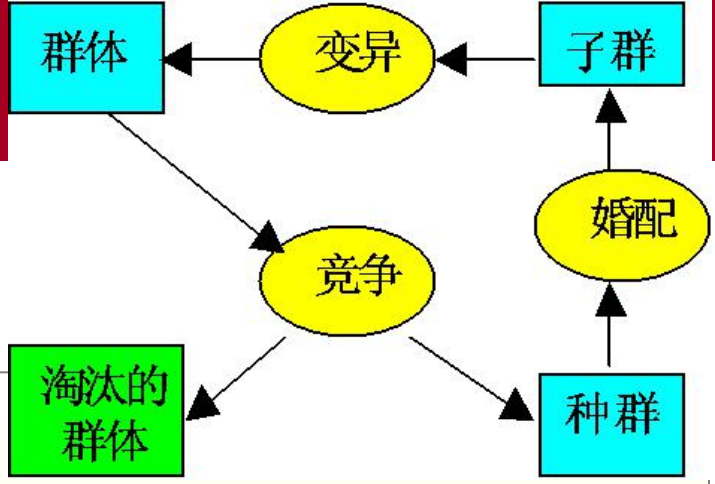
• 1965年，Holland首次提出人工遗传操作的重要性。

• 1967年，Bagley首次提出遗传算法这一术语，并应用于优化游戏程序中的参数设定。

• 1970年，Cavicchio把遗传算法应用于模式识别中。

• 1971年，Hollstien在论文《计算机控制系统中人工遗传自适应方法》中阐述了遗传算法用于数字反馈控制的方法。

6.2.1 遗传算法的产生与发展



❑遗传算法的发展概况

- 产生（20世纪60年代）：

- 1975年，Holland出版《Adaption in Natural and Artificial System》，阐述了遗传算法的基本理论和方法，提出模式理论(Schemata Theorem)及隐并行性(Implicit Parallelism)，用以解释遗传算法的运行机理，标志着遗传算法的正式诞生；

- 1975年，DeJong 在博士论文《An analysis of the behavior of a class of genetic adaptive system》中进行了大量纯数值函数优化计算实验，建立De Jong 五函数测试平台，定义了评价遗传算法收敛和动态性能的标准。

6.2.1 遗传算法的产生与发展

❑遗传算法的发展概况

- 发展 (20世纪80年代后) :

• 1989年，Goldberg出版《Genetic Algorithms in Search，Optimization and Machine Learning》，奠定了现代遗传算法的科学基础。

• 1991年，Davis出版《Handbook of Genetic Algorithms》，包含了遗传算法在科学计算、工程技术和社会经济中的大量应用实例，为推广和普及遗传算法的应用起到了重要的指导作用。

• 1992年，Koza将遗传算法应用于计算机程序的优化设计及自动生成，提出遗传编程的概念。

• Evolutionary Computation (MIT Press，1993创刊，DeJong主编)和IEEE Transactions on Evolutionary Computation (IEEE汇刊，1997年创刊，DavidB .Fo gel主编)

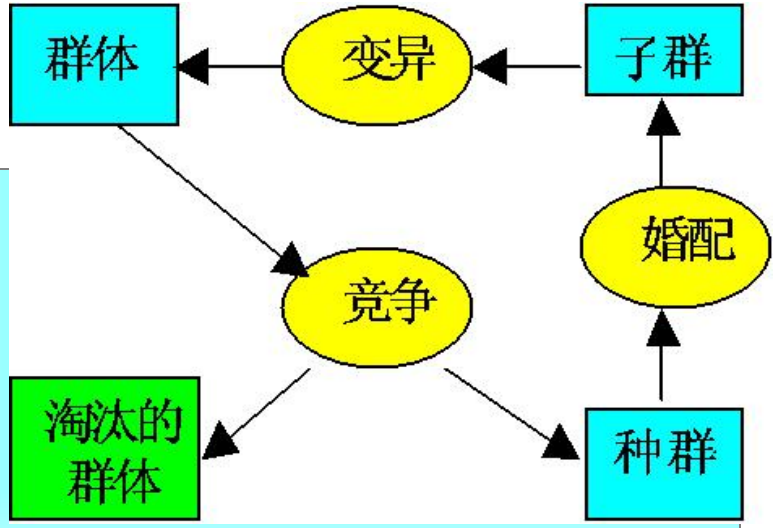
6.2 基本遗传算法

- 6.2.1 遗传算法的产生与发展
- 6.2.2 遗传算法的基本操作
- 6.2.3 遗传算法的设计与实现

- 重点：
  - 选择操作、交叉操作、变异操作
  - 遗传算法的基本流程
- 难点：
  - 不同的编码方式特点，如何确定适应度函数，不同的交叉操作、变异操作、选择操作之间的区别

6.2.2遗传算法的基本操作

1．遗传算法的生物遗传学基础



- 1）遗传——基础
- 2）变异——条件
- 3）选择——方向

- 选择通过遗传和变异起作用，变异为选择提供资料，遗传巩固与积累选择的资料；
- 选择控制变异与遗传的方向，使变异和遗传向着适应环境方向发展。

问题：自然系统的进化机制与GA的启发随机搜索机制对应？

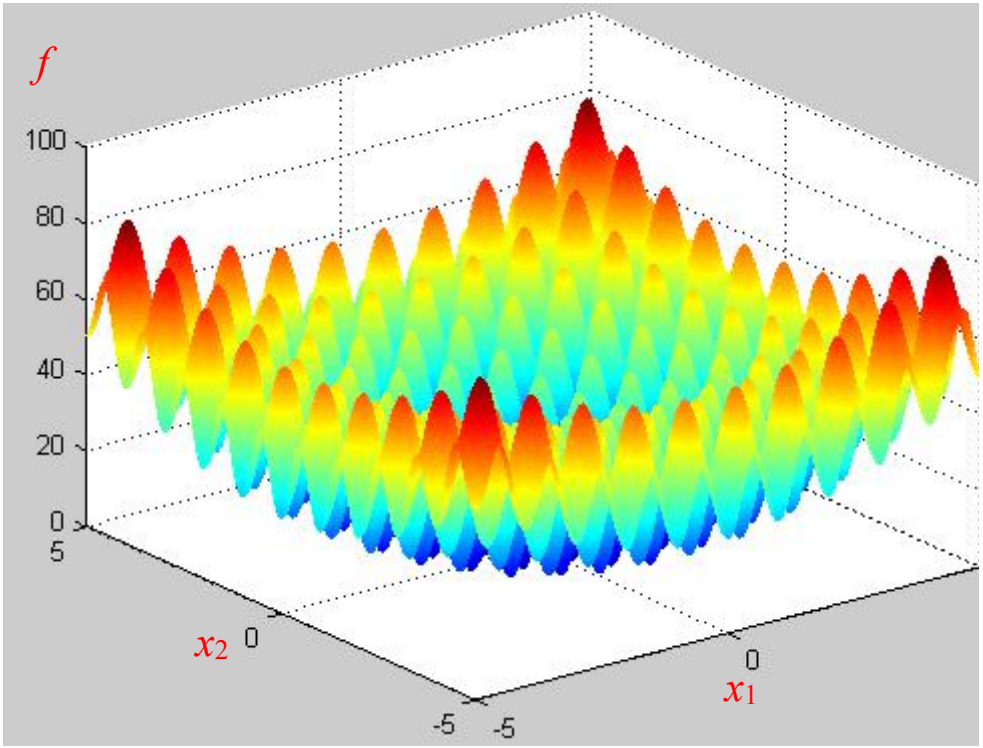
6.2.2 遗传算法的基本操作

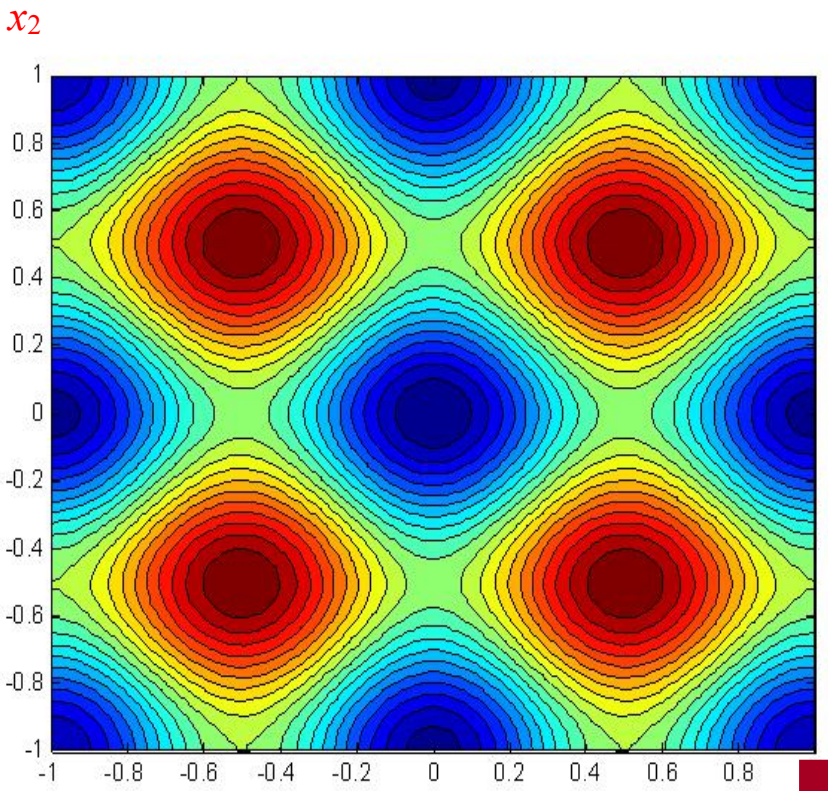
生物遗传概念	遗传算法中的应用
环境	问题
个体（Individual）/ 染色体（Chromosome）	问题的一个解/ 解的编码（字符串、向量等）
基因（Gene）	编码的元素
群体（Population）	被选定的一组解（解的个数为群体的规模）
交叉（Crossover）	选择两个染色体进行交叉产生一组新的染色体的过程
变异（Mutation）	编码的某些分量发生变化的过程
适应性（Fitness）	适应度函数（对应问题的目标函数）值
选择（Selection）/ 适者生存	适应度函数值大的解被保留的概率大

□ 例：用遗传算法求解下面一个 **Rastrigin** 函数的最小值。

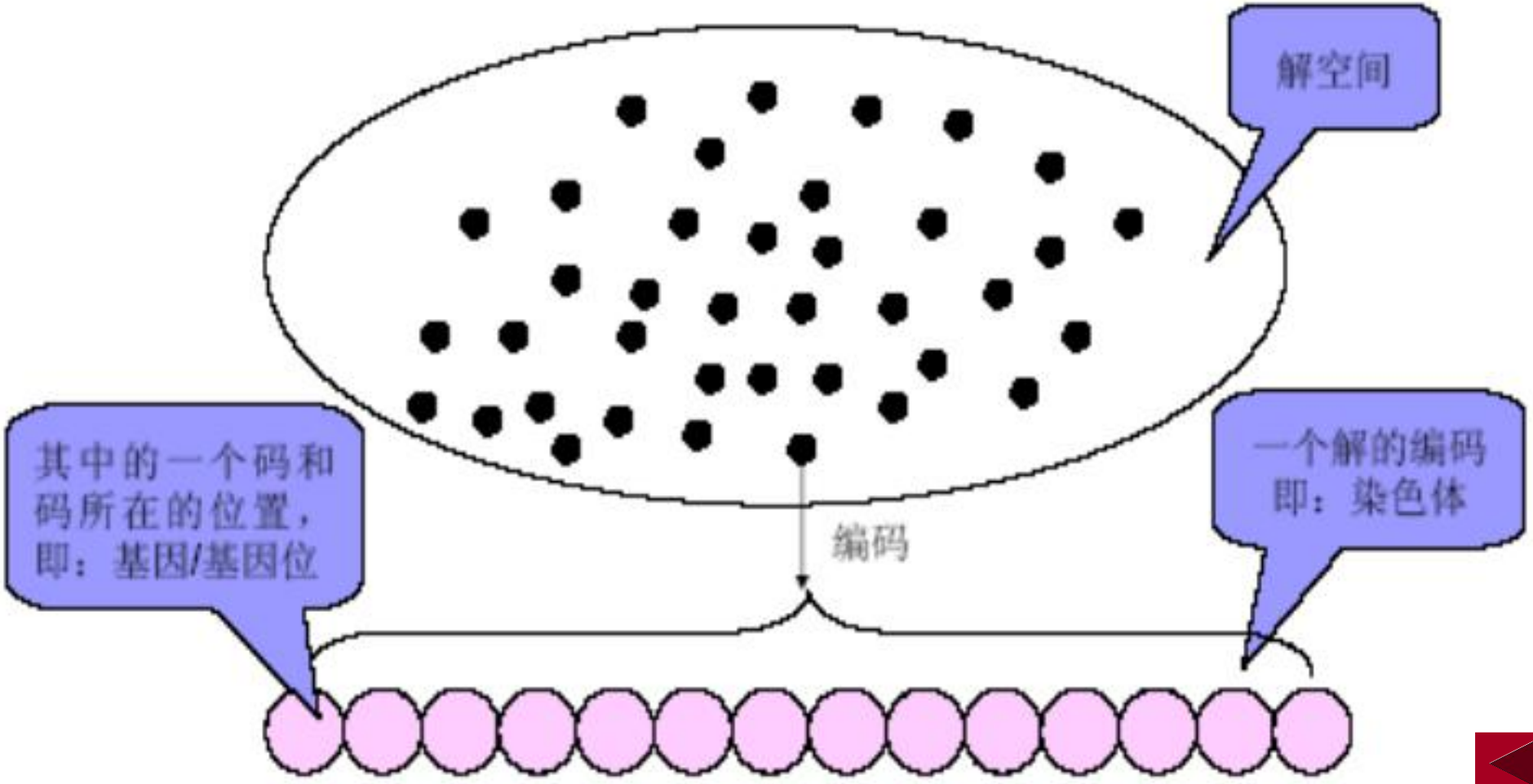
$$f(x_1, x_2) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$$

$$-5 \le x_i \le 5 \qquad i = 1, 2$$





解的编码－GA的基础

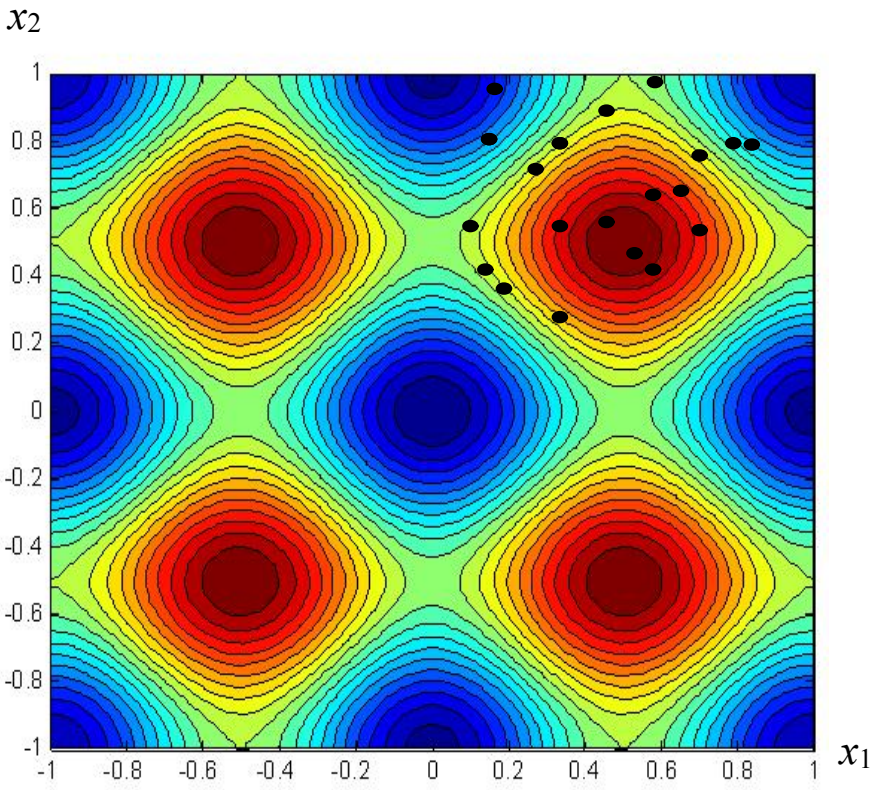


- 例： 用遗传算法求解下面一个**Rastrigin**函数的最小值。

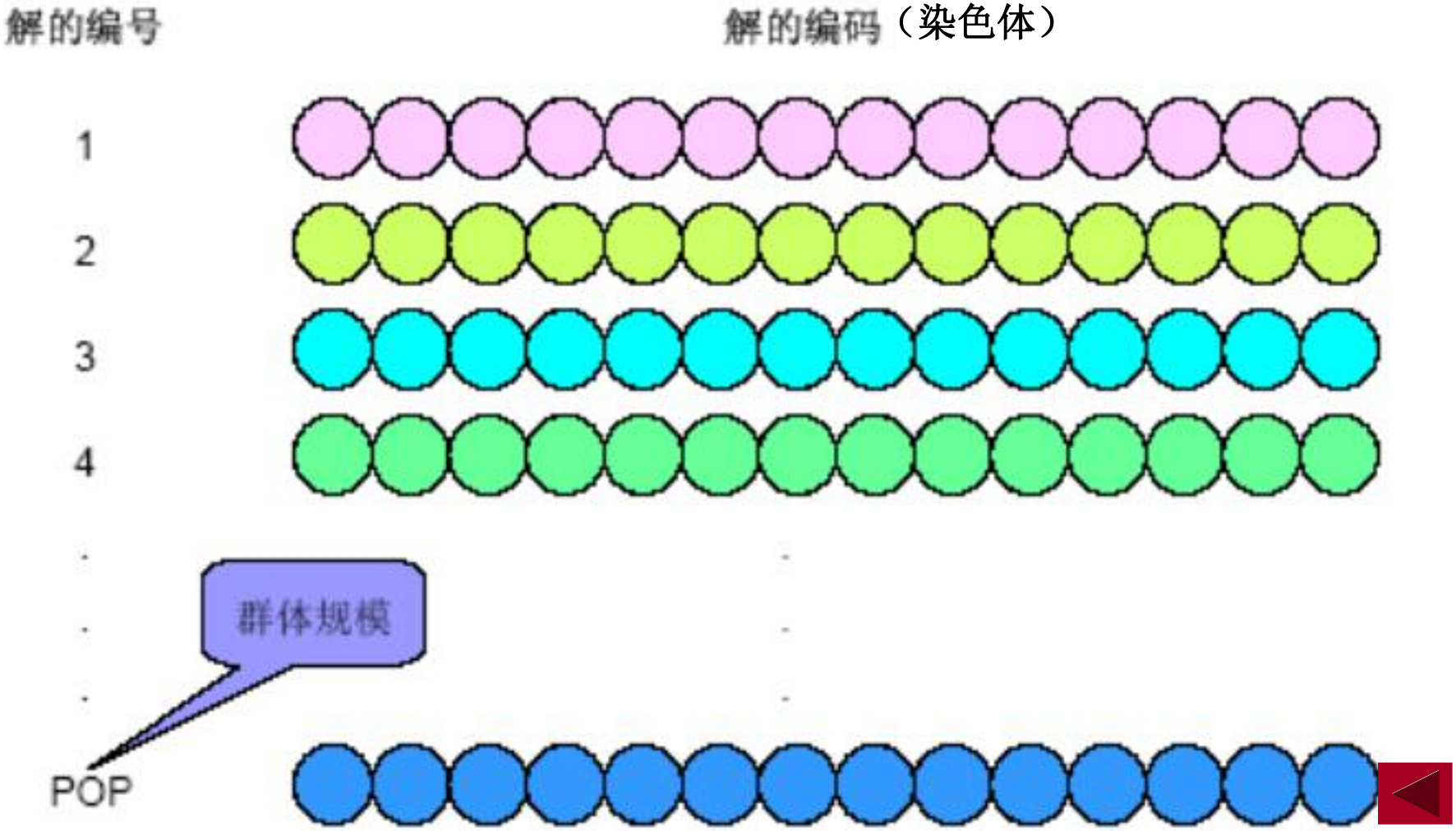
$$f(x_1, x_2) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$$

$$-5 \le x_i \le 5 \qquad i = 1, 2$$

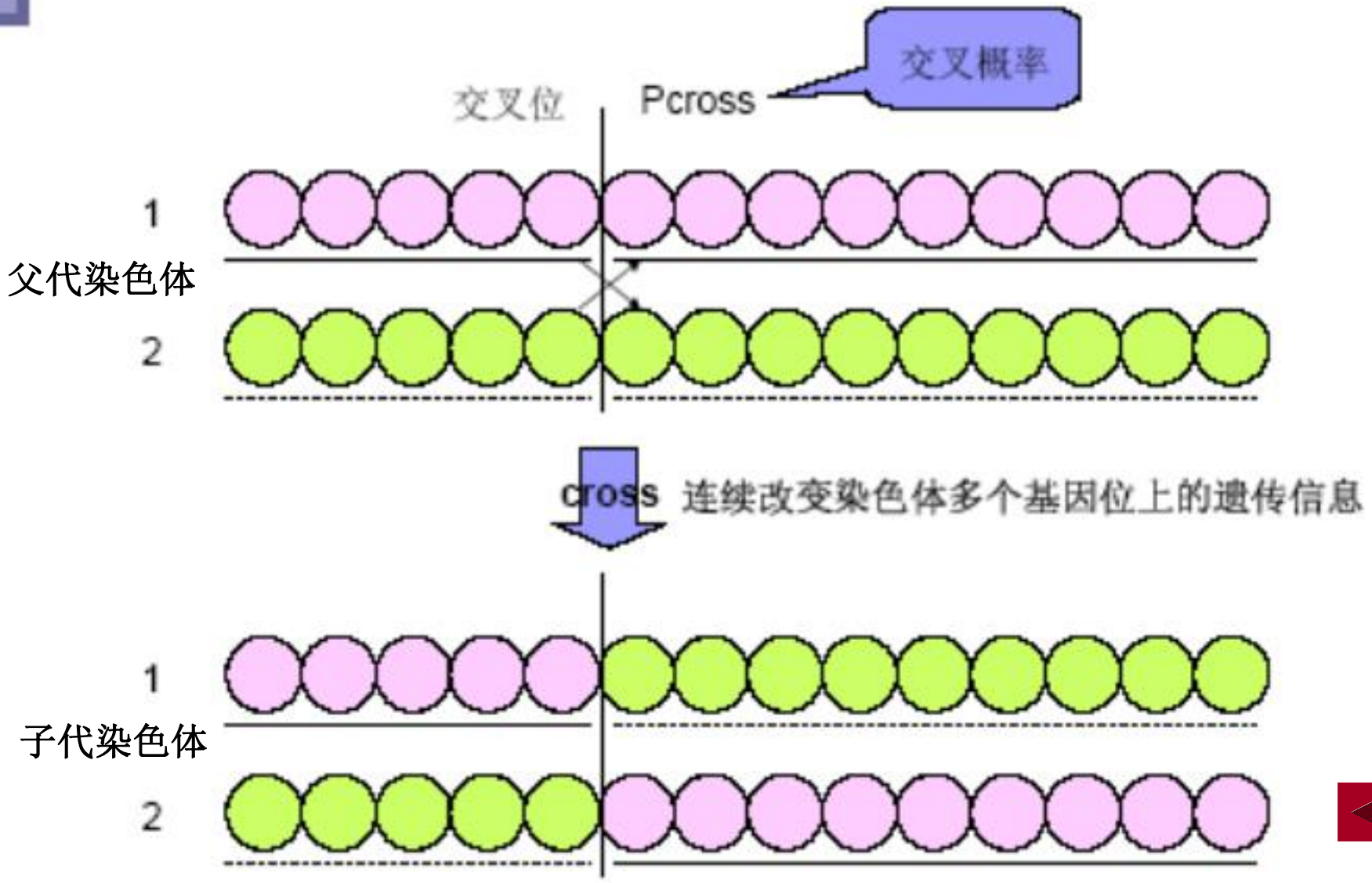
- 初始种群：随机生成**20**个个体



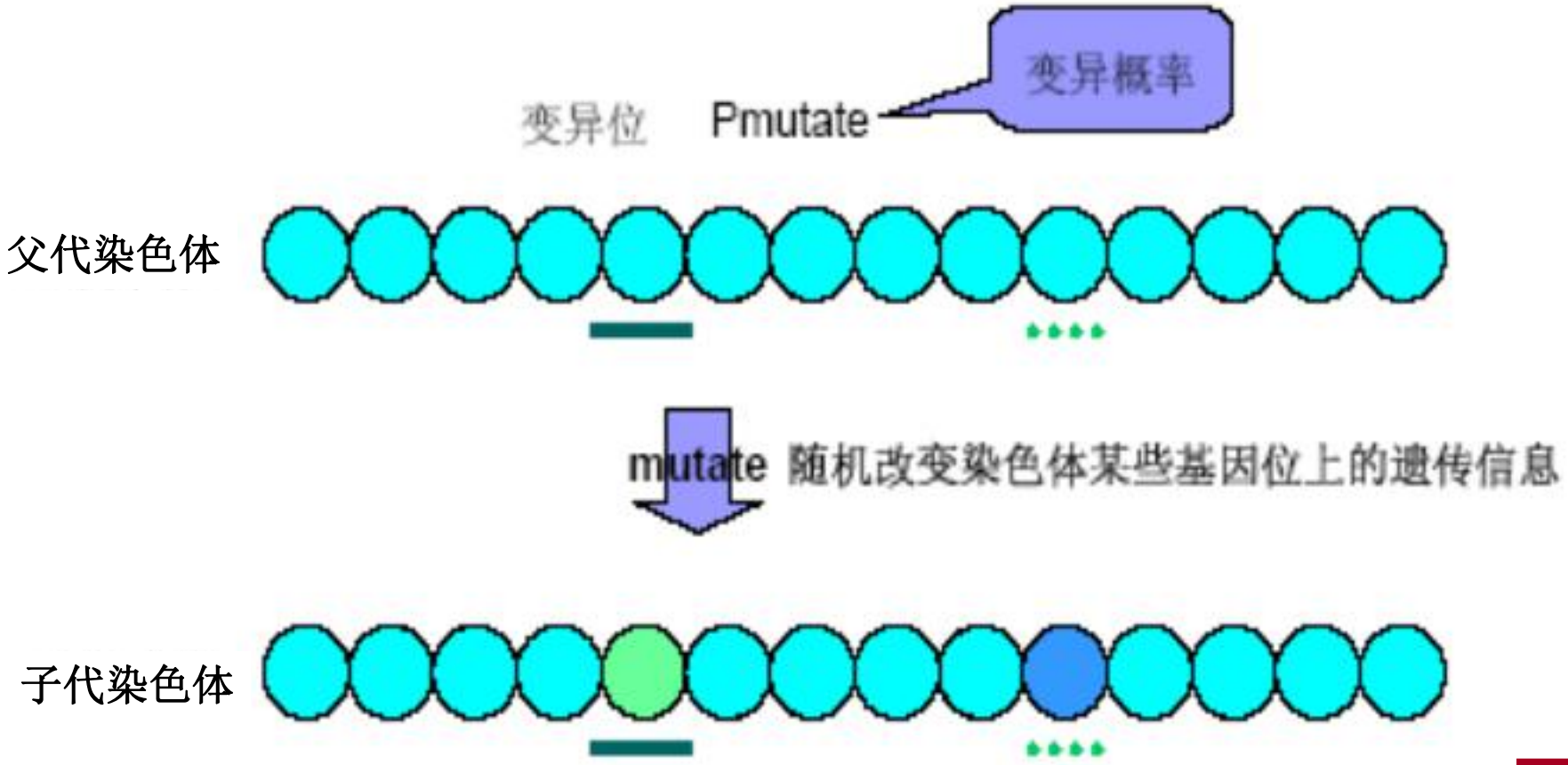
初始化——Initialization



交叉操作— Crossover Operator

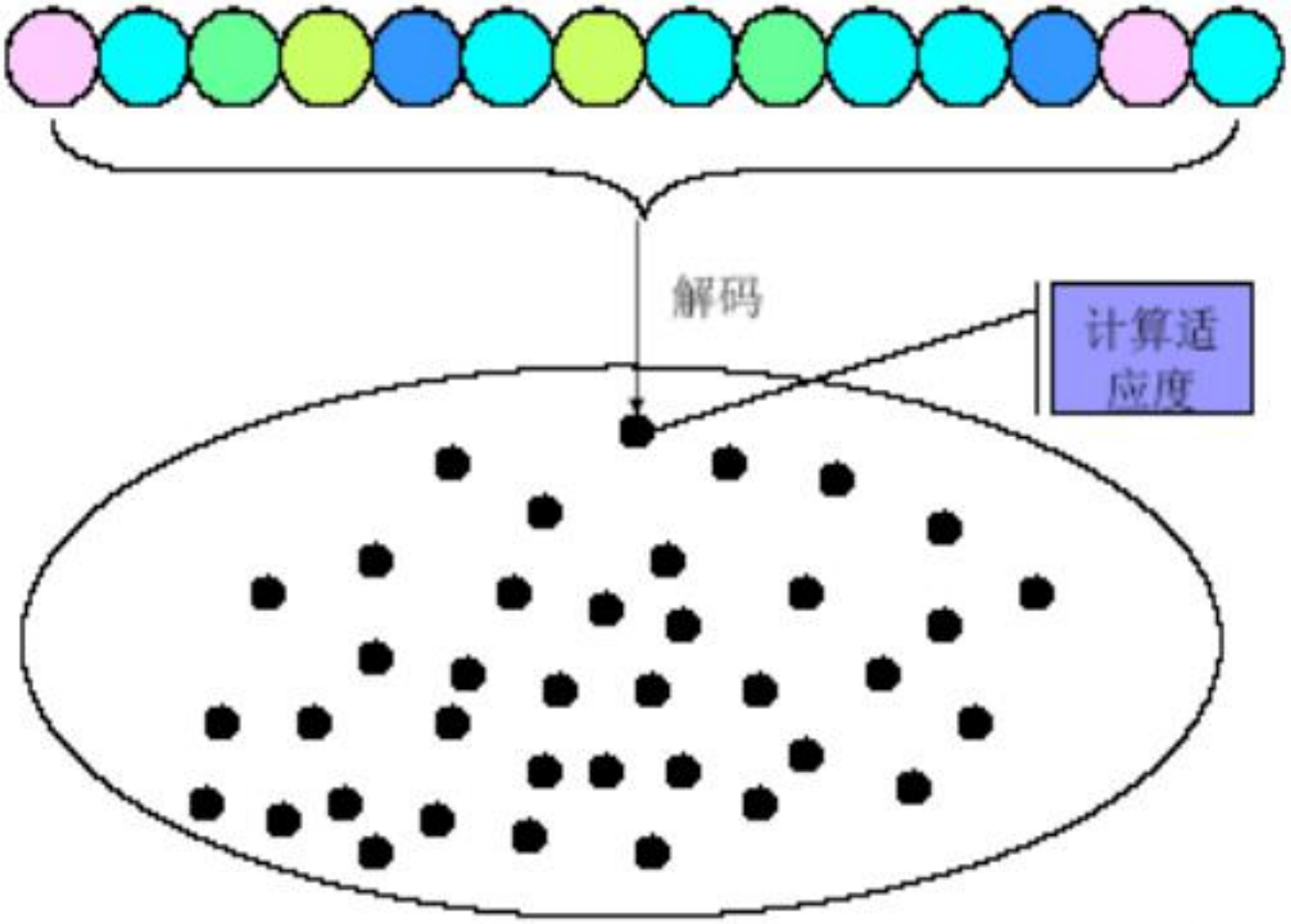


变异操作— Mutate Operator

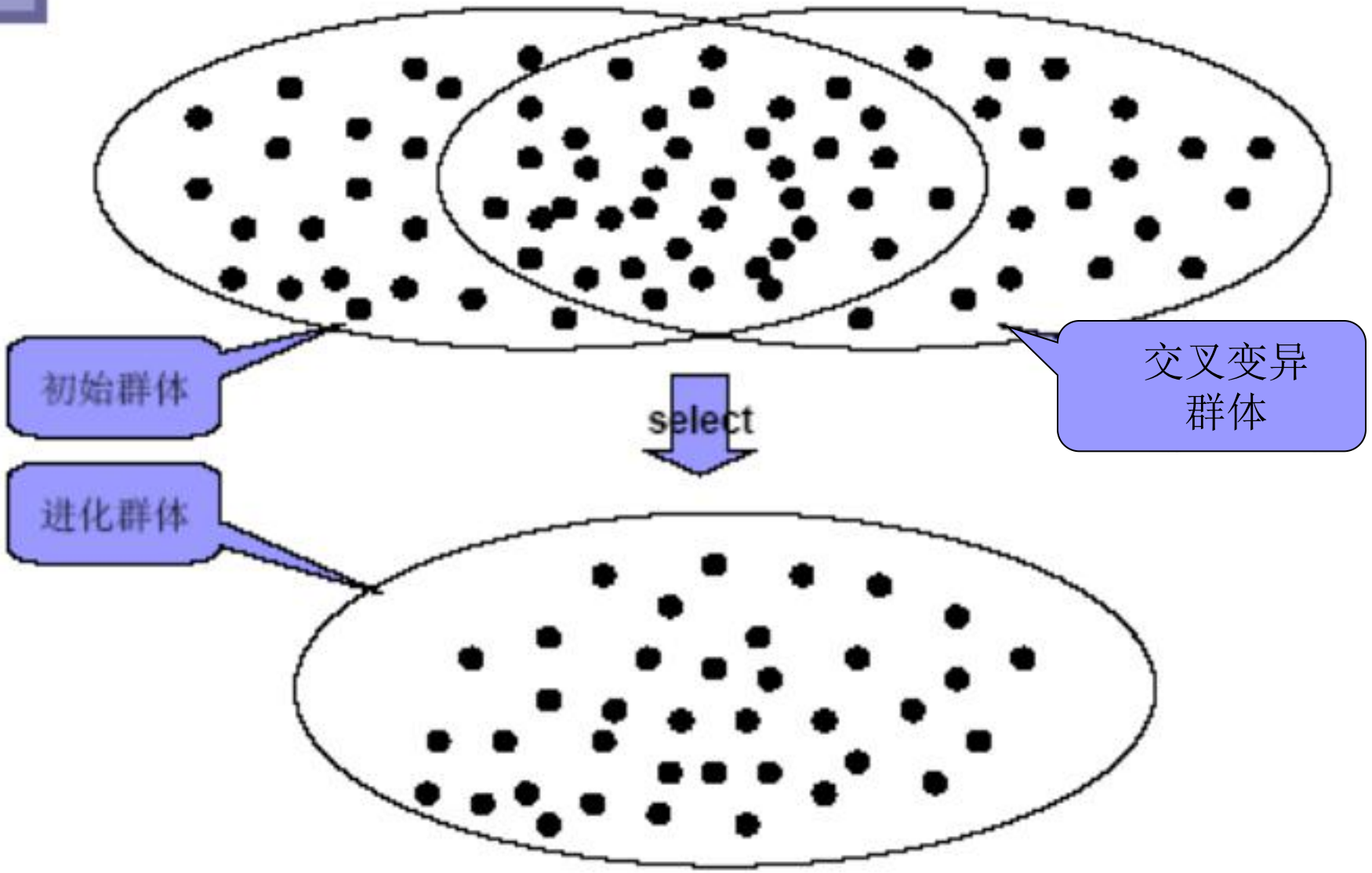


适应度计算

染色体



选择操作— Selection Operator

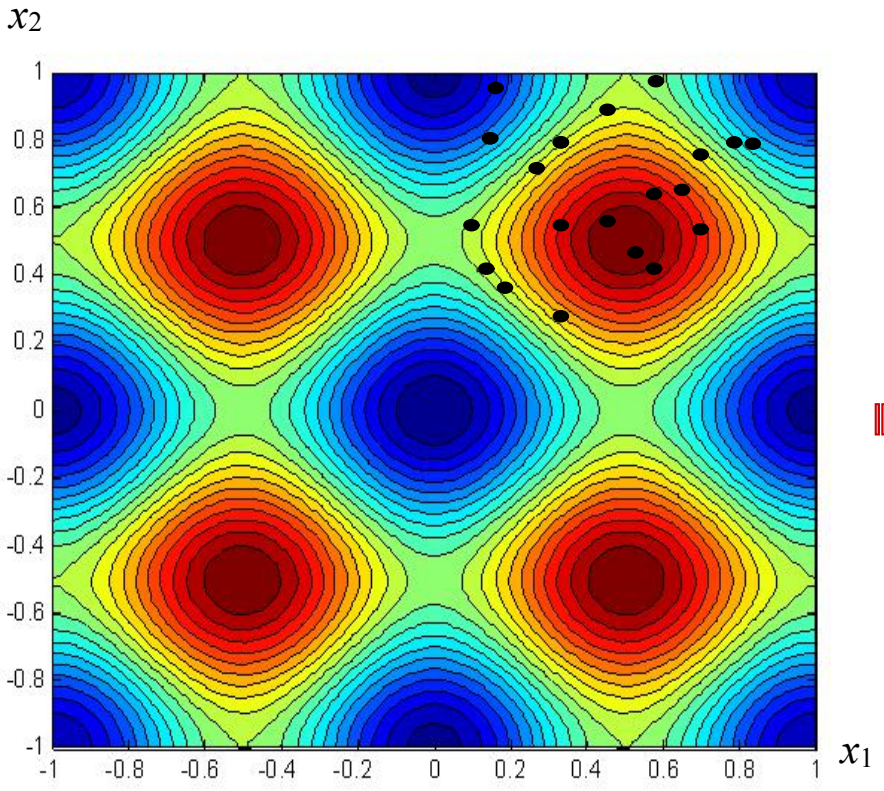


例： 用遗传算法求解下面一个**Rastrigin**函数的最小值。

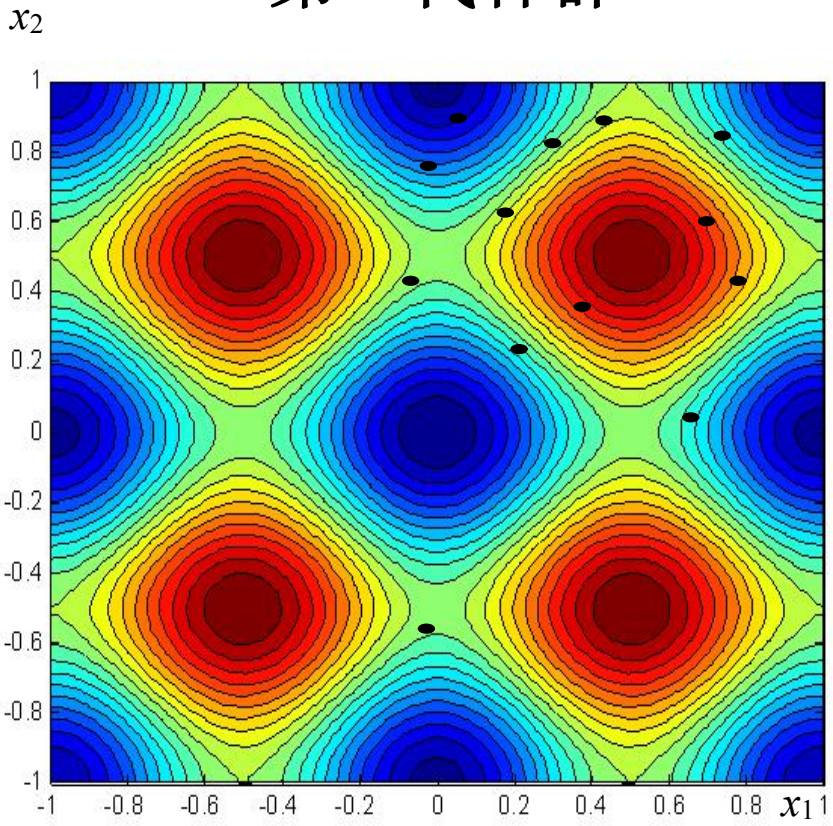
$$f(x_1, x_2) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$$

$$-5 \le x_i \le 5 \qquad i = 1, 2$$

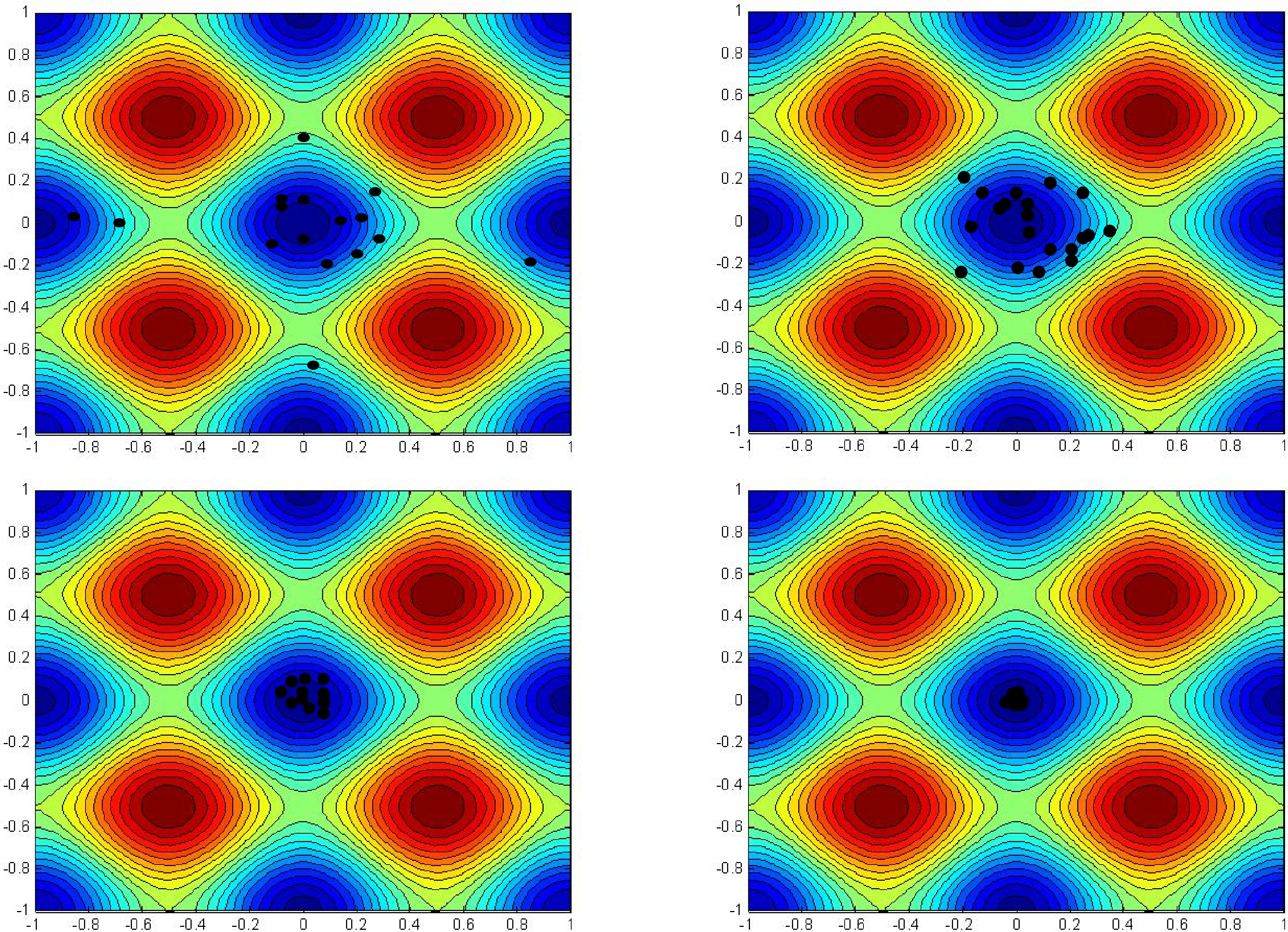
初始种群



第二代种群



在迭代60、80、95、100次时的种群



用随机遗传教机器人站起来

6.2 基本遗传算法

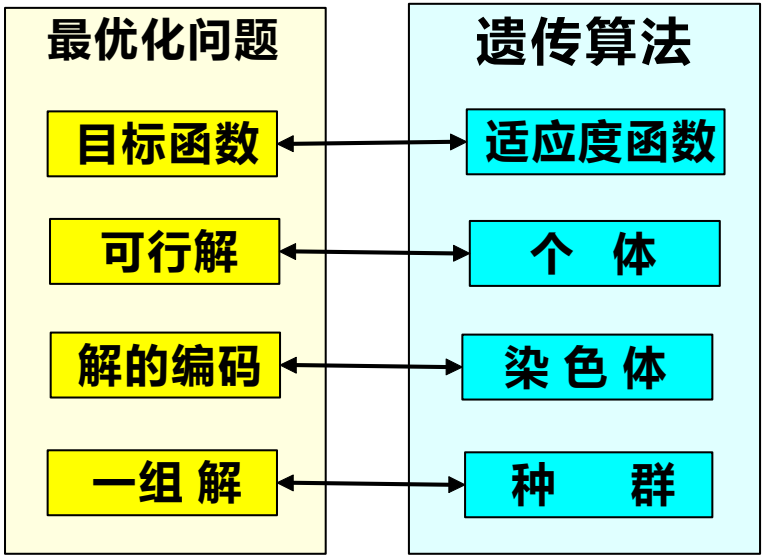
- 6.2.1 遗传算法的产生与发展
- 6.2.2 遗传算法的基本操作
- 6.2.3 遗传算法的设计与实现

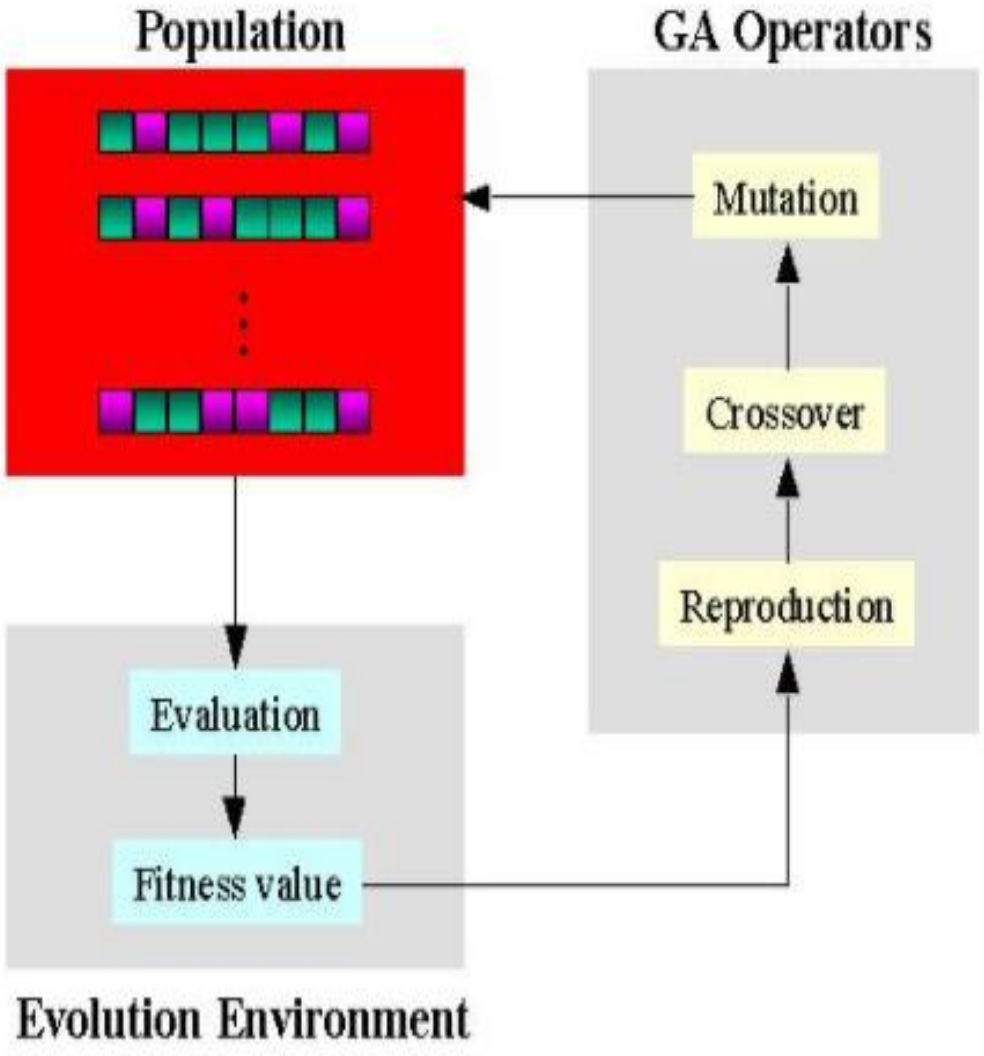
遗传算法 (Genetic Algorithms, GA):一类借鉴生物界自然选择和基因遗传学原理的启发式随机搜索算法。

- 重点：
  - 选择操作、交叉操作、变异操作
  - 遗传算法的基本流程
- 难点：
  - 不同的编码方式特点，如何设计适应度函数，不同的交叉操作、变异操作、选择操作之间的区别

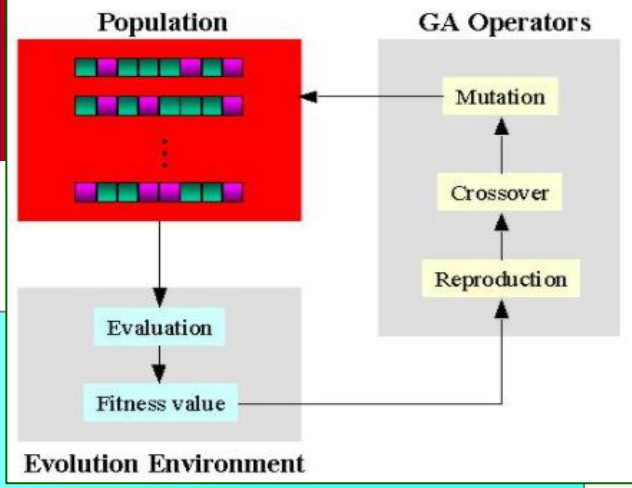
6.2.2 遗传算法的基本操作

- 2．遗传算法的基本流程（p145）





6.2.2 遗传算法的基本操作

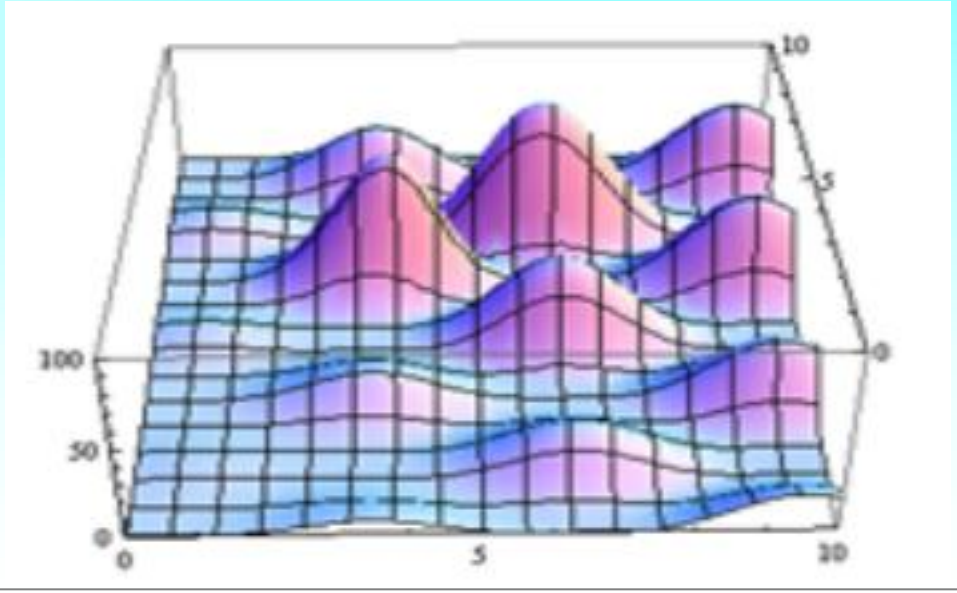


3．遗传算法的基本操作

- 问题：求下列函数的最大值。

$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))^2}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

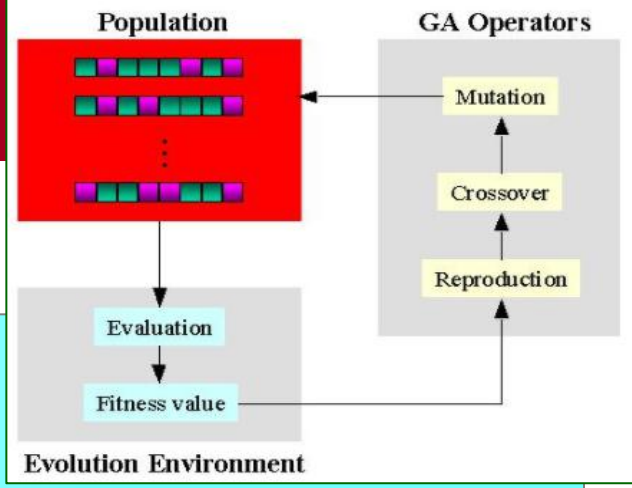
$$x\in[0,10),y\in[0,10)$$



(1) 实数编码

- 假设染色体长度n=8，
  等位基因由{0,1,…,9}组成。
- 例如染色体为70298267，
  则x=7.029，y=8.267

6.2.2 遗传算法的基本操作



□ 3．遗传算法的基本操作

- 问题：求下列函数的最大值。

$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))^2}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

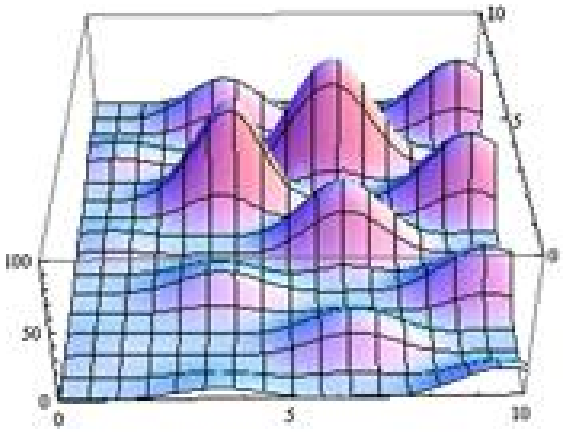
$$x\in[0,10),y\in[0,10)$$

- (2) 初始种群的产生（设种群大小*N*=10，染色体长度*n*=8 ）

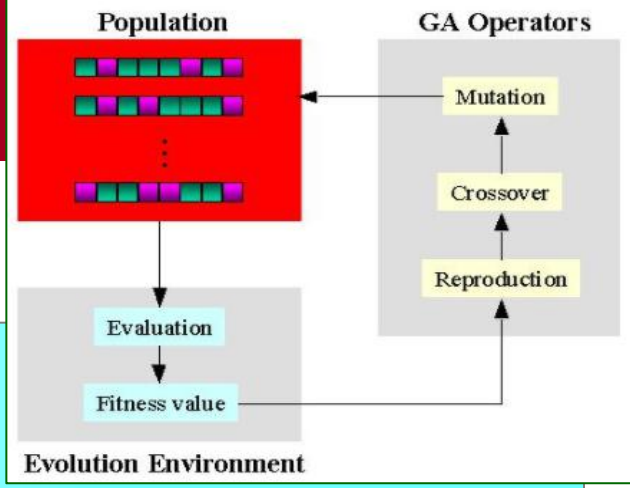
- 用随机数发生器产生：产生0～9之间均匀分布的80个随机数。

初始种群

35507844	60159798
64898050	06730709
62777010	50928956
48048252	28284207
44777682	08226907



6.2.2 遗传算法的基本操作



3．遗传算法的基本操作

- 问题：求下列函数的最大值。

$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))^2}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

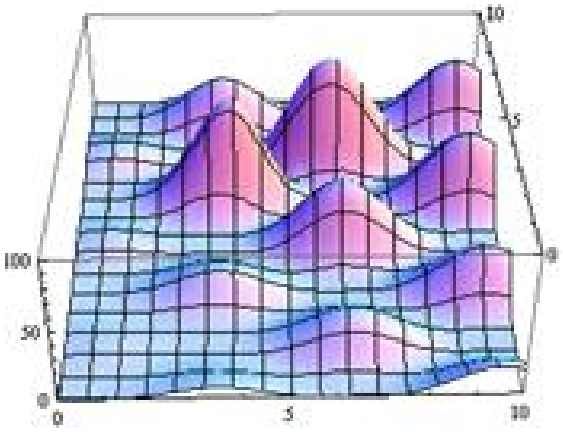
$$x\in[0,10),y\in[0,10)$$

- (3) 计算适应度（要求适应度函数：非负、求最大化）

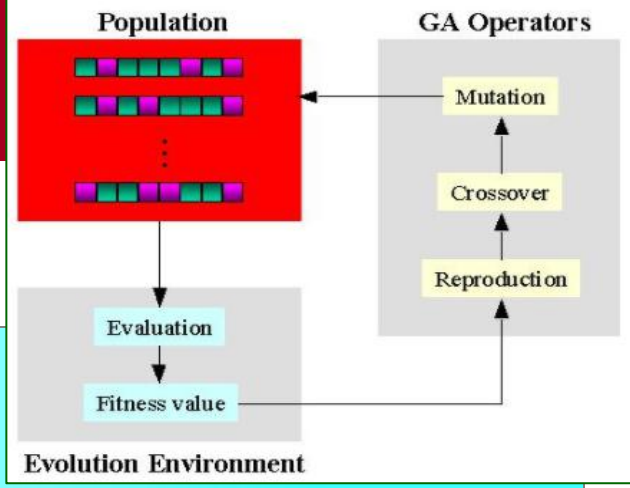
  这里适应度函数为 $f(x,y)$ 。

初始种群

染色体	适应度	排名	染色体	适应度	排名
35507844	25.426	4	60159798	32.165	6
64898050	86.963	10	06730709	02.490	1
62777010	38.530	9	50928956	29.537	5
48048252	37.982	8	28284207	14.465	2
44777682	37.871	7	08226907	22.692	3



6.2.2 遗传算法的基本操作



□ 3．遗传算法的基本操作

- 问题：求下列函数的最大值。

$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))^2}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

$$x\in[0,10),y\in[0,10)$$

- **(4) 选择（复制，reproduction）**

- 选择操作：从当前群体中按照一定概率选择优良的个体，使它们有机会作为父代繁殖下一代。

- ① 个体选择概率分配(适应度比例方法或蒙特卡罗法)

- 个体 *i* 被选择的概率：

$$p_i=\frac{f_i}{\sum_{i=1}^{N}f_i}$$

6.2.2 遗传算法的基本操作

□ 3．遗传算法的基本操作

- (4) 选择（复制，reproduction）
- ① 个体选择概率分配(适应度比例方法或蒙特卡罗法)
  - 个体 i 被选择的概率：

$$p_i = \frac{f_i}{\sum_{i=1}^{N} f_i}$$

$$\sum_{i=1}^{N} p_i = \frac{\sum_{i=1}^{N} f_i}{\sum_{i=1}^{N} f_i} = 1$$

	染色体	适应度	选择概率	染色体	适应度	选择概率
初始种群	35507844	25.426	0.077	60159798	32.165	0.098
	64898050	86.963	0.265	06730709	02.490	0.008
	62777010	38.530	0.117	50928956	29.537	0.090
	48048252	37.982	0.116	28284207	14.465	0.044
	44777682	37.871	0.115	08226907	22.692	0.069

6.2.2 遗传算法

初始种群

染色体	适应度	选择概率	染色体	适应度	选择概率
35507844	25.426	0.077	60159798	32.165	0.098
64898050	86.963	0.265	06730709	02.490	0.008
62777010	38.530	0.117	50928956	29.537	0.090
48048252	37.982	0.116	28284207	14.465	0.044
44777682	37.871	0.115	08226907	22.692	0.069

3．遗传算法的

- ② 选择个体

- **轮盘赌选择（Roulette Wheel Selection）**

  - 计算每个染色体的累计概率

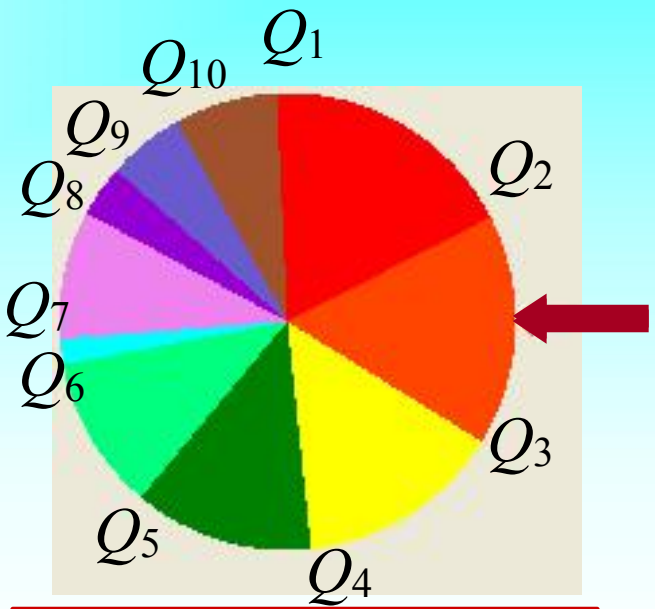
$Q_1 = p_1 = 0.077$

$Q_2 = Q_1 + p_2 = 0.077 + 0.265 = 0.332$

$Q_3 = Q_2 + p_3 = 0.332 + 0.117 = 0.449$

…

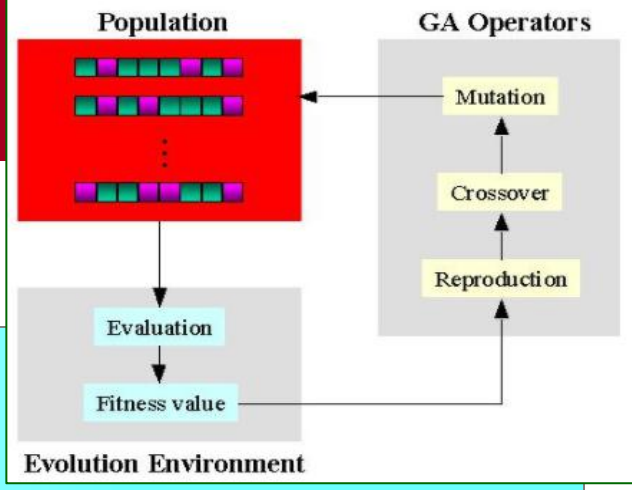
$Q_{10} = Q_9 + p_{10} = 1$



如果落在Q_4和Q_5中间
A：第4个个体
B：第5个个体

6.2.2 遗传算法的基本操作

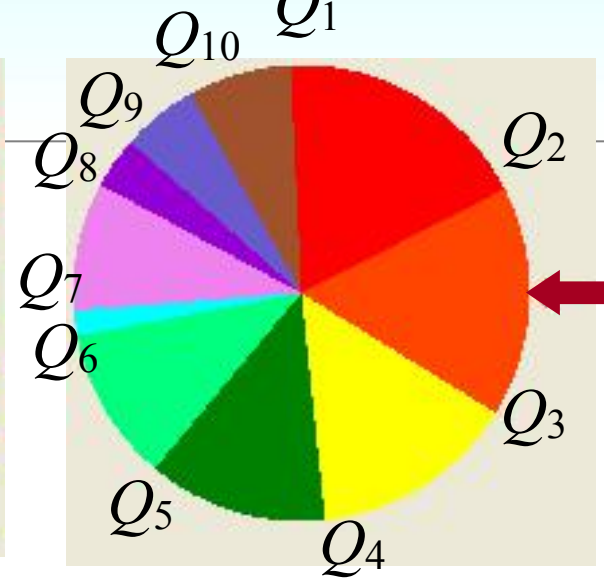
□ 3．遗传算法的基本操作



- **(4)选择（复制，reproduction）**

- **轮盘赌选择（Roulette Wheel Selection）**

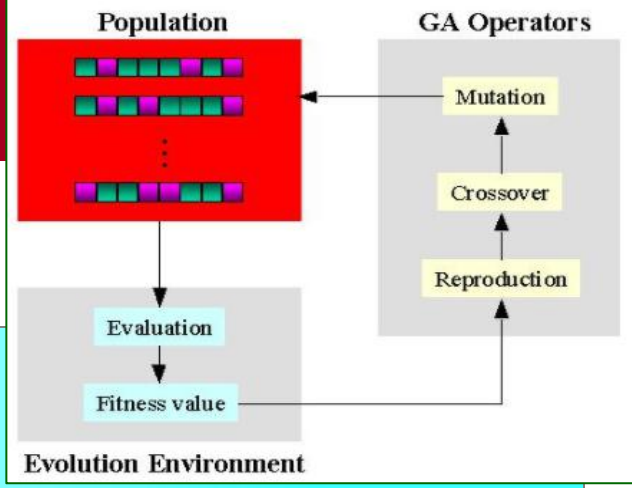
  - 产生一个随机数 $r \in [0,1]$；
  - 若 $Q_{k-1} < r \le Q_k$，则选择复制 第k个染色体 。
  - 复制前：



  - 复制后：

6.2.2 遗传算法的基本操作

□ 3．遗传算法的基本操作

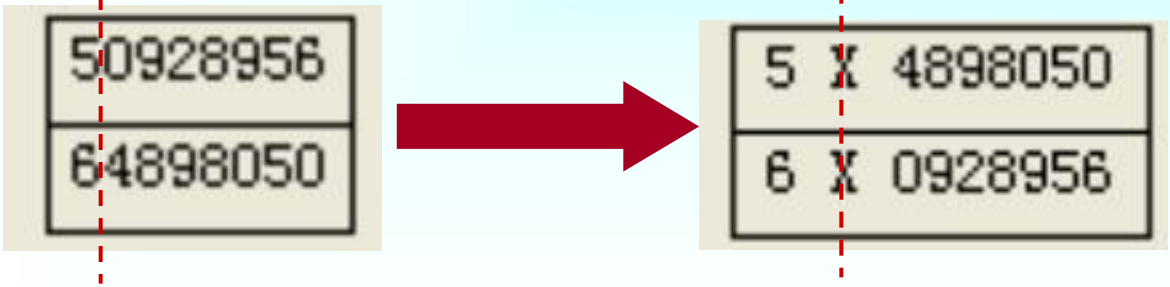


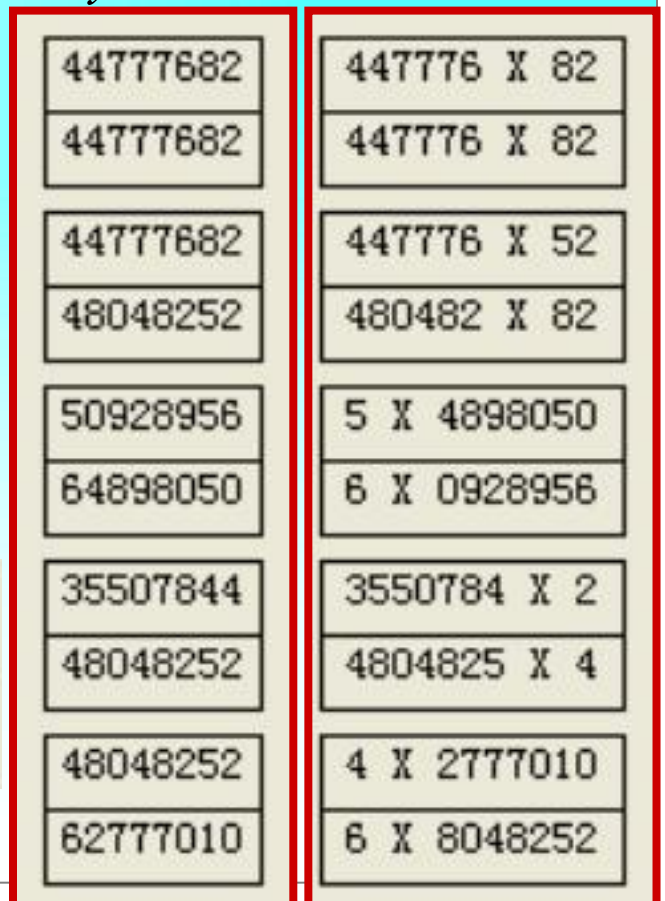
- 问题：求下列函数的最大值。

$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))^2}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

$$x\in[0,10),y\in[0,10)$$

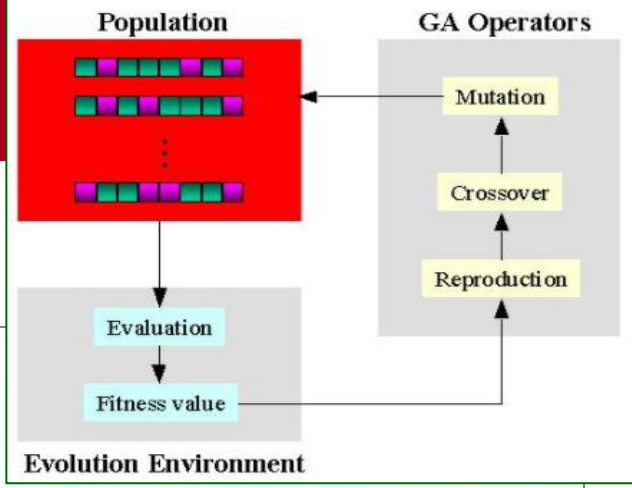
- **(5) 交叉（基因重组 recombination）**
- 先两两配对，后按交叉概率进行交叉
- 单点交叉(假设交叉概率Pc＝0.85)：





6.2.2 遗传算法的基本操作

□ 3．遗传算法的基本操作



- 问题：求下列函数的最大值。

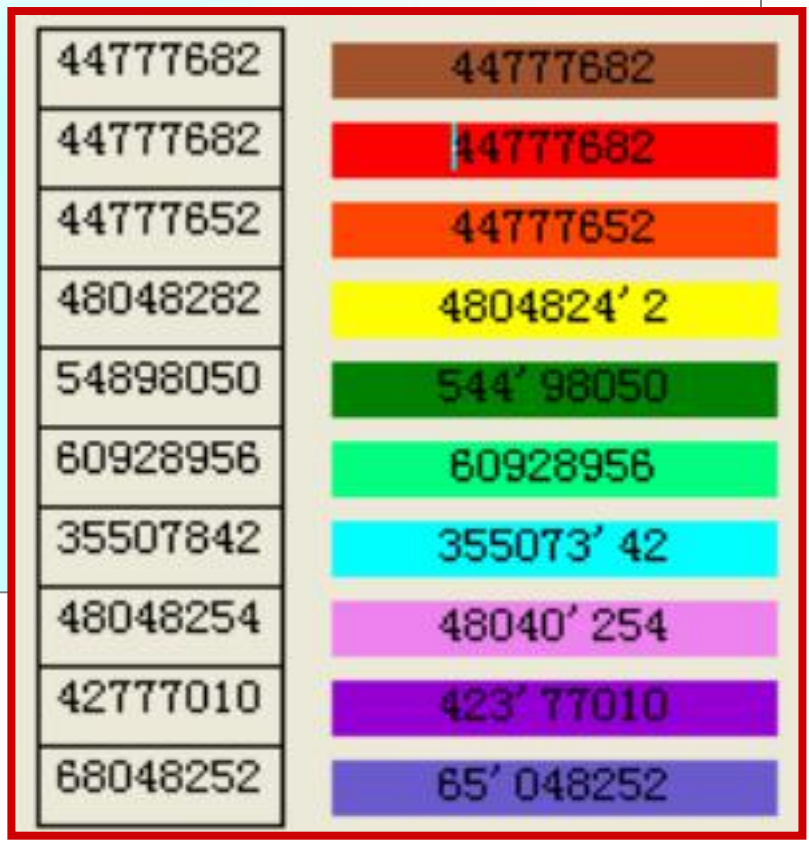
$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))^2}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

$$x\in[0,10),y\in[0,10)$$

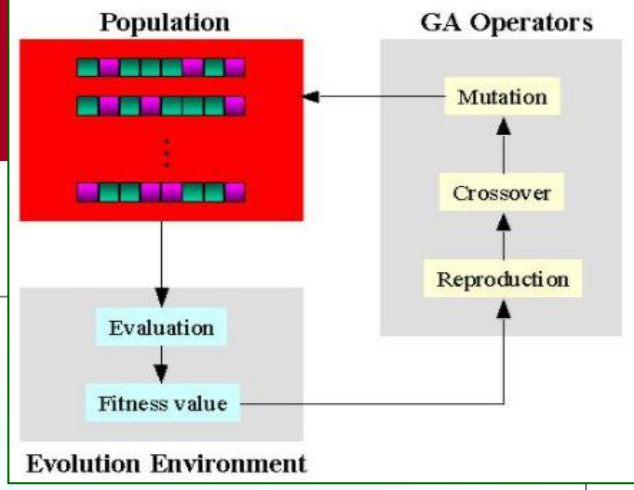
- **(6) 变异(mutation)**

- 变异在遗传算法中的作用是第二位，但是必不可少的。

- 设变异概率 p_m=0.05, 采用位点变异.



6.2.2 遗传算法的基本操作



3．遗传算法的基本操作

- 问题：求下列函数的最大值。

$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))^2}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

$$x\in[0,10),y\in[0,10)$$

	第一代	第二代
适应度总和	328.121	394.524
平均适应度	32.812	39.4524
最大适应度	86.963	75.618

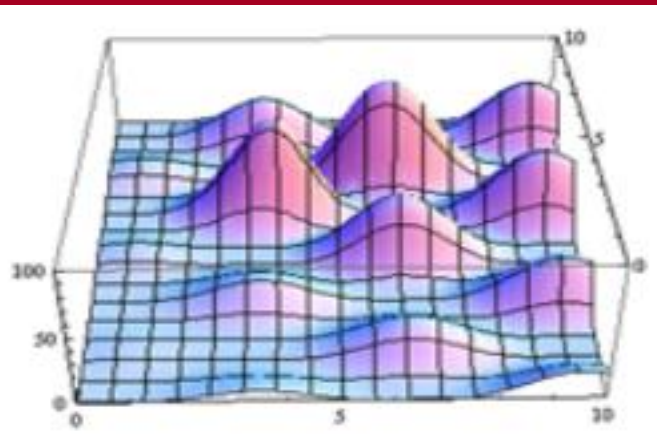
第一代

染色体	适应度	排名
35507844	25.426	4
64898050	86.963	10
62777010	38.530	9
48048252	37.982	8
44777682	37.871	7
60159798	32.165	6
06730709	02.490	1
50928956	29.537	5
28284207	14.465	2
08226907	22.692	3

第二代

染色体	适应度	排名
44777682	37.871	6
44777682	37.871	7
44777652	37.250	5
48048242	38.427	8
54498050	75.618	10
60928956	30.933	3
35507342	27.465	2
48040254	02.808	1
42377010	34.780	4
65048252	71.501	9

6.2.2 遗传算法的基本操作

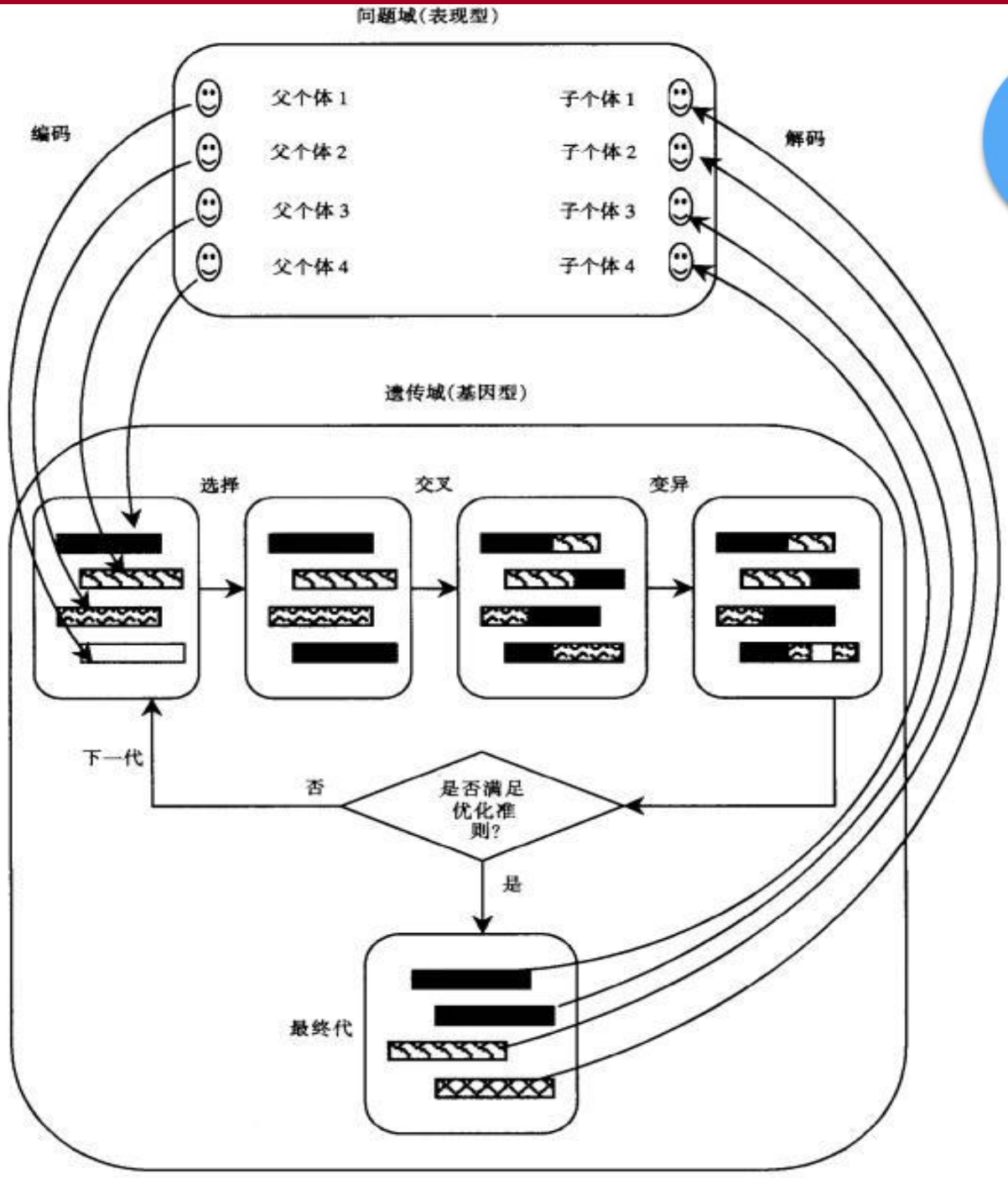


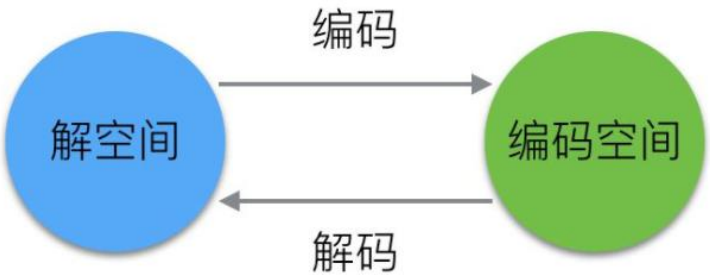
3．遗传算法的基本操作

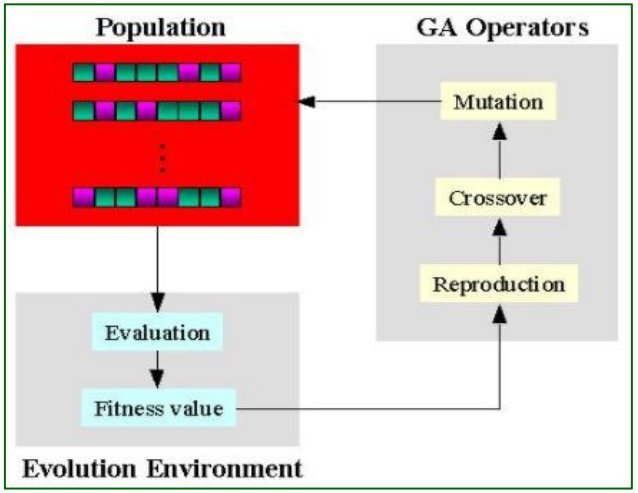
- 问题：求下列函数的最大值。

$$f(x,y)=\frac{6.452(x+0.125y)(\cos(x)-\cos(2y))}{\sqrt{0.8+(x-4.2)^2+2(y-7)^2}}+3.226y$$

$$x\in[0,10),y\in[0,10)$$

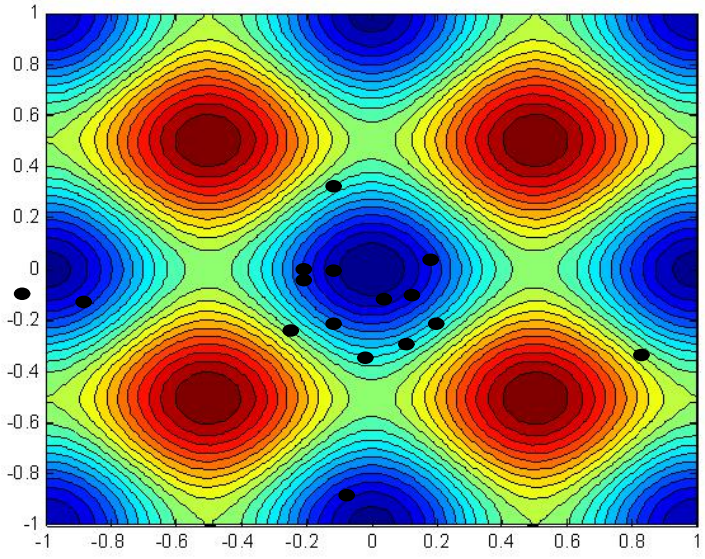






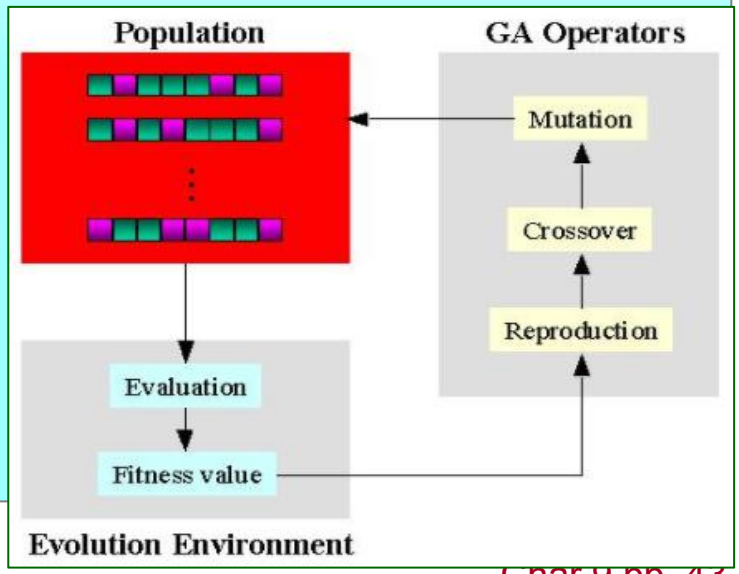
染色体：个体的基因型
解：个体的表现型

6.2.2 遗传算法的基本操作



❑ 特点

- 只对参数集的编码进行操作，而非对参数本身进行操作；
- 从许多初始点开始并行搜索，搜索的全局性强，不易陷入局部最优；
- 通过目标函数计算适应度，对问题的依赖性小；
- 寻优规则是由概率决定的，而非确定性的；
- 在解空间进行高效启发式搜索，非盲目地穷举或完全随机搜索；
- 具有并行计算的特点，适合于大规模复杂问题的优化；
- 计算简单，功能强大。



6.2 基本遗传算法

- 6.2.1 遗传算法的产生与发展
- 6.2.2 遗传算法的基本操作
- 6.2.3 遗传算法的设计与实现

■ 重点：

➢ 选择操作、交叉操作、变异操作

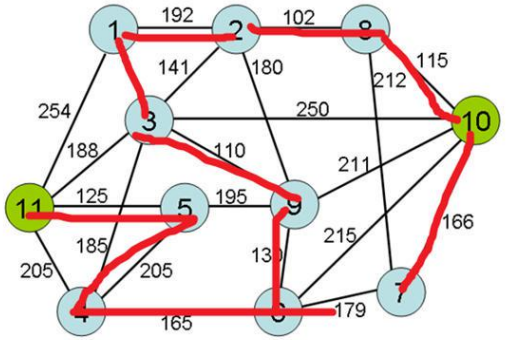
➢ 遗传算法的基本流程

■ 难点：

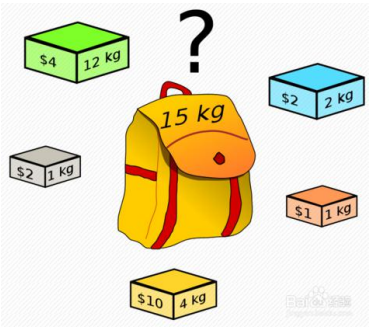
➢ 不同的编码方式特点，如何设计适应度函数，不同的交叉操作、变异操作、选择操作之间的区别

$$\text{minimize/maximize}_x \quad f(x)$$
$$\text{subject to} \quad x \in \Omega$$

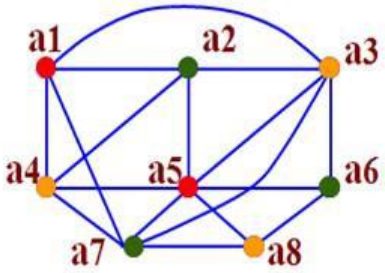
- x: Decision
- $f(\cdot)$: Objective
- Ω: Constraints



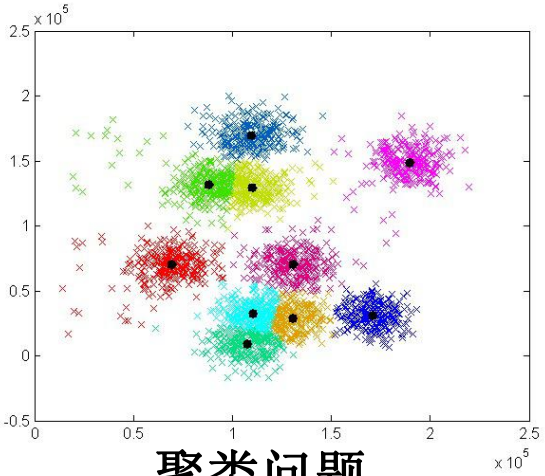
旅行商问题



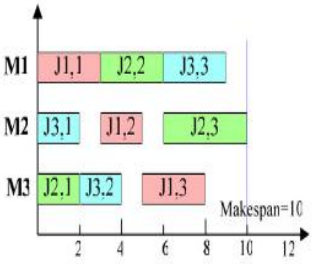
0-1背包问题



图着色问题



聚类问题

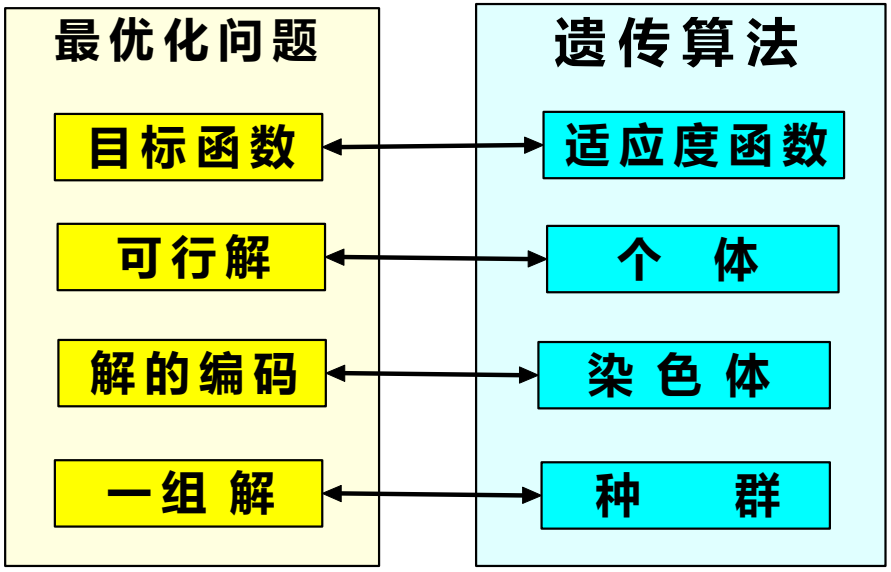


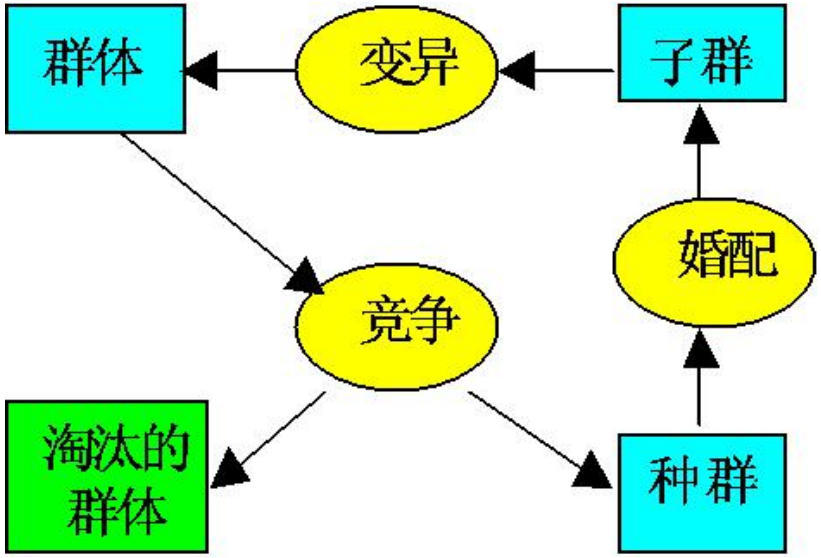
	工序1	工序2	工序3
J1：	M1	M2	M3
J2：	M3	M1	M2
J3：	M2	M3	M1

作业调度问题

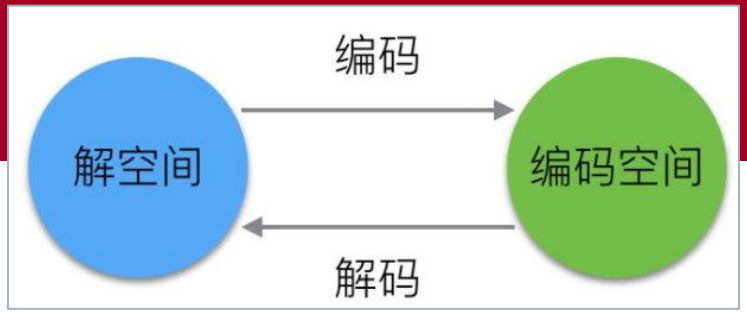
6.2.3 遗传算法的设计与实现

- 解的编码
- 初始种群的设定
- 适应度函数的设计
- 遗传操作（选择、交叉、变异）
- 算法控制参数的设定
- 约束条件的处理





6.2.3 遗传算法的设计与实现



❑ 一、编码

- 参数的编码：把一个问题的可行解从其解空间转换到遗传算法所能处理的搜索空间的转换方法。
- 编码对于遗传算子，尤其是对交叉和变异算子的功能和设计有很大的影响。

问题 的 编 码一般应满足以下三个规则:

- 完备性(completeness): 原问题空间中的所有点(可行解)都能成为编码后的点 ;
- 健全性(soundness): 编码后的空间中的点能对应原问题空间所有的点;
- 非冗余性(non-redundancy): 编码前后空间的点一一对应.

6.2.3 遗传算法的设计与实现

$$\min f(x_1,x_2)=20+x_1^2+x_2^2-10(\cos 2\pi x_1+\cos 2\pi x_2)$$
$$-5\le x_i\le 5 \qquad i=1,2$$

❑ 一、编码

- **1. 位串编码（一维染色体编码方法）**
- **二进制编码（基因用0或1表示，如：00100011010）**
- **二进制串长取决于求解精度**

串长与精度之间的关系：

若要求$x1$的求解精度到小数点后1位，而 $x1$的区间长度为5-（-5）=10，即需将区间分为10/0.1=100等份。

- **$2^6=64 < 100 < 2^7=128$**
- **所以x_1的二进制编码串长为7位，**
- **整条染色体长度呢？ 14位**

0000000: -5
0000001: -4.9
…
1111111:5

A：7位，B：14位

6.2.3 遗传算

$$\min f(x_1,x_2)=20+x_1^2+x_2^2-10(\cos 2\pi x_1+\cos 2\pi x_2)$$
$$-5\le x_i\le 5 \qquad i=1,2$$

❑ 一、编码

- **1. 位串编码（一维染色体编码方法）**
- **二进制编码（基因用0或1表示，如：001000**

编码	x1
0000000:	-5
0000001:	-4.9
…	
1111111:	5

- 根据具体问题确定待寻优的参数： $x_1, x_2, \ldots, x_m$
- 对每个参数 x_i 确定其变化范围[$x_{i\min}, x_{i\max}$] 和编码精度

$$\delta_i=\frac{x_{i\max}-x_{i\min}}{2^{n_i}}$$

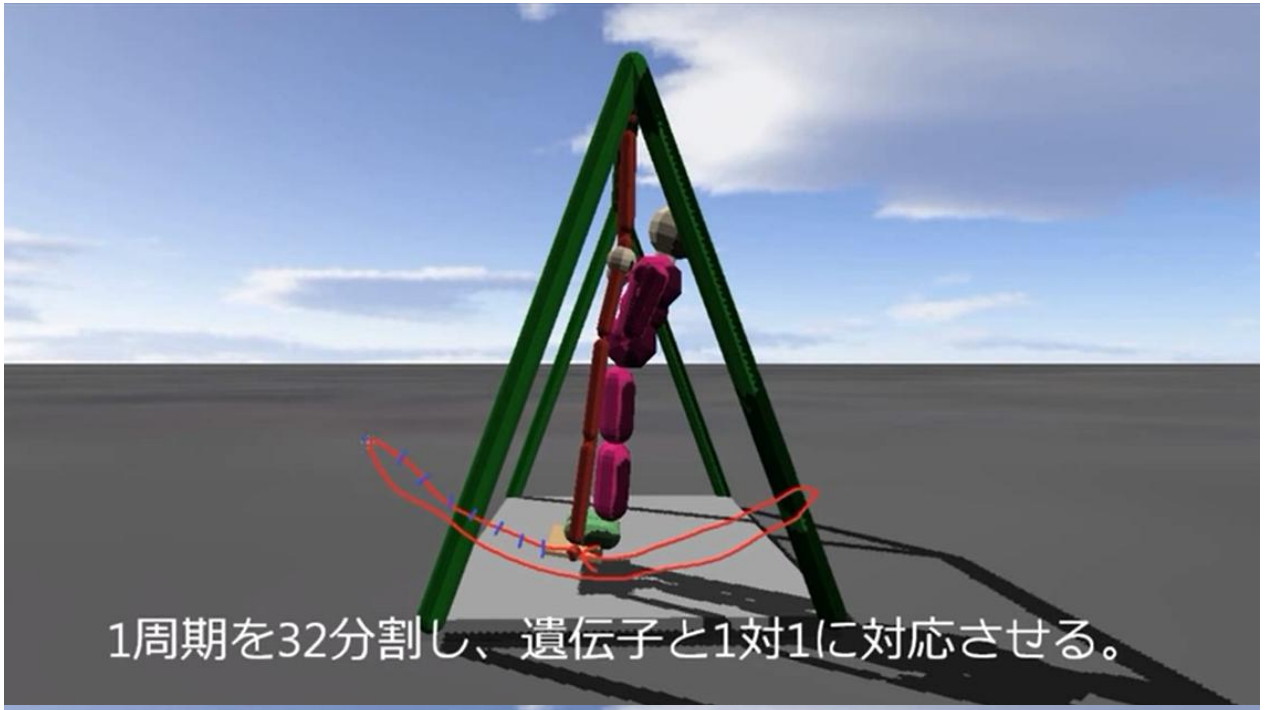
x_1和x_2的染色体：0010001010001

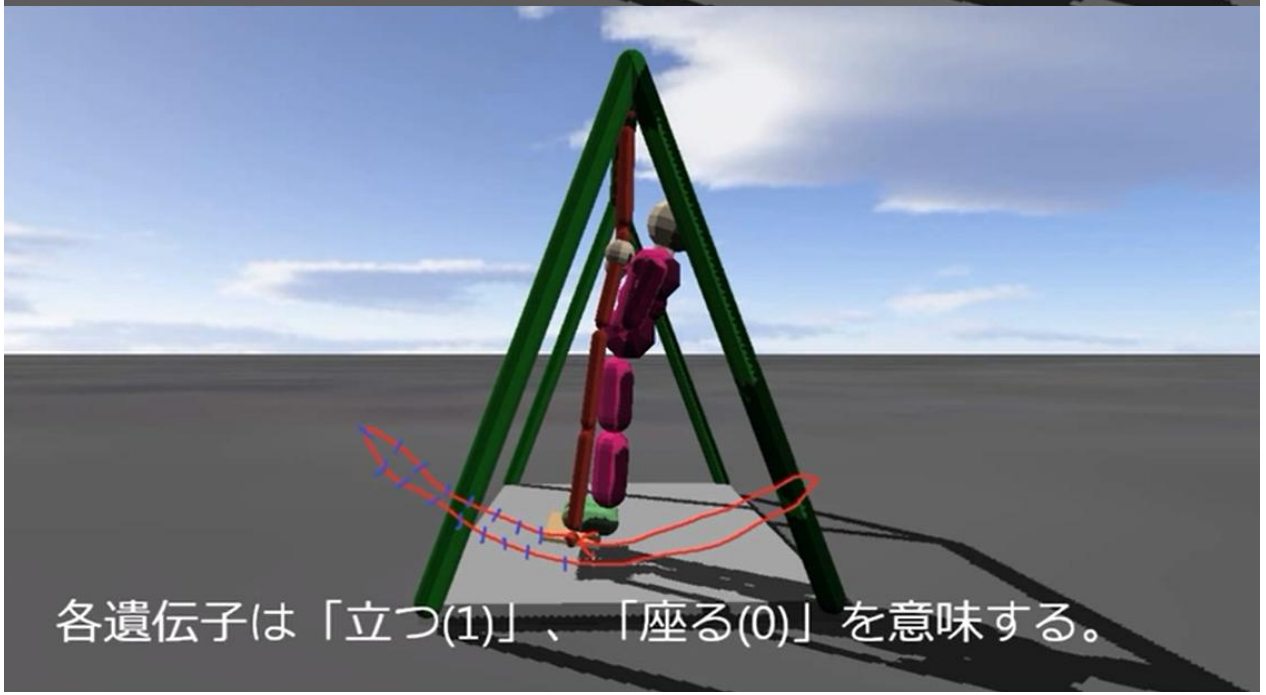
x_1= -5+(dec(0010001)/(2^7-1))×(5-(-5))
=-3.7

- 则对应二进制数为 $A_i=a_{ni-1}\ldots a_1a_0$
- x_i 和A_i的关系： $x_i=x_{i\min}+\frac{\mathrm{dec}(A_i)}{2^{n_i}-1}(x_{i\max}-x_{i\min})$

解码公式，其中dec() 是把二进数转换成十进制数

- 将二进制数串接起来组成一个长的二进制字串$A_1A_2\ldots A_m$，整条染色体长度为$n_1+n_2+\ldots n_m$。


1周期を32分割し、遺伝子と1対1に対応させる。


各遺伝子は「立つ(1)」、「座る(0)」を意味する。

6.2.3 遗传算法的设计与实现

- 1. 位串编码（一维染色体编码方法）
- 二进制编码

- 优点：类似生物染色体结构，易于用生物遗传理论解释，各种遗传操作易实现。

- 缺点：

① 造成Hamming 悬崖，降低遗传算子的搜索效率。

例 7：0111 ，8： 1000 则 Hamming 距离＝4

② 要先给出求解精度，缺乏微调功能。

③ 高维优化问题的二进制编码串长，算法的搜索效率低。

6.2.3 遗传算法的设计与实现

- **1. 位串编码（一维染色体编码方法）**
- **Gray 编码**（格雷编码，二进制编码通过一个变换得到的编码）

二进制串：$<\beta_1\beta_2\cdots\beta_n>$	Gray编码：$<\gamma_1\gamma_2\cdots\gamma_n>$
$\beta_k=\sum_{i=1}^{k}\gamma_i(\mathrm{mod}2)=\begin{cases}\gamma_1 & k=1\\ \beta_{k-1}\oplus\gamma_k & k>1\end{cases}$	$\gamma_k=\begin{cases}\beta_1 & k=1\\ \beta_{k-1}\oplus\beta_k & k>1\end{cases}$ 异或运算
7： 0111 8： 1000 Hamming 距离＝4	7： 0100 8： 1100 Hamming 距离＝1

格雷码编码是其连续的两个整数所对应的编码之间只有一个码位是不同的，其余码位完全相同。

6.2.3 遗传算法的设计与实现

- **2. 实数编码**

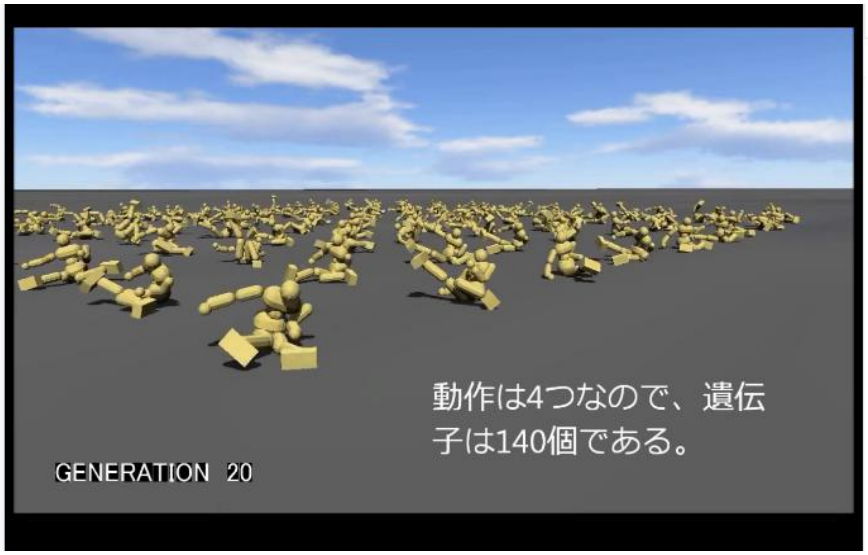
$$\min f(x_1, x_2) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$$
$$-5 \le x_i \le 5 \qquad i = 1, 2$$

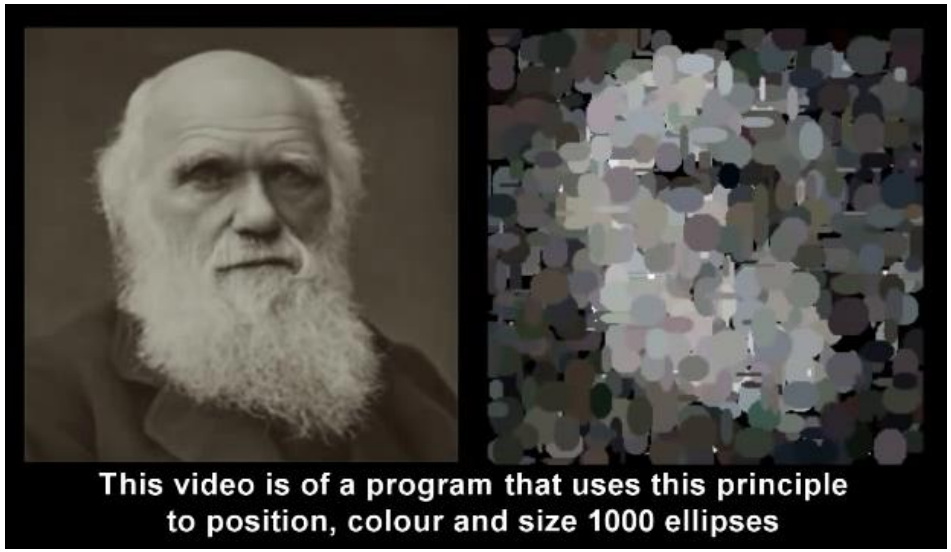
- 优点：

① 不必进行数制转换，可直接进行遗传操作；

② 适合于表示范围较大的数；

③ 适合于精度要求较高的遗传算法。





6.2.3 遗传算法的设计与实现

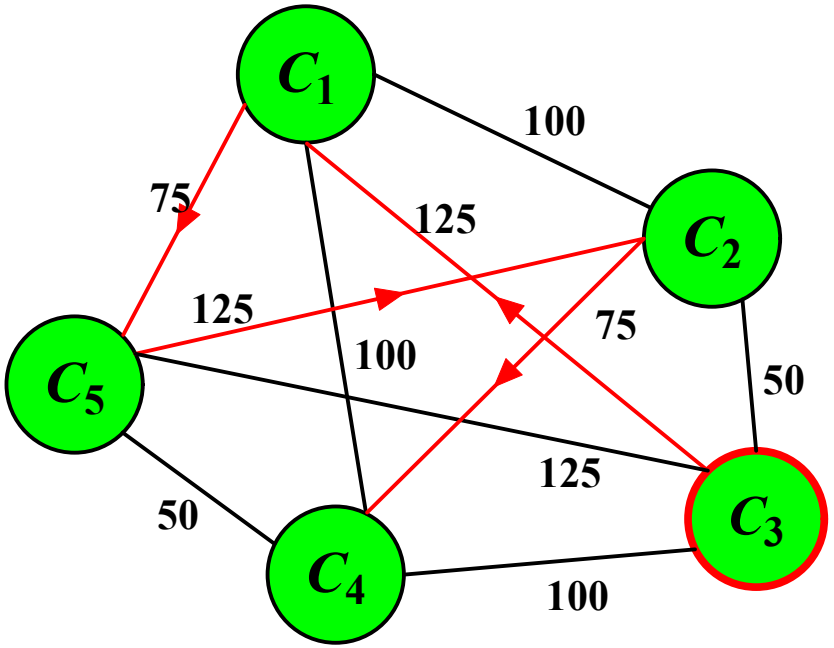
- **3. 有序串编码**

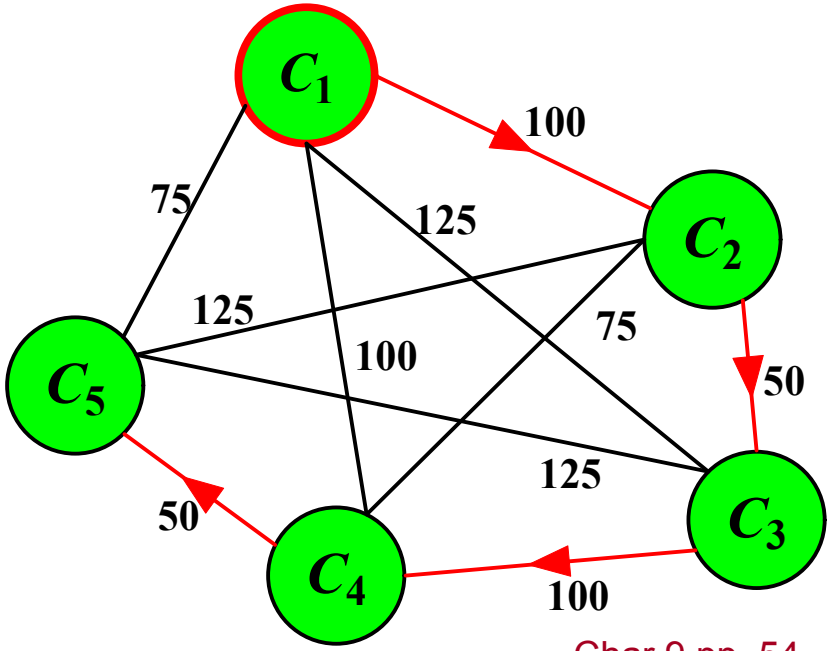
旅行商（TSP）问题、作业调度（job-scheduling）问题以及资源调度（resource-scheduling）问题等

- 有序问题：目标函数的值不仅与表示解的字符串的值有关，而且与其所在字符串的位置有关。
- 例如 5个城市的TSP问题的有序串编码用来表示遍历的城市顺序：

3 1 5 2 4: 路径长度为500

1 2 3 4 5: 路径长度为 375



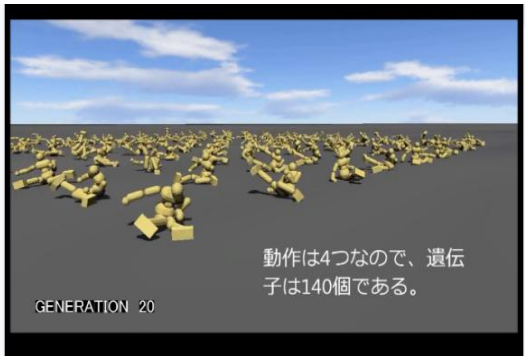


6.2.3 遗传算法的设计与实现

❑ 二、群体设定

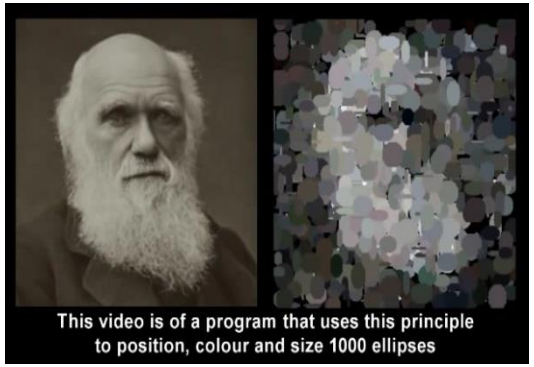
▪ 1. 种群规模的确定

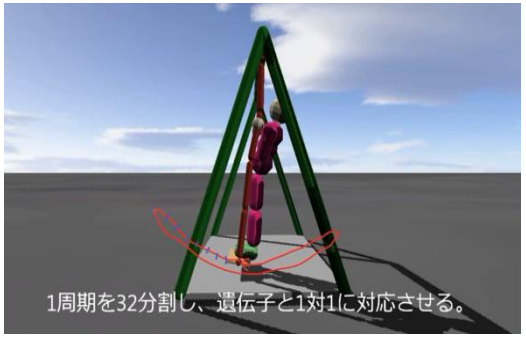
- 规模太小，易陷入局部最优解，出现早熟；
- 规模太大，计算量增加，影响算法效率。
- 一般种群规模 $N=20\sim200$。问题越难，种群规模应适当大些。



▪ 2. 初始种群的产生

- 完全随机的方法：用随机数发生器产生；
- 基于先验知识(例如最优解在整个问题空间的分布范围)的随机产生方法；
- 先随机产生一些个体，从中选取N个最好的个体作为初始种群。





6.2.3 遗传算法的设计与实现

❑ 三、适应度函数（求最大化，非负）设计

- 目标函数 $f(x)$ 到适应度函数 $Fit(x)$ 的变换

目标函数 $f(x)$	适应度函数 $Fit(x)$
max $f(x)$, $f(x)\geq 0$	$Fit(x)=f(x)$
min $f(x)$, $f(x)>0$	$Fit(x)=\dfrac{1}{f(x)}$
max $f(x)$	$Fit(x)=\begin{cases} f(x)-C_{min} & f(x)>C_{min} \\ 0 & \text{其他情况} \end{cases}$
min $f(x)$	$Fit(x)=\begin{cases} C_{max}-f(x) & f(x)<C_{max} \\ 0 & \text{其他情况} \end{cases}$

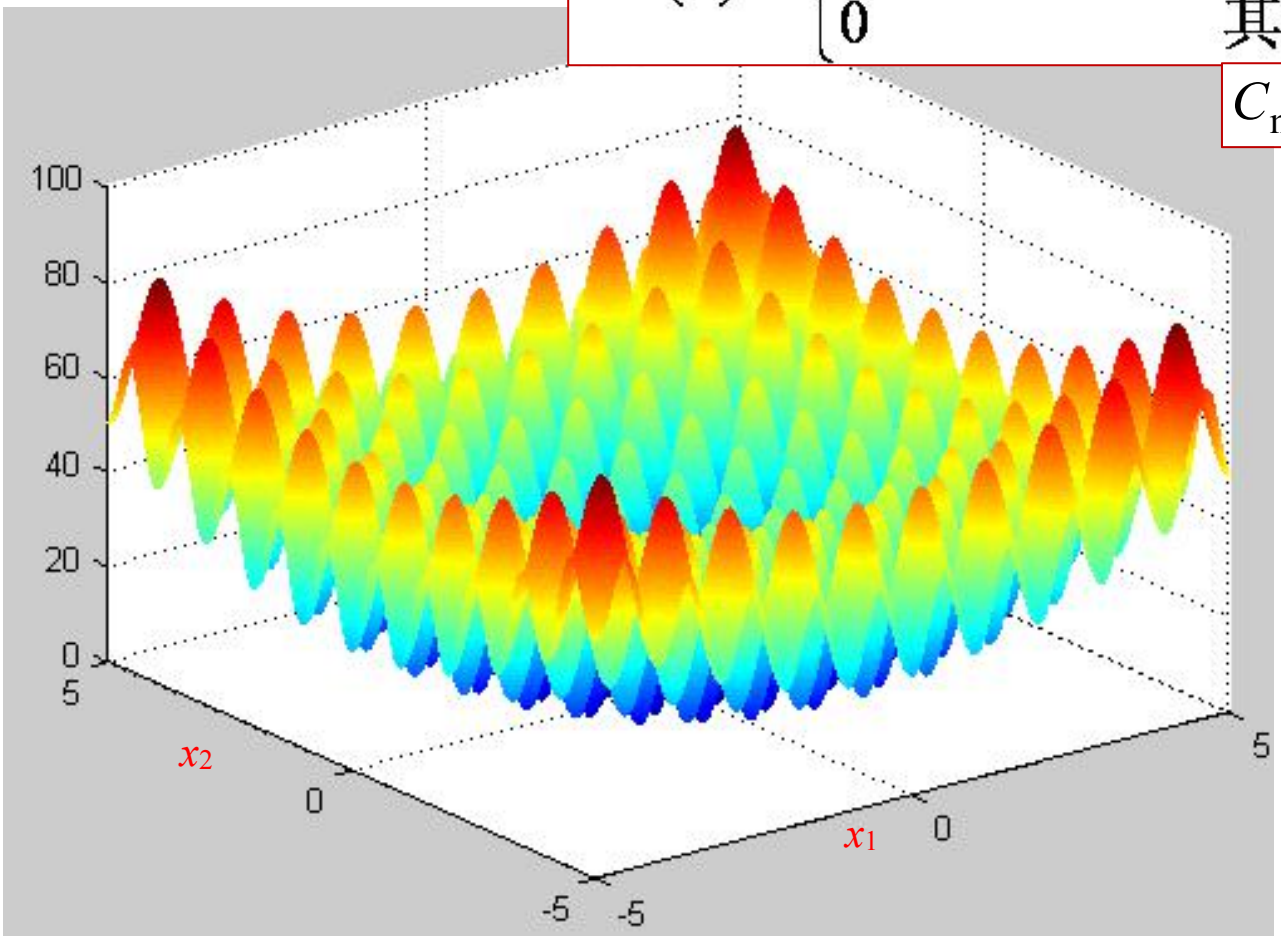
- 例： 用遗传算法求解下面一个Rastrigin函数的最小值。

$$f(x_1, x_2) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)$$

$$-5 \le x_i \le 5 \qquad i = 1, 2$$

$$Fit(x) = \begin{cases} C_{\max} - f(x) & f(x) < C_{\max} \\ 0 & \text{其他情况} \end{cases}$$

$$C_{\max} = 100$$



6.2.3 遗传算法的设计与实现

❑ 三、适应度函数（求最大化，非负）设计

- 目标函数 $f(x)$ 到适应度函数 $Fit(x)$ 的变换

✓适应度函数设计不当有可能会出现欺骗问题：
进化初期，个别超级个体控制选择过程；
进化末期，个体差异太小导致陷入局部极值。

✓适应度函数的尺度变换或者定标：对适应度函数值域的某种映射变换。p138

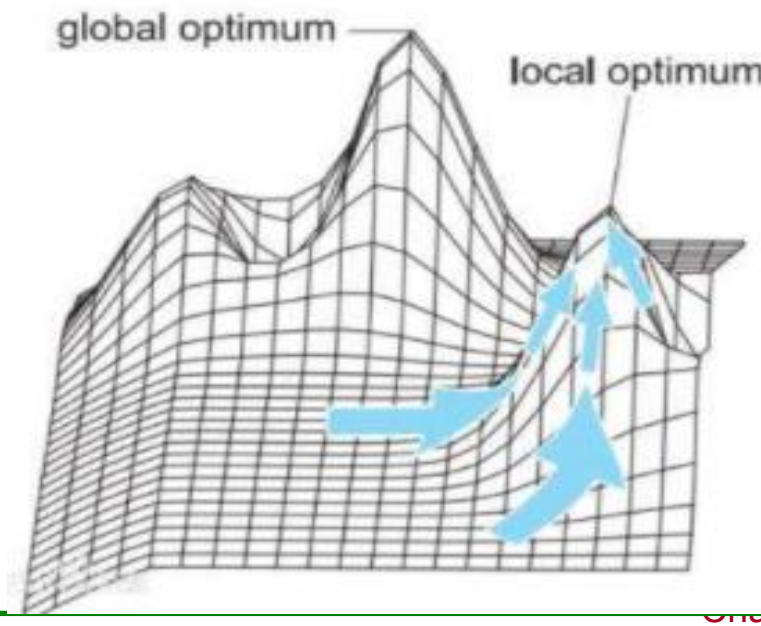
原适应度

$$f' = af + b$$

定标后的适应度

$$a = \frac{f_{avg}}{f_{avg} - f_{\min}}$$

$$b = \frac{-f_{\min} f_{avg}}{f_{avg} - f_{\min}}$$



6.2.3 遗传算法的设计与实现

选择操作：确定从亲代种群中按某种方法选取哪些个体遗传到下一代种群中的

四、选择（复制）

- 1. 个体选择概率分配方法
- (1) 适应度比例方法(fitness proportional model) 或蒙特卡罗法(Monte Carlo)

- 各个个体的选择概率和其适应度值成比例。
- 个体 *i*的选择概率：

$$p_{si} = \frac{f_i}{\sum_{i=1}^{N} f_i}$$

选择操作的主要目的是在保持种群大小恒定的情况下复制种群中适应度高的个体，去除种群中适应度低的个体。

6.2.3 遗传算法的设计与实现

- **1. 个体选择概率分配方法**
- **(2) 排序方法 (rank-based model)**

① 线性排序: **J. E. Baker**

- 个体按适应度值从大到小依次排列： $x_1, x_2, \cdots, x_N$
- 个体 x_i 分配选择概率 $p_i = \dfrac{a - bi}{N(N+1)}$

② 非线性排序: **Z. Michalewicz**

- 将群体成员按适应值从大到小依次排列，并按下式分配选择概率：

$$p_i = \begin{cases} q(1-q)^{i-1} & i = 1,2,\ldots, N-1 \\ (1-q)^{N-1} & i = N \end{cases}$$

6.2.3 遗传算法的设计与实现

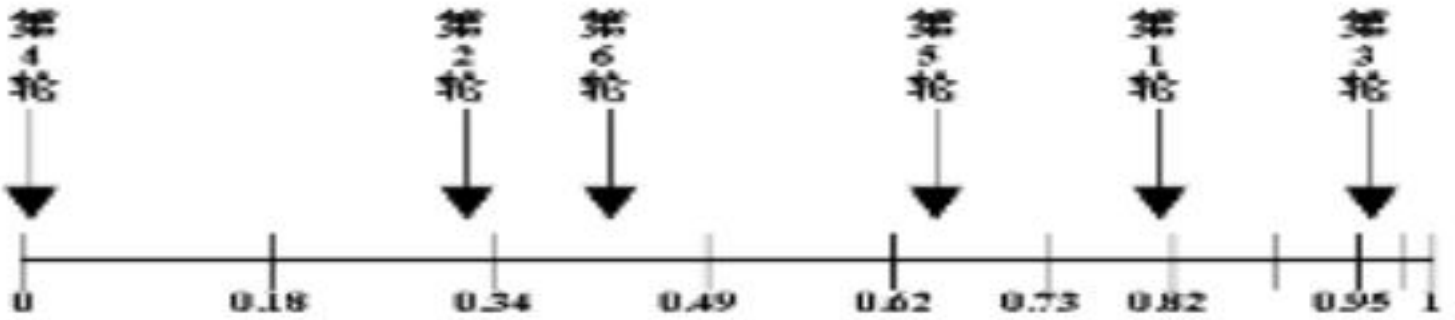
四、选择（复制）

2. 选择个体方法

1）转盘赌选择（roulette wheel selection）

个体	1	2	3	4	5	6	7	8	9	10	11
适应度	2.0	1.8	1.6	1.4	1.2	1.0	0.8	0.6	0.4	0.2	0.1
选择概率	0.18	0.16	0.15	0.13	0.11	0.09	0.07	0.06	0.03	0.02	0.0
累积概率	0.18	0.34	0.49	0.62	0.73	0.82	0.89	0.94	0.97	0.99	1.00

- 第1轮产生一个随机数：0.81
- 第2轮产生一个随机数：0.32



6.2.3 遗传算法的设计与实现

- 2. 选择个体方法
- 2）锦标赛选择方法(tournament selection model)

- 从群体中随机选择 k 个个体，将其中适应度最高的个体保存到下一代。这一过程反复执行，直到保存到下一代的个体数达到预先设定的数量为止。

- 优点：
- 能得到更加多样化的群体
- 能避免超级个体的影响，一定程度避免了过早收敛和停滞现象的发生

6.2.3 遗传算法的设计与实现

- 2. 选择个体方法
- 3）（ μ, λ ）和 μ ＋ λ选择

➢ (μ, λ) 选择：从规模为 μ 的群体所生成的 λ ($\geq \mu$)个后代中选取 μ 个最优的后代作为新一代种群。

➢ μ＋λ 选择：从λ 个后代与其父代中选取 μ 个最优的后代作为新一代种群。

- 4）最佳个体（elitist model）保存方法

- 把群体中适应度最高的一个或多个（2%~5%）个体不进行交叉而直接复制到下一代中，保证遗传算法终止时得到的最后结果一定是历代出现过的最高适应度的个体。

6.2.3 遗传算法的设计与实现

交叉操作：模仿自然界有性繁殖的基因重组过程，对两个父代个体进行基因操作，作用是把原有优良基因遗传到下一代种群中，并生成包含更复杂基因结构的新个体。

五、交叉（基因重组）

- 1. 基本的交叉算子（二进制编码）
- 1）一点交叉（单点交叉）

A_1= 0 1 1 0 **1**　　A_1= 0 1 1 0 **0**

A_2= 1 1 0 0 **0**　　A_2= 1 1 0 0 **1**

- 2）二点交叉（两点交叉）

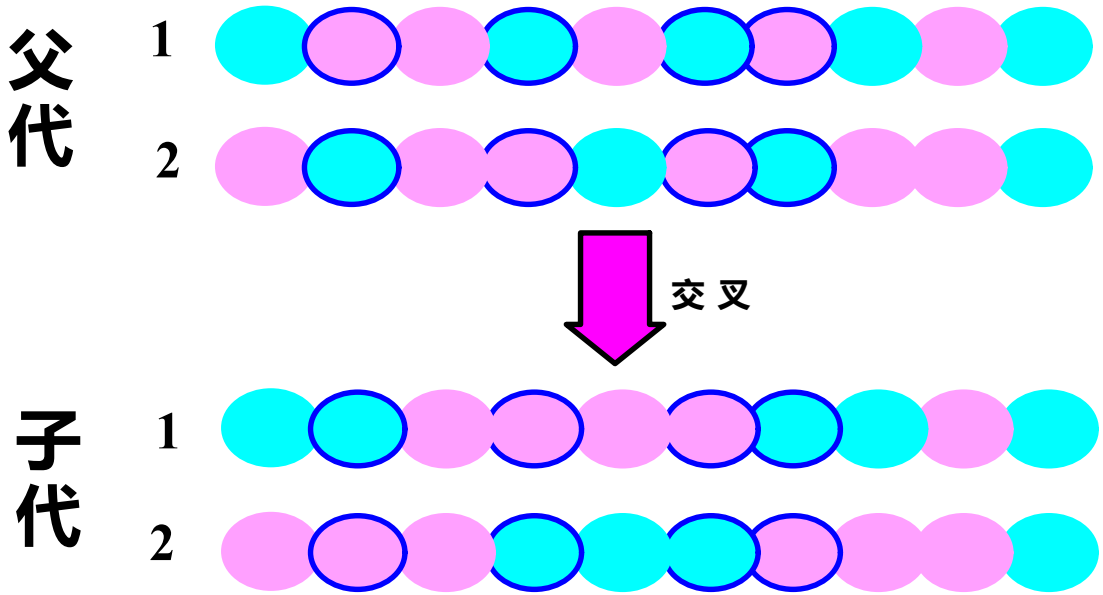
A_1= 0 1 **1 0** 1　　A_1= 0 1 **0 0** 0

A_2= 1 1 **0 0** 0　　A_2= 1 1 **1 0** 1

6.2.3 遗传算法的设计与实现

❑ 五、交叉（基因重组）

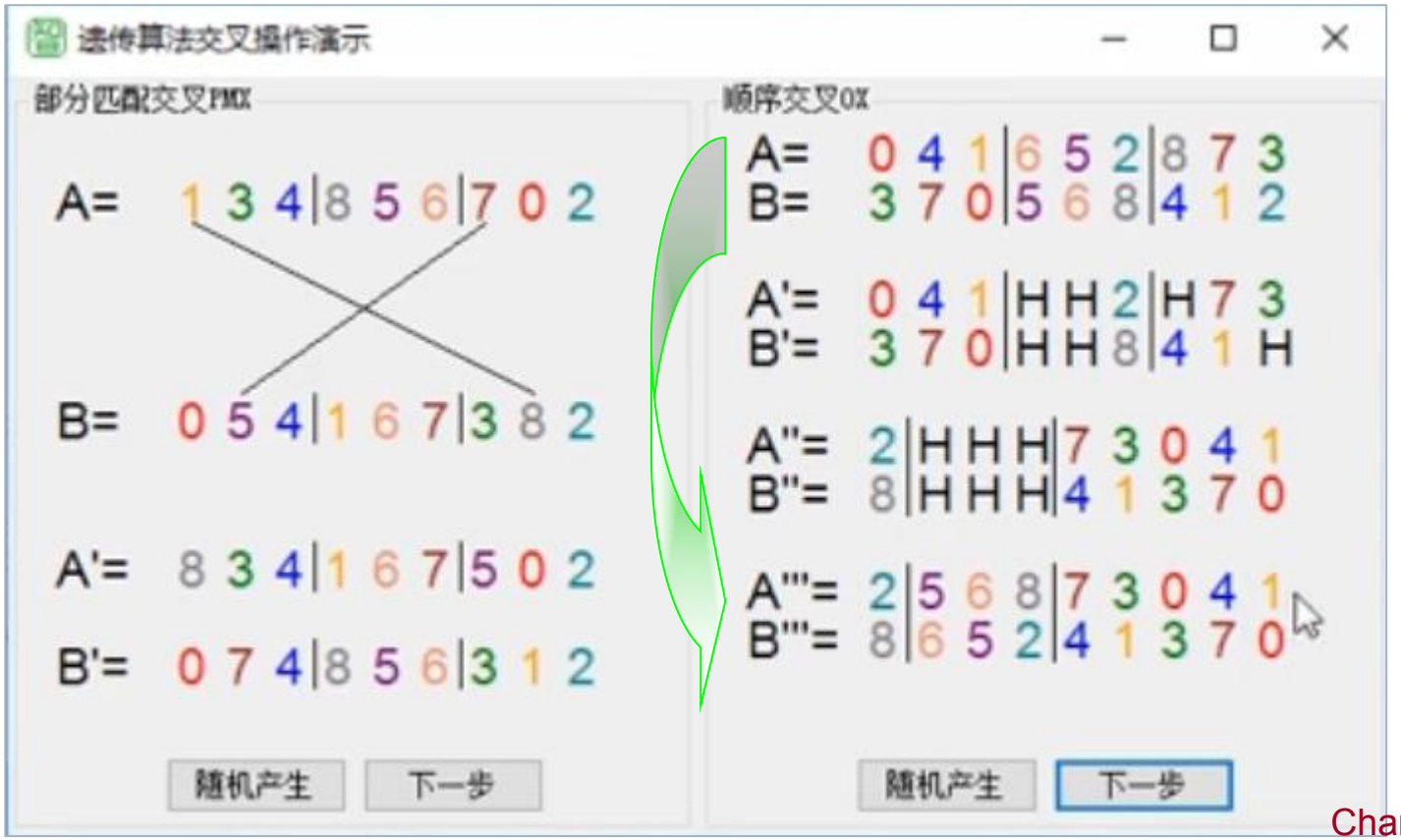
- **1. 基本的交叉算子**
- **3）均匀交叉（uniform crossover)或一致交叉）**



6.2.3 遗传算法的设计与

交叉算子的设计和实现与所研究的问题密切相关，一般要求它既不要太多地破坏个体编码串中表示优良性状的优良模式，又要能够有效地产生出一些较好的新个体模式。

- 2. 改进的交叉算子（有序
- 1）部分匹配交叉PMX： G
- 2）顺序交叉OX: Davis L.(



6.2.3 遗传算法的设计与实现

六、变异

将个体染色体编码串中某些基因座上的基因值用该基因座的其他等位基因来替换，从而形成一个新的个体。

6.2.3 遗传算法的设计与实现

p_c越大，产生新个体的速度就越快；
p_c过大，组块被破坏的概率也越大；
p_c过小，使搜索变慢、停滞。

❑ 七、控制参数选择

- 需要选择的参数：种群规模 N，交叉概率 p_c，变异概率 p_m 等。

p_m过小，不易产生新个体；
p_m过大，变成纯粹的随机搜索算法。

- 遗传算法运行时选择的参数应该视解决的具体问题而定，到目前为止，还没有一个适用于遗传算法所有应用领域的关于算法参数的理论。
- 常用的参数范围：$N=20\sim200$

$$p_c=0.85\sim1.0$$

$$p_m=0.001\sim0.15$$

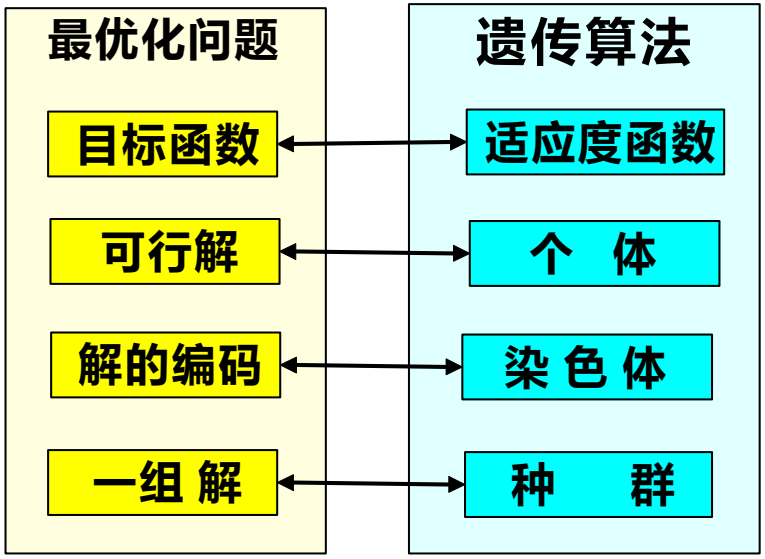
第6章 智能计算及其应用

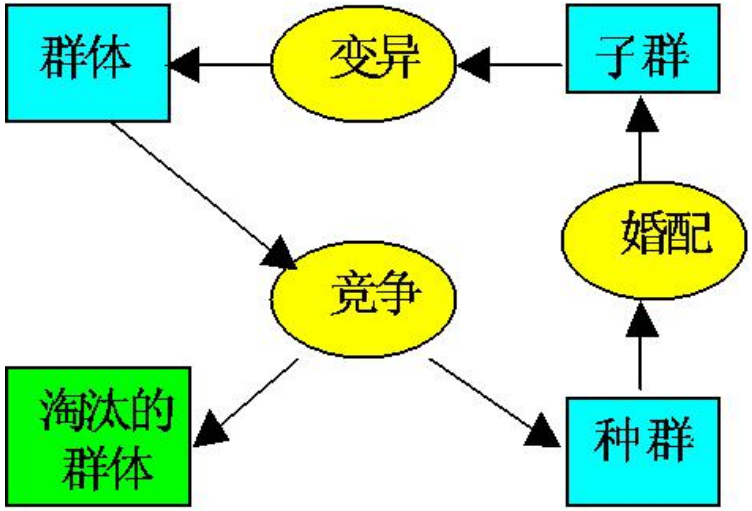
- 6.1 进化算法的产生与发展
- 6.2 基本遗传算法
- 6.3 遗传算法的改进算法
- 6.4 遗传算法的应用
- 6.5 群智能算法产生的背景
- 6.6 粒子群优化算法及其应用
- 6.7 蚁群算法及其应用

- MOOC上：第8讲的课堂讨论、单元测试、在线作业
- 网络教学平台作业W6-2
- 准备下周四的遗传算法实验

遗传算法实验

- 解的编码
- 初始种群的设定
- 适应度函数的设计
- 遗传操作（选择、交叉、变异）
- 算法控制参数的设定
- 约束条件的处理



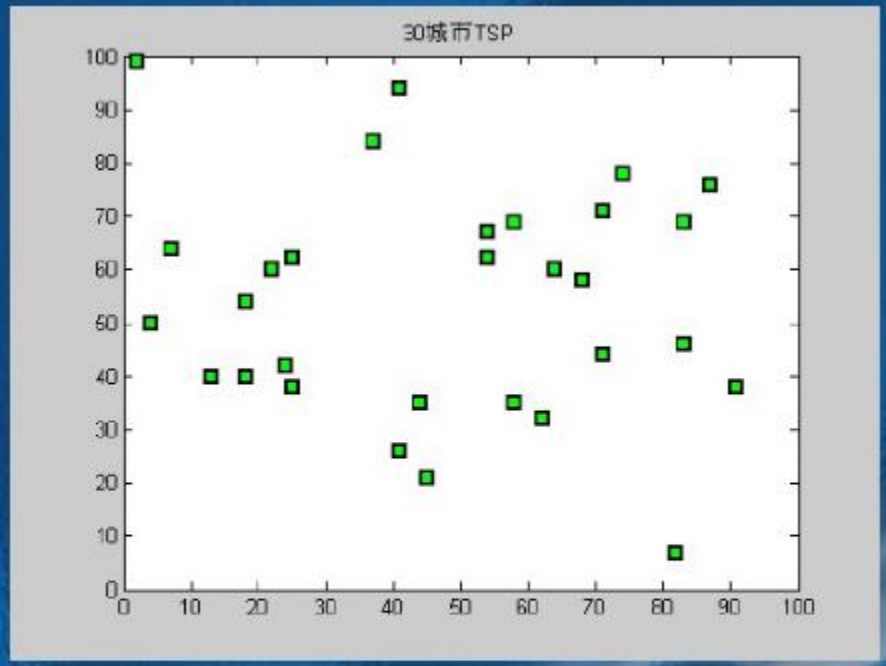


Char 9 pp. 70

遗传算法求解30个城市的TSP问题

- TSP Benchmark 问题

41 94;37 84;54 67;25 62;
7 64;2 99;68 58;71 44;54
62;83 69;64 60;18 54;22
60;83 46;91 38;25 38;24
42;58 69;71 71;74 78;87
76;18 40;13 40;82 7;62 32;
58 35;45 21;41 26;44 35;4 50



30个城市

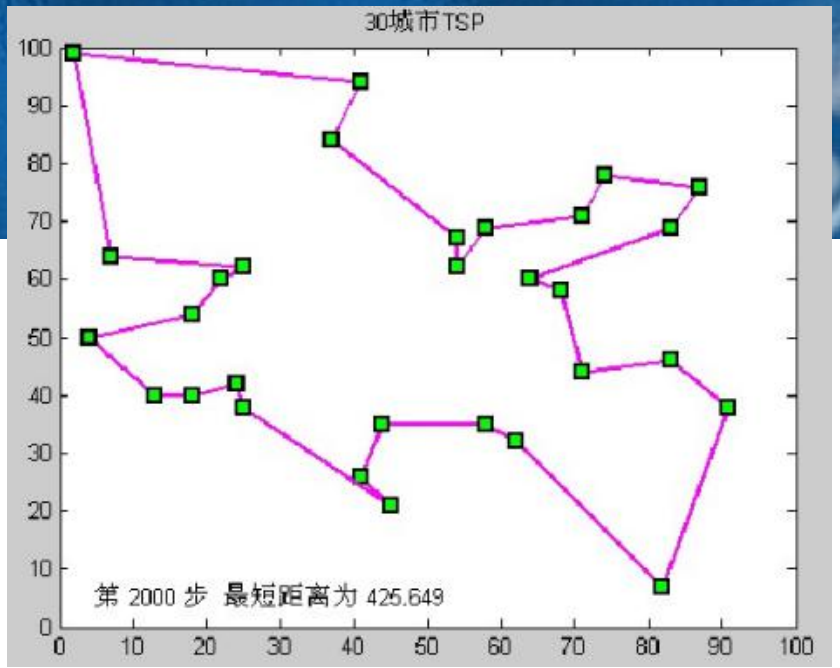
遗传算法求解30个城市的TSP问题

- **TSP Benchmark 问题**

  编码：直接采用解的表示形式，30位（30个城市）长，每位代表所经过的城市序号（无重复）；

  适应度函数：个体所代表的路径距离的倒数；

  选择：轮盘赌方法



遗传算法求解30个城市的TSP问题

- **TSP Benchmark 问题**

交叉：有序交叉法

1）随机选取两个交叉点；

2）两个父个体交换中间部分；

3）替换交换后重复的城市序号。

X1: 9 8 | 4 5 6 7 1 | 3 2 0　　X1': 9 8 | 1 4 0 3 2 | 3 2 0

X2: 8 7 | 1 4 0 3 2 | 9 6 5　　X2': 8 7 | 4 5 6 7 1 | 9 6 5

X1': 9 8 | 1 4 0 3 2 | 7 5 6

X2': 8 3 | 4 5 6 7 1 | 9 0 2

遗传算法求解30个城市的TSP问题

- TSP Benchmark 问题

  变异：随机选择同一个个体的两个点进行交换；

  初始参数：

  种群规模 100

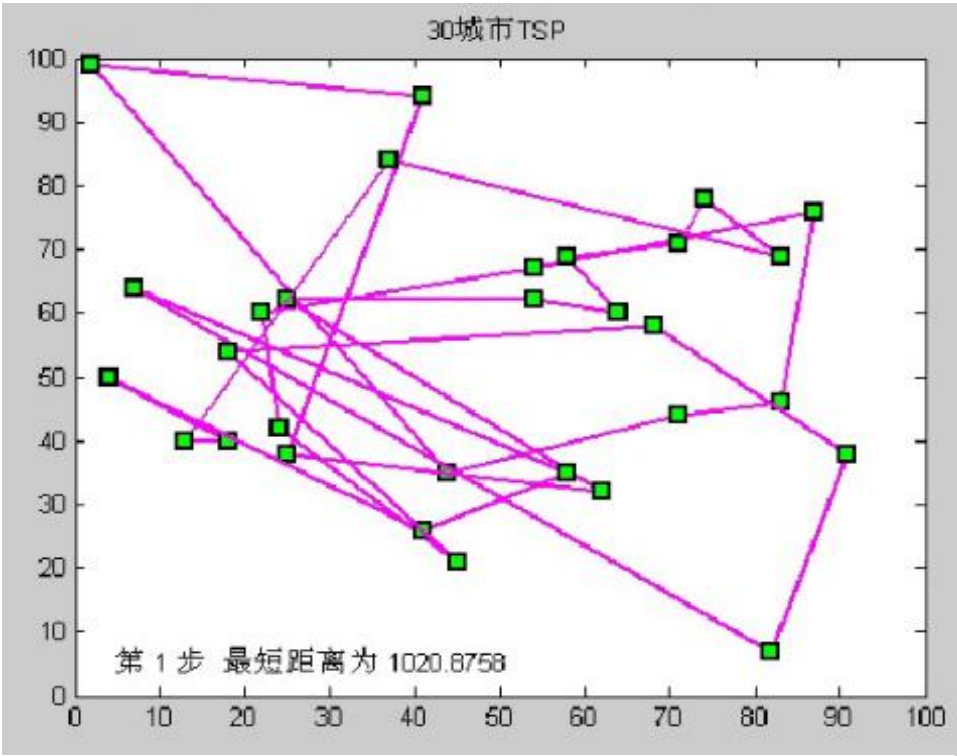
  交叉概率 0.8

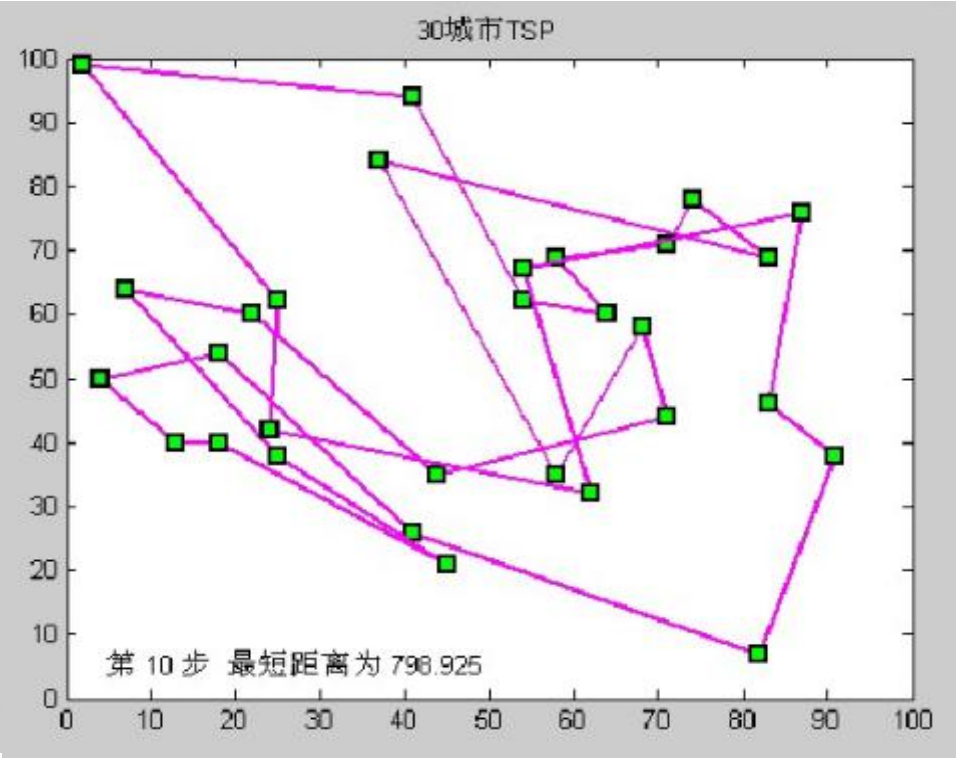
  变异概率 0.8

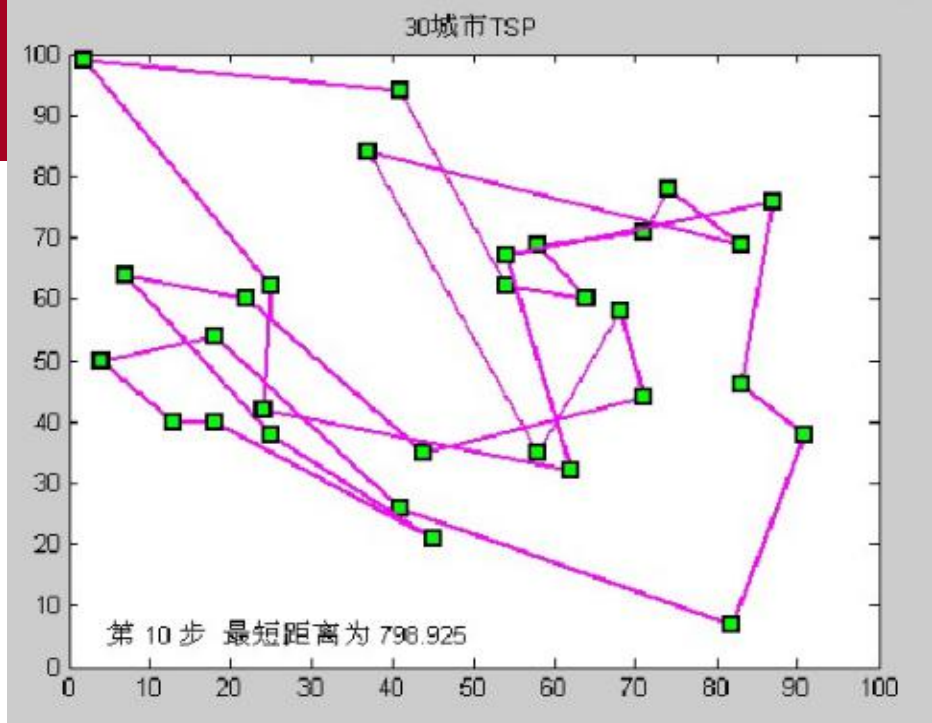
  终止代数 2000

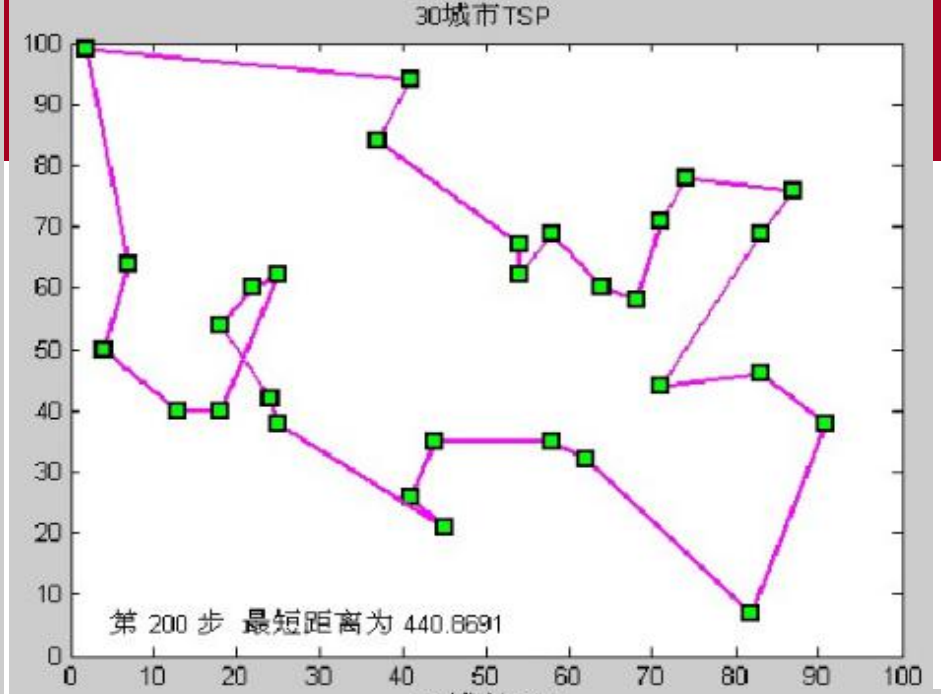
遗传算法求解30个城市的TSP问题

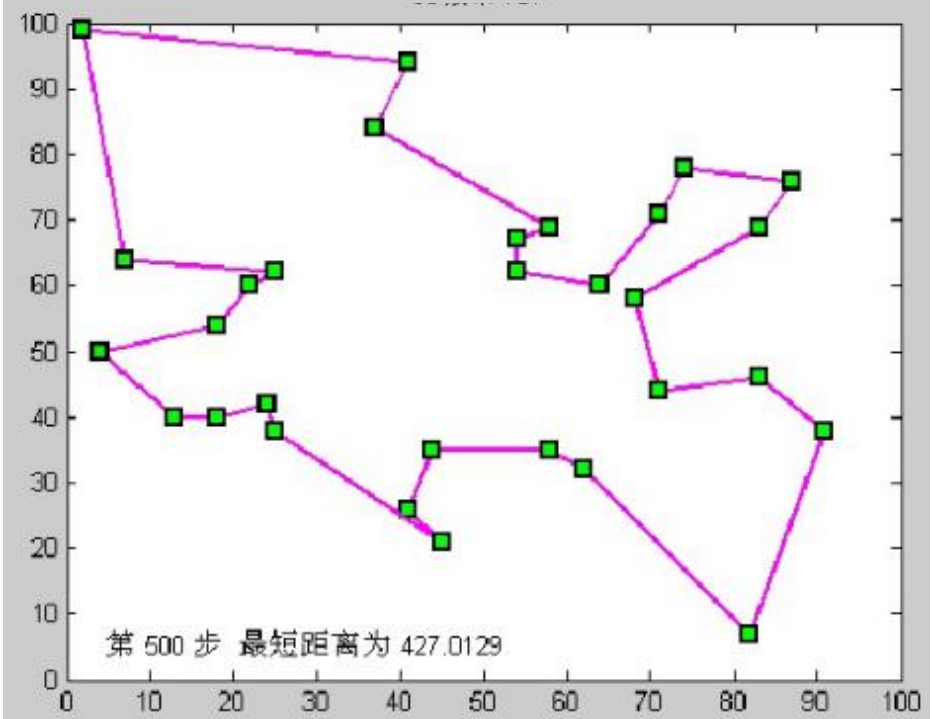
- 运行结果：

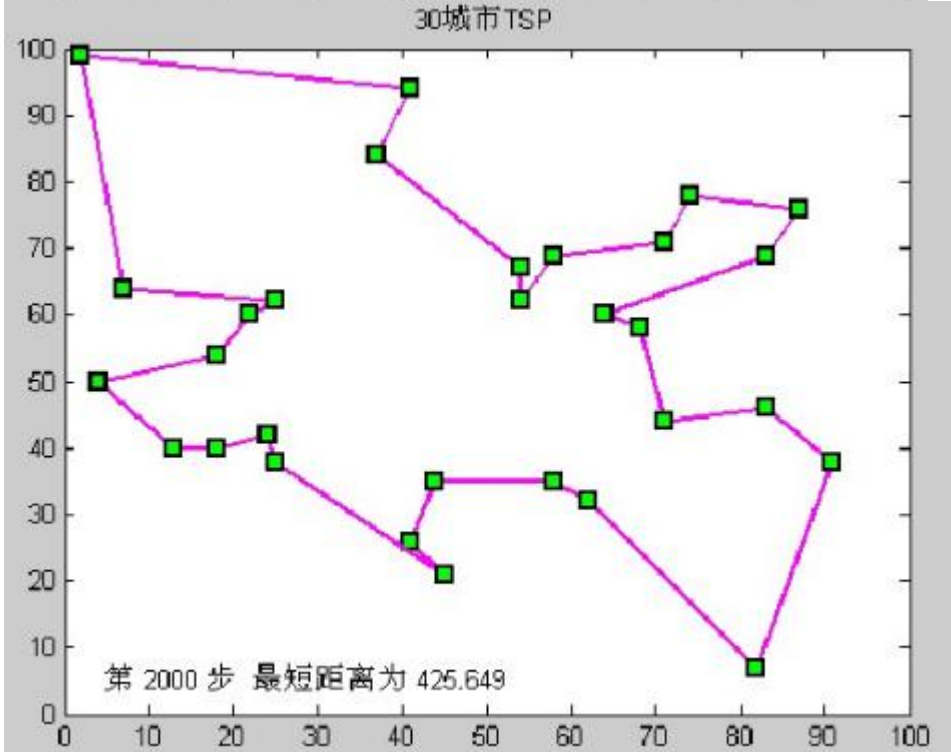




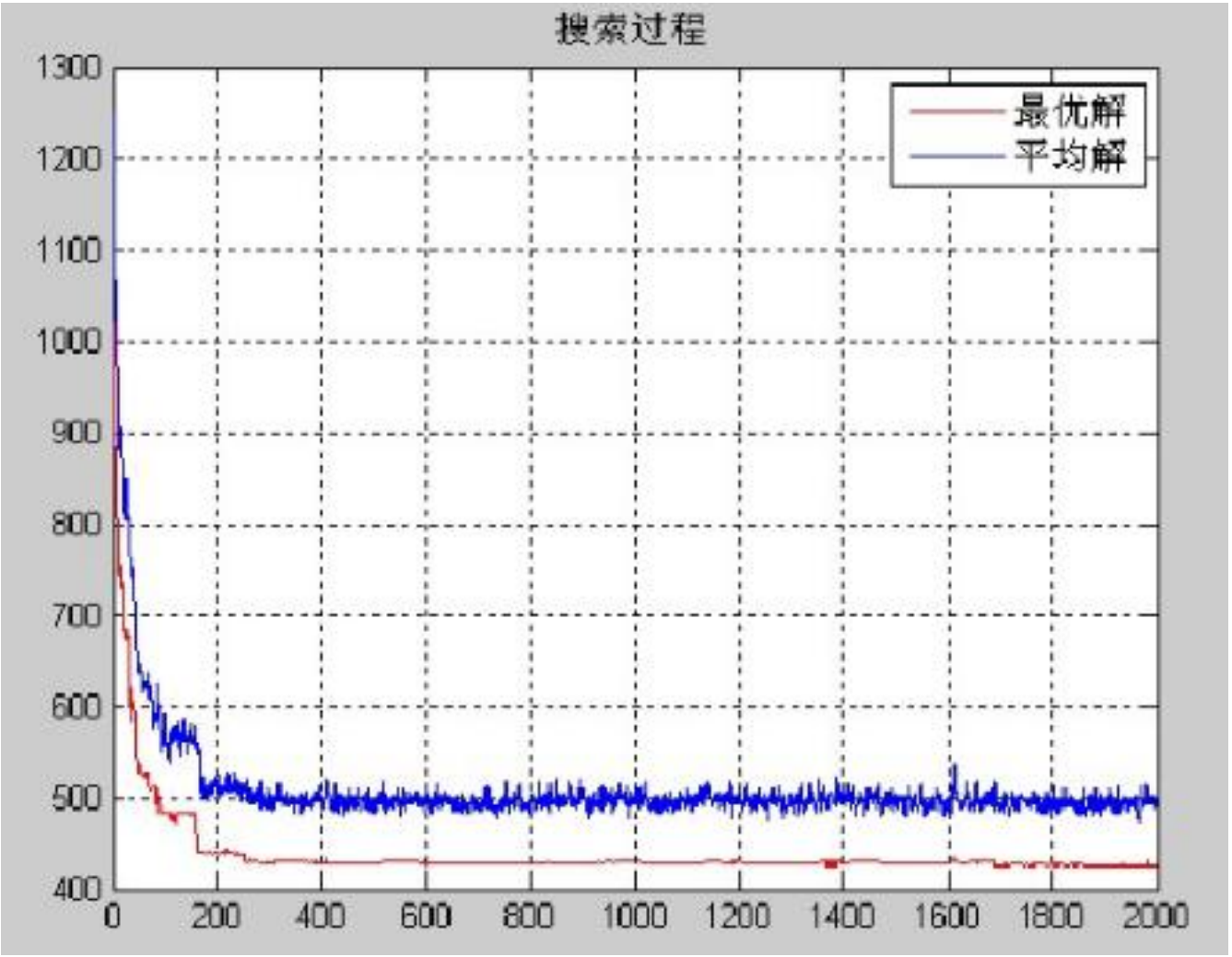

30城市TSP
第 10 步 最短距离为 798.925


30城市TSP
第 200 步 最短距离为 440.8691


第 500 步 最短距离为 427.0129


30城市TSP
第 2000 步 最短距离为 425.649

遗传算法求解30个城市的TSP问题



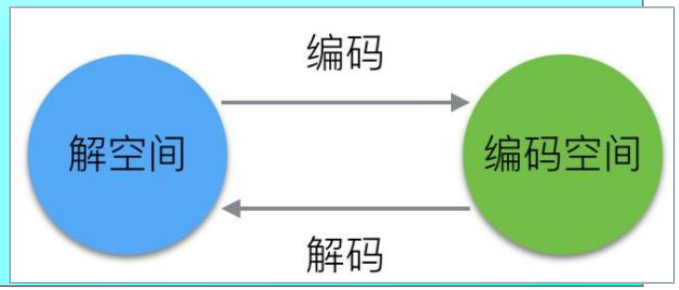
第6章 智能计算及其应用

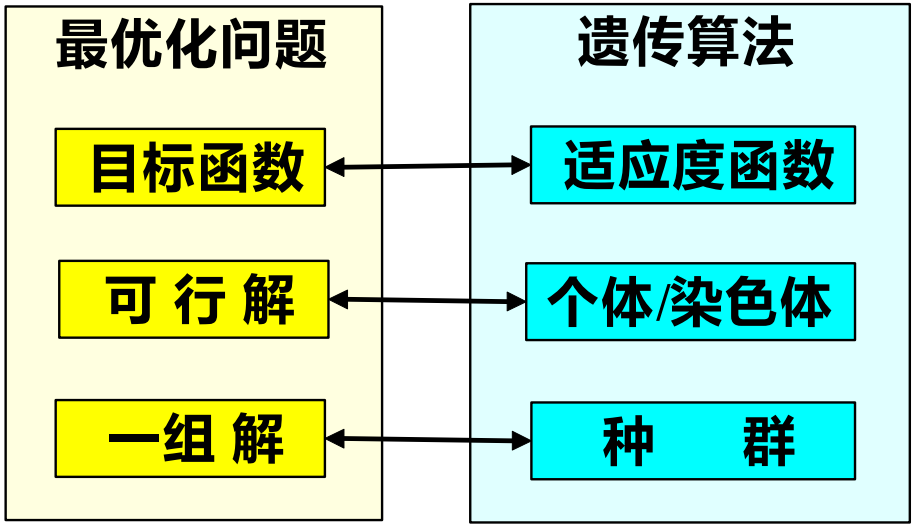
- 6.1 进化算法的产生与发展
- 6.2 基本遗传算法
- 6.3 遗传算法的改进算法
- 6.4 遗传算法的应用
- 6.5 群智能算法产生的背景
- 6.6 粒子群优化算法及其应用
- 6.7 蚁群算法及其应用

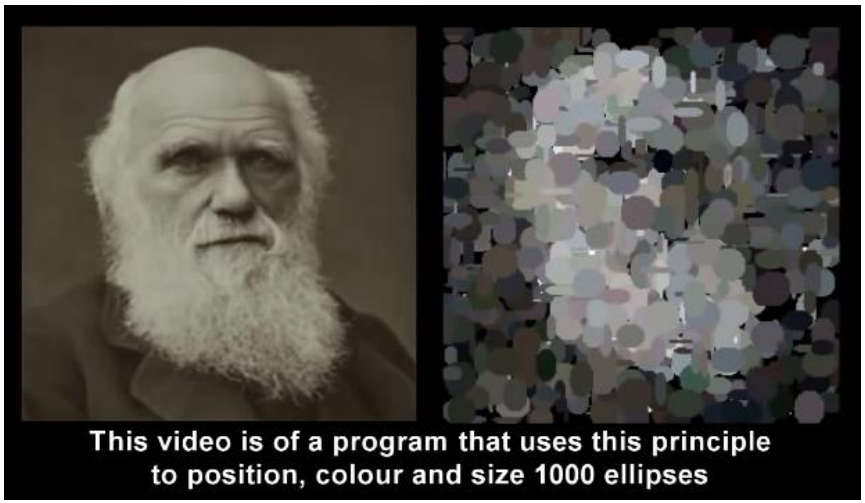
- MOOC上：第4讲在线作业互评
- 准备本周四的遗传算法实验

❑ 基本遗传算法的不足:

- 早熟: 遗传算法的探索能力是有限的，易收敛到局部最优解。
- 大量计算: 当问题复杂时，计算时间是个问题。
- 处理规模小。目前对于维数较高的问题，还是很难处理和优化的。
- 稳定性差: 遗传算法属于随机类算法，不能稳定的得到解，需要多次运算。







6.3 遗传算法的改进算法

- 6.3.1 双倍体遗传算法
- 6.3.2 双种群遗传算法
- 6.3.3 自适应遗传算法

■ 基本遗传算法的不足：

- 早熟：已陷入局部极值
- 大量计算
- 稳定性差

● 改进策略大致可以概括为以下几个方面：

(1) 改进遗传算法的组成成分或使用技术，如选用优化控制参数、适合问题特性的编码技术等；

(2) 采用 混合遗传算法，例如禁忌搜索、模拟退火、混沌等；

(3) 采用动态自适应技术，在进化过程中调整控制参数；

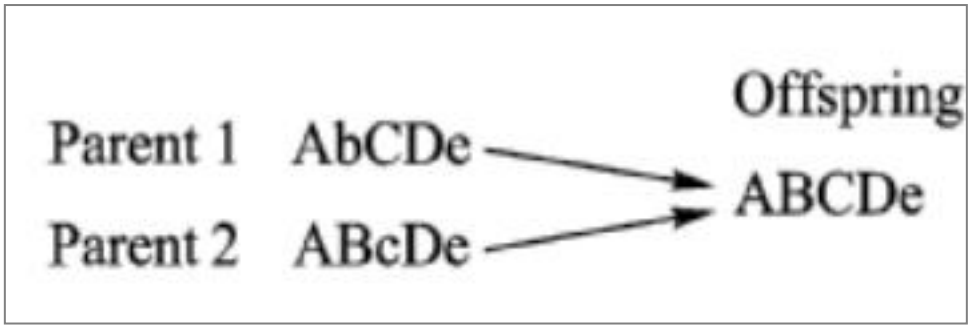
(4) 采用并行遗传算法；

(5) 采用非标准的遗传操作算子，例如灾变。

6.3.1 双倍体遗传算法

1. 基本思想

- 双倍体遗传算法采用显性和隐性两个染色体同时进行进化，提供了一种记忆以前有用的基因块的功能。
- 双倍体遗传算法采用显性遗传。



- 双倍体遗传延长了有用基因块的寿命，提高了算法的收敛能力，在变异概率低的情况下能保持一定水平的多样性。

6.3.1 双倍体遗传算法

2. 双倍体遗传算法的设计

（1）编码：两个染色体（显性、隐性）

（2）复制(选择)算子：计算显性染色体的适应度，按照显性染色体的选择概率将个体复制到下一代群体中。

（3）交叉算子：两个个体的显性染色体交叉、隐性染色体也同时交叉。

（4）变异算子：个体的显性染色体按正常的变异概率变异；隐性染色体按较大的变异概率变异。

（5）显隐性重排算子：个体中适应值较大的染色体设为显性染色体，适应值较小的染色体设为隐性染色体。

6.3 遗传算法的改进算法

- **6.3.1 双倍体遗传算法**
- **6.3.2 双种群遗传算法**
- **6.3.3 自适应遗传算法**

6.3.2 双种群遗传算法

1. 双种群遗传算法：建立两个遗传算法群体，分别独立地运行复制、交叉、变异操作，同时当每一代运行结束以后，选择两个种群中的随机个体及最优个体分别交换。

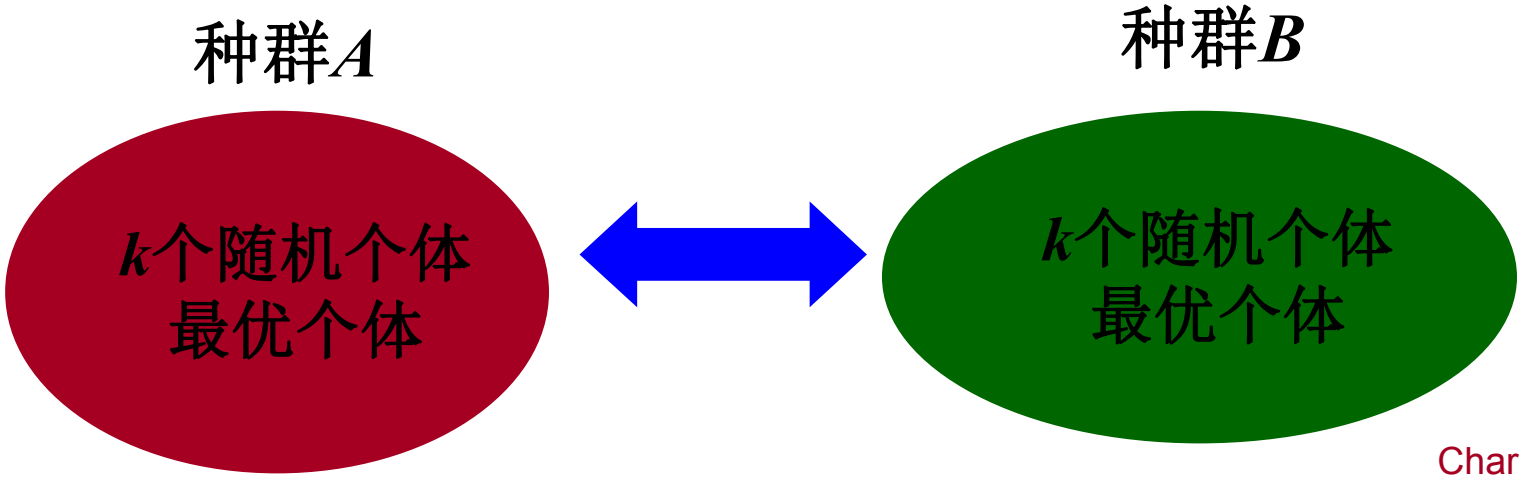
2. 双种群遗传算法的设计

（1）编码设计

（2）交叉算子、变异算子

（3）杂交算子（迁移操作）

打破原种群内的平衡态达到更高的平衡态，有利于算法跳出局部最优。



6.3 遗传算法的改进算法

- 6.3.1 双倍体遗传算法
- 6.3.2 双种群遗传算法
- 6.3.3 自适应遗传算法

Srinvivas M.，Patnaik L. M.等在1994年提出一种自适应遗传算法(adaptive genetic algorithms，AGA)：交叉概率P_c和变异概率P_m能随个体的适应度自动改变。

6.3.3 自适应遗传算法

1. 基本思想

- **AGA**：当种群各个体适应度趋于一致或者趋于局部最优时，使 P_c和P_m增加，以跳出局部最优；而当群体适应度比较分散时，使 P_c和P_m减少，以利于优良个体的生存。
- 对于适应度高于群体平均适应度(f_{avg})的个体，选择较低的 P_c和P_m，使得该解得以保护进入下一代；对低于平均适应度(f_{avg})的个体，选择较高的 P_c和P_m ，使该解被淘汰。

$$P_c=\begin{cases}\dfrac{k_1(f_{max}-f')}{f_{max}-f_{avg}}, & f'>f_{avg}\\ k_2, & f'\le f_{avg}\end{cases}$$

$$P_m=\begin{cases}\dfrac{k_3(f_{max}-f)}{f_{max}-f_{avg}}, & f>f_{avg}\\ k_4, & f\le f_{avg}\end{cases}$$

交叉个体中较大的适应度

$k_2>k_1$，$k_4>k_3$

变异个体的适应度

6.3.3 自适应遗传算法

2. 自适应遗传算法的步骤

（1） 编码设计。

（2） 初始种群产生：产生N(N 是偶数)个候选解，组成初始种群。

（3） 计算适应度 f_i 。

（4） 按轮盘赌规则选择N 个个体，计算f_{avg}和f_{max} 。

（5）将群体中的各个个体随机搭配成对，共组成$N/2$对。对每一对个体，按照自适应公式计算自适应交叉概率P_c，随机产生$R(0,1)$，如果$R<P_c$, 则对该对染色体进行交叉操作。

6.3.3 自适应遗传算法

2. 自适应遗传算法的步骤（续）

（6）对于群体中的所有个体，共N个，按照自适应变异公式计算自适应变异概率P_m，随机产生$R(0,1)$，如果$R<P_m$，则对该染色体进行变异操作。

（7）计算由交叉和变异生成新个体的适应度，新个体与父代一起构成新群体。

（8）判断是否达到预定的迭代次数，是则结束；否则转第（4）步。

6.3.3 自适应遗传算法

交叉中个体较大的适应度

3. 自适应的交叉概率与变异概率

变异个体的适应度

$$P_c=\begin{cases}\dfrac{k_1(f_{\max}-f')}{f_{\max}-f_{\text{avg}}}, & f'>f_{\text{avg}}\\ k_2, & f'\le f_{\text{avg}}\end{cases}$$

$$P_m=\begin{cases}\dfrac{k_3(f_{\max}-f)}{f_{\max}-f_{\text{avg}}}, & f>f_{\text{avg}}\\ k_4, & f\le f_{\text{avg}}\end{cases}$$

- 普通自适应算法中，当个体适应度值越接近最大适应度值时，交叉概率与变异概率就越小；当等于最大适应度值时，交叉概率和变异概率为零。这种方法对于进化后期比较合适，但进化初期不利。

- 改进的思想：当前代的最优个体的交叉概率与变异概率最小，较优个体要对应于较大的交叉概率与变异概率，较差个体对应于最大的交叉概率与变异概率。

6.3.3 自适应遗传算法

3. 自适应的交叉概率与变异概率（进一步改进）

- **F—自适应方法：**

$$P_c=\begin{cases}P_{c1}-\dfrac{(P_{c1}-P_{c2})(f'-f_{avg})}{f_{max}-f_{avg}}, & f'>f_{avg}\\ P_{c1}, & f'\le f_{avg}\end{cases}$$

$$P_m=\begin{cases}P_{m1}-\dfrac{(P_{m1}-P_{m2})(f-f_{avg})}{f_{max}-f_{avg}}, & f>f_{avg}\\ P_{m1}, & f\le f_{avg}\end{cases}$$

$$P_{c1}=0.9, P_{c2}=0.6, P_{m1}=0.1, P_{m2}=0.001$$

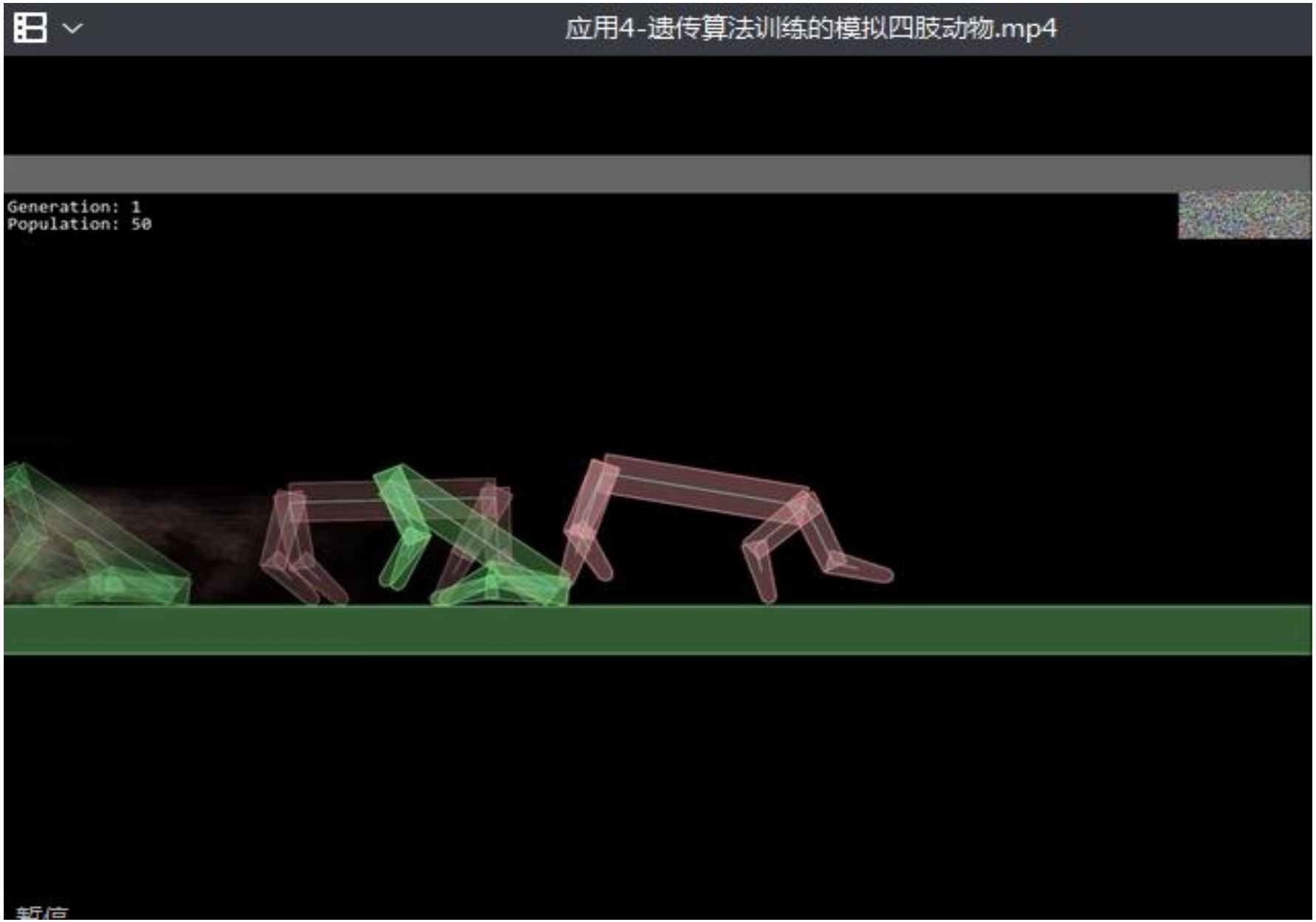
当$f'=f_{max}$，$P_c=P_{c2}>0$;

当$f=f_{max}$，$P_m=P_{m2}>0$。

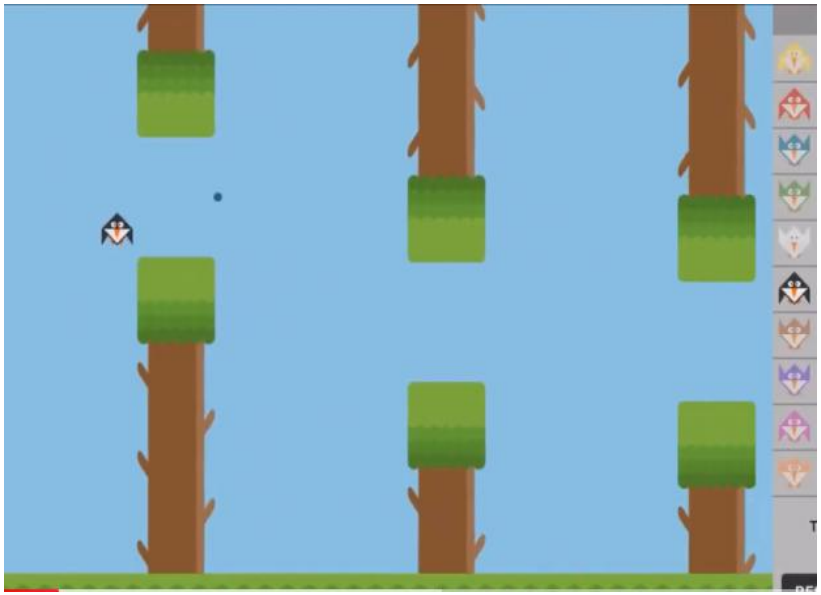
- 最优个体的交叉概率与变异概率最小
- 较优个体要对应于较大的交叉概率与变异概率
- 较差个体对应于最大的交叉概率与变异概率

- 6.1 进化算法的产生与发展
- 6.2 基本遗传算法
- 6.3 遗传算法的改进算法
- 6.4 遗传算法的应用
- 6.5 群智能算法产生的背
- 6.6 粒子群优化算法及其
- 6.7 蚁群算法及其应用

- 基于小生境技术的遗传算法：包括预选机制、排挤机制、共享机制，具有更好的多样性和全局搜索性能。
- 多目标遗传算法


应用4-遗传算法训练的模拟四肢动物.mp4
Generation: 1
Population: 50

遗传算法与人工神经网络相结合的应用



Flappy Bird游戏

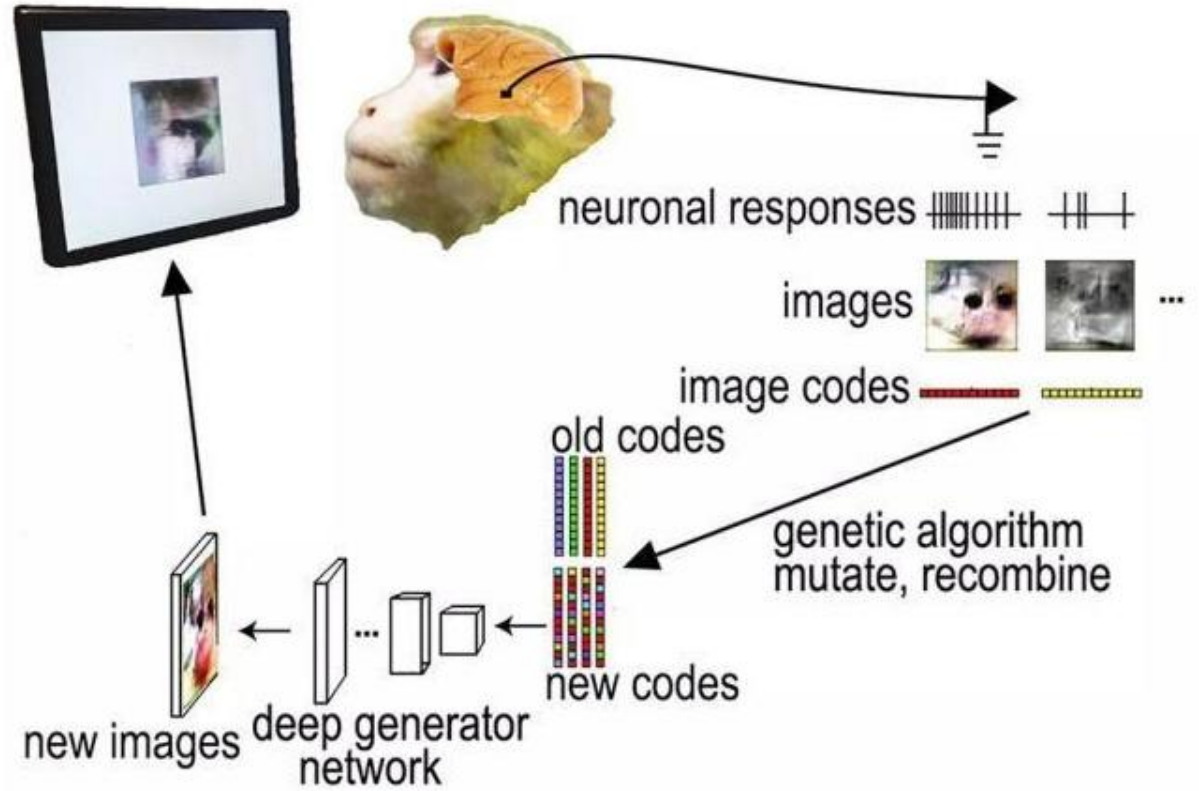
https://haokan.baidu.com/v?vid=11540271615102508208&pd=bjh&fr=bjhauthor&type=video

学赛车

https://www.bilibili.com/video/av32652759/?p=2

遗传算法与人工神经网络相结合的应用

Evolving images for visual neurons using a deep generative network reveals coding principles and neural preferences, 2019.5.2



在神经元放电的引导下，深度神经网络和遗传算法进化生成图像演进的图像使猕猴视觉皮层的神经元放电最大化演进的图像比大量的自然图像更能激活神经元与演进图像的相似性可以预测神经元对新图像的反应

THE END