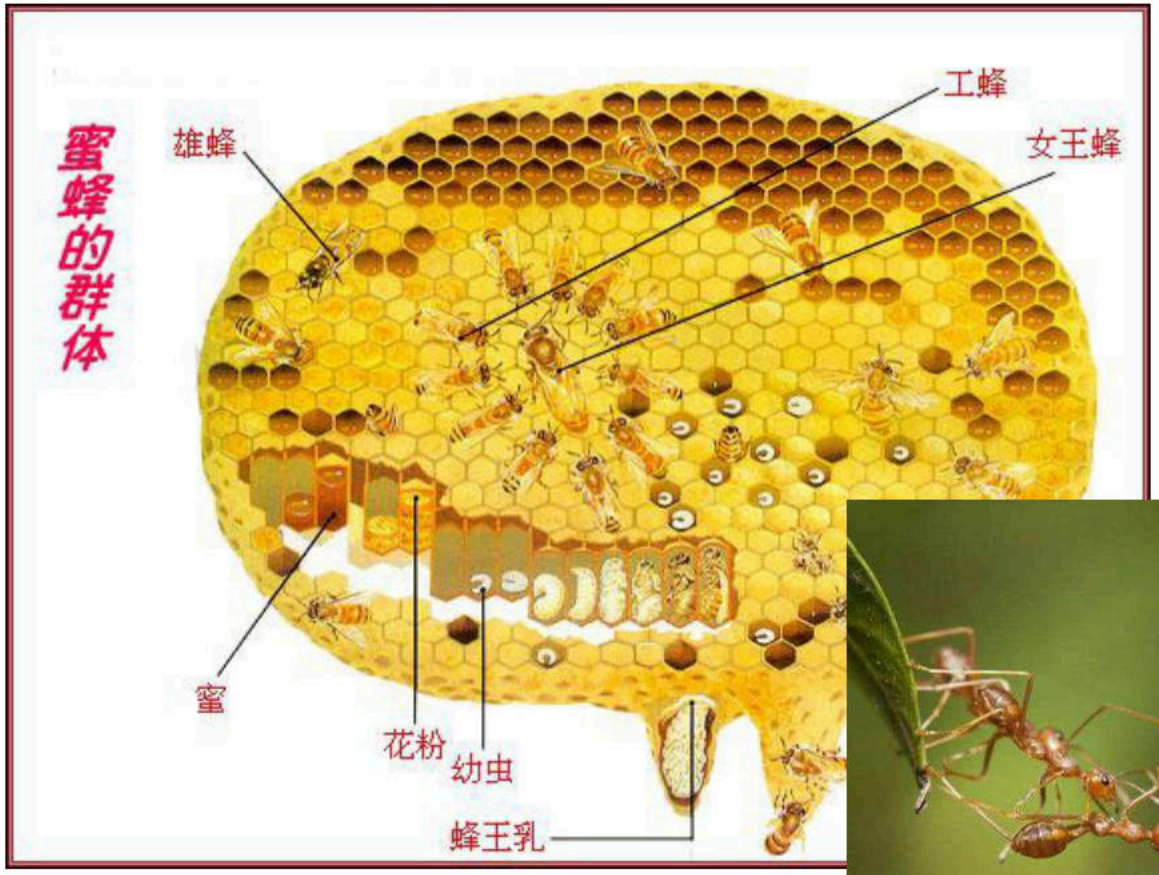
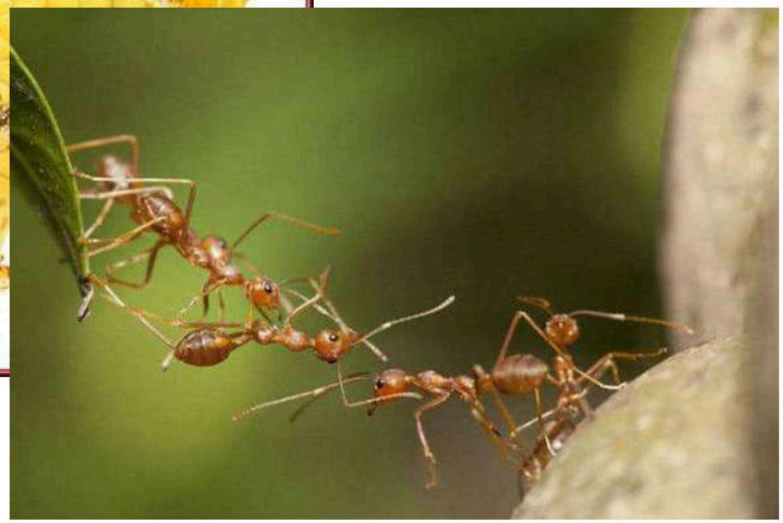
高级人工智能 群智能算法

Advanced Artificial Intelligence

Swarm intelligence algorithms

生物的智能行为





生物的智能行为

生物的智能行为

- 蜂巢，结构庞大、复杂而精美；蜜蜂协作觅食
- 蚂蚁完成觅食、清扫、搬运等高效工作
- 大雁在高速运动中保持和变换优美有序的队形

- 特点：个体都很简单，并不具备高智能，看起来也没有集中的指挥控制，却能协同工作，依靠群体的能力，发挥超出个体的智能，使群体表现出复杂而有序的行为

生物的智能行为及其算法

群：一组相互之间可以进行直接或间接通信的主体

群体智能：无智能或简单智能的主体通过任何形式的聚集、 协作而表现出智能行为的特性

群体智能算法：任何一种由昆虫群体或其它动物社会行为机制激发设计出的算法或分布式解决问题的策略

群智能算法的流程及含义

基本流程：种群初始化 → 个体更新 → 评价个体（适应度计算）→ 群体更新

初始化种群

- 问题解形式的确定（状态空间表示形式）
- 算法参数的选取：种群规模，迭代次数，控制参数
- 评估函数的设计：评价个体的优劣程度

群智能算法的流程及含义

个体更新

- 依靠自身的能力进行局部搜索：如交叉、变异
- 受到其它个体的影响来改善自我

群体更新

- 通过个体更新实现
- 通过子群更新实现
- 通过选择机制实现

群智能算法的分类

仿生过程类算法

- 模仿生物种群进化发展的过程
- 遗传算法；进化规划算法；进化策略算法；差分进化算法；人工免疫算法；量子进化算法等

仿生行为类算法

- 模仿生物种群的社会行为模式和协同合作机制
- 模仿觅食模式：粒子群（鸟群）、人工鱼群、混合蛙跳、蜂群、猫群、蚁群、细菌觅食算法等
- 共性：通过模仿指导个体移动位置趋向食物源
- 个性：不同的搜索方式

群智能算法的特点

后续学习时，大家要思考这些特点在不同算法中的表现形式

- 渐进式寻优
- 体现“优胜劣汰，适者生存”的自然选择规律
- 有指导的随机搜索
- 并行式搜索
- 较好的全局寻优能力
- 通用性强，易与其它算法结合
- 较强的鲁棒性（通俗而言，算法对输入数据的小扰动不敏感，亦即输入数据的小扰动不会对算法结果产生大的影响。）

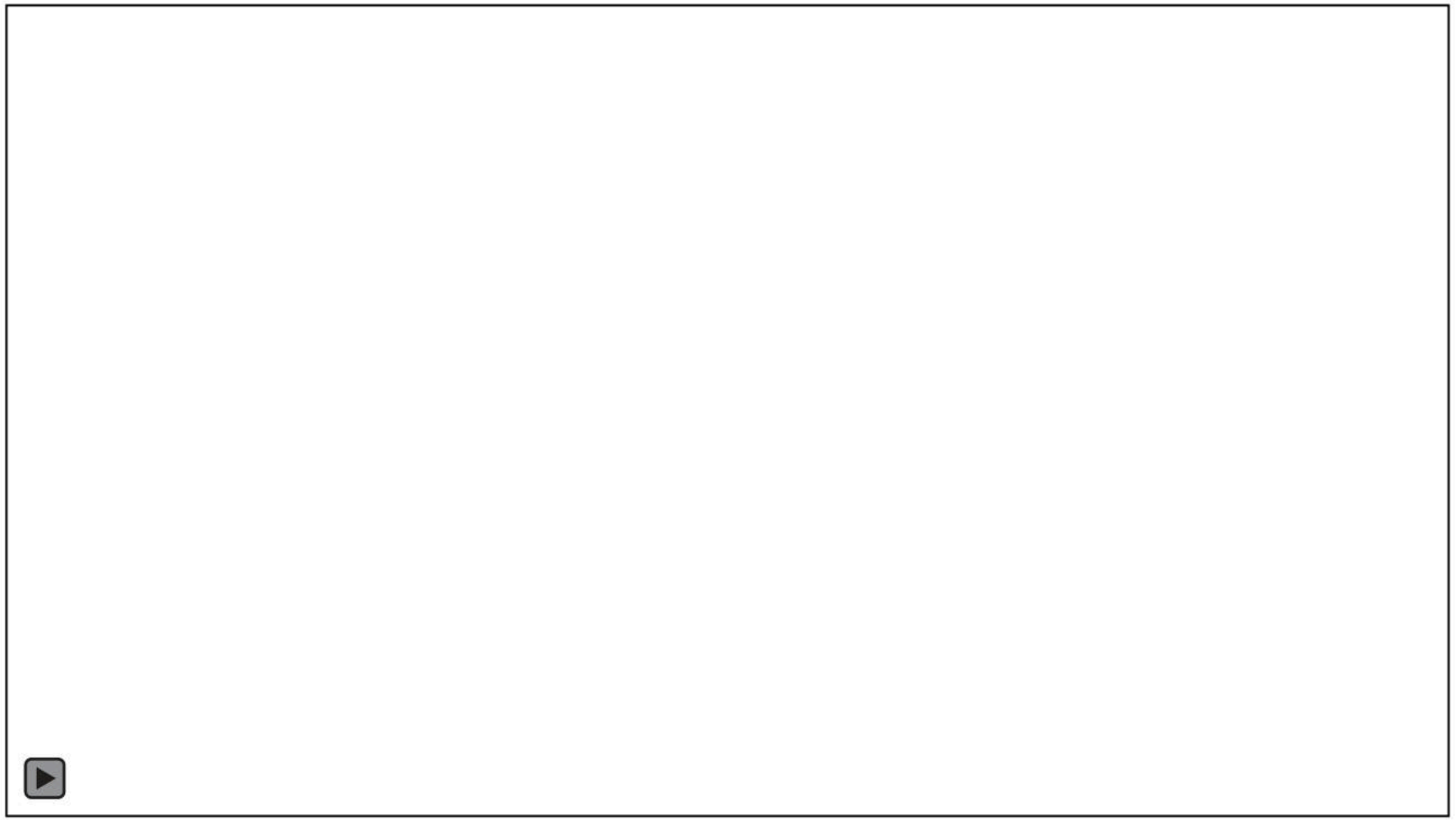
鲁棒性对不同的问题有不同层面的解释。比如一个神经网络模型，其鲁棒性还包含模型结构的容错性等……

遗传算法概述

（Genetic Algorithms,GA）

物竞天择，适者生存（survival of the fittest)

- 1975年，美国Michigan大学的Holland教授提出
- 起源于生物的遗传进化
- 80年代得到了广泛应用与研究
- 1990年，D.J.Goldberg 博士出版著作对遗传算法作了系统阐述，确立了现代遗传算法的基础，特别是描述了各类遗传参数的确定以及模拟计算方面的问题



遗传算法的基本思想

- 一种概率搜索算法：将问题状态编码成数串（称为染色体），从多个数串（初始群）开始，通过模拟生物遗传过程染色体中基因变化（3个算子）的特点，实现代与代之间的更迭，逼近最优解
- 通过随机但有指导的信息交换来传承和扩散那些适应性好的染色体
- 采用适应值函数的形式对产生的每个染色体进行评价，使适应性好的染色体有更多的繁殖机会

基本流程：

种群初始化→评价个体（适应度计算）→ 个体更新

→群体更新

基本遗传算子

- 选择： 从当前种群中选出优良的个体，使之有机会作为父代繁殖下一代的操作对象，称作选择运算（selection），或复制运算（reproduction）

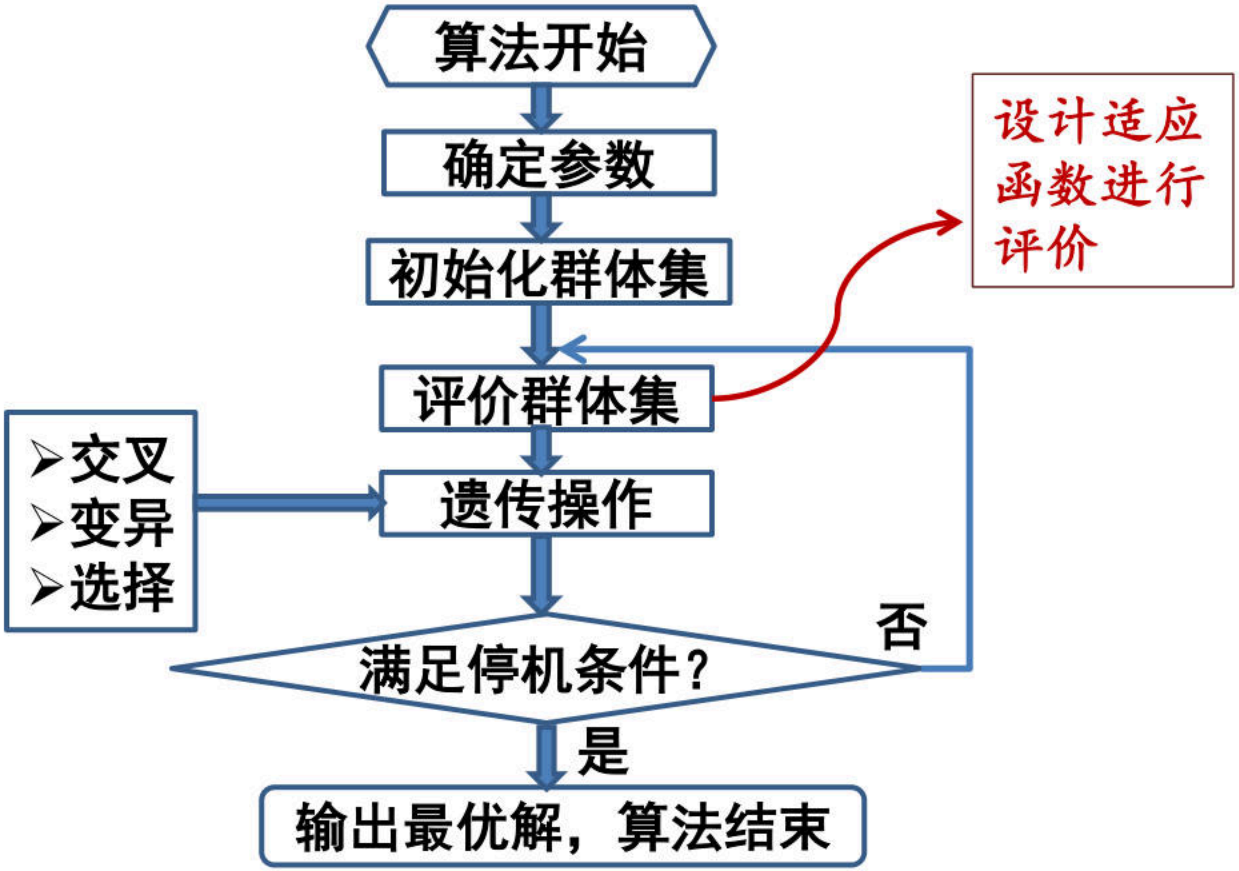
- 交叉： 对两个个体进行部分基因互换，产生两个新个体的操作，称为交叉或杂交运算（crossover）

- 变异： 对个体的某些基因进行改变，产生新个体的操作，称作变异运算（mutation）

遗传算法的常用术语

生物进化中的用语	遗传算法中的含义或作用
染色体（个体）	问题的一个解（编码表示）
群体	选定的一组解
基因	编码的位元素
环境适应性	适应函数来衡量
交配	交叉算子：两组编码位串交叉互换
变异	变异算子：编码中的位串变化
早熟	尚未找到全局最优解，而算法过早收敛

遗传算法流程图


算法开始
确定参数
初始化群体集
评价群体集
设计适应函数进行评价
交叉
变异
选择
遗传操作
满足停机条件？
否
是
输出最优解，算法结束

遗传算法构成要素

染色体编码

- 二进制编码：个体基因为{0,1}
- 符号编码：个体基因为符号，如{A,B,C…}，{1,2,3…}
- 浮点数编码：个体基因为浮点数

适应度函数

- 用于评价个体的优劣
- 适应度大小决定了个体繁衍的概率
- 可能出现某些个体适应度远高于平均值而导致早熟

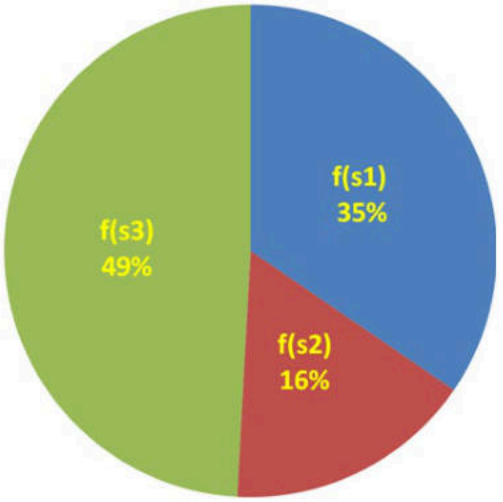
后续学习完遗传算法的过程后，
请解释为什么会发生这种情况

遗传算法的选择算子-1

选择算子：用来确定从上代群体中保留个体的方式，各代的个数与初始种群的个数N都是相同的

- 精英策略：上代适应度最好的个体直接保留到下一代（不单独使用）
- 赌轮选择法：个体被选中的概率正比于其适应度
- 确定选择法：建立选择次数期望值，以此为依据进行确定性选择
- 排序选择法：群体中的个体按适应度大小排序，将事先设计好的概率表分配给各个体

遗传算法的选择算子–2



选择算子——赌轮选择法

- 个体被选中的概率正比于其适应度

x_i：第 i 个个体　　$i = 1,2,\cdots,N$

N：群体规模

$F(x_i)$：第 i 个个体的适应度

$p(x_i)$：第 i 个个体被选择留到下一代的概率

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

以概率 $p(x_i)$ 选中 x_i

```
/* once of roulette wheel selection
 * 输出参数：
 * 选中的染色体
*/
 procedure RWS
1   m←0;
2   r←Random(0,1); //0至1的随机数
3   for i=1 to N
4     m←m+Pi;
5     if r <= m
6       return i;
7     end if
8   end for
 end procedure
```

遗传算法的选择算子-3

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

选择算子——确定选择法

1）对于规模为N的群体，首先计算选择概率为$p(x_i)$的染色体x_i被选中次数的期望值$e(x_i)=p(x_i)N$

2）对于群体中的每一个x_i，首先选择$\lfloor e(x_i)\rfloor$次；共得到$\sum_{i=1}^{N}\lfloor e(x_i)\rfloor$个下一代染色体

向下取整

3）按照$e(x_i)-\lfloor e(x_i)\rfloor$从大到小对染色体排序，依次取出$N-\sum_{i=1}^{N}\lfloor e(x_i)\rfloor$个染色体，得到总共$N$个染色体。

遗传算法的选择算子-4

N=3时确定选择法示例

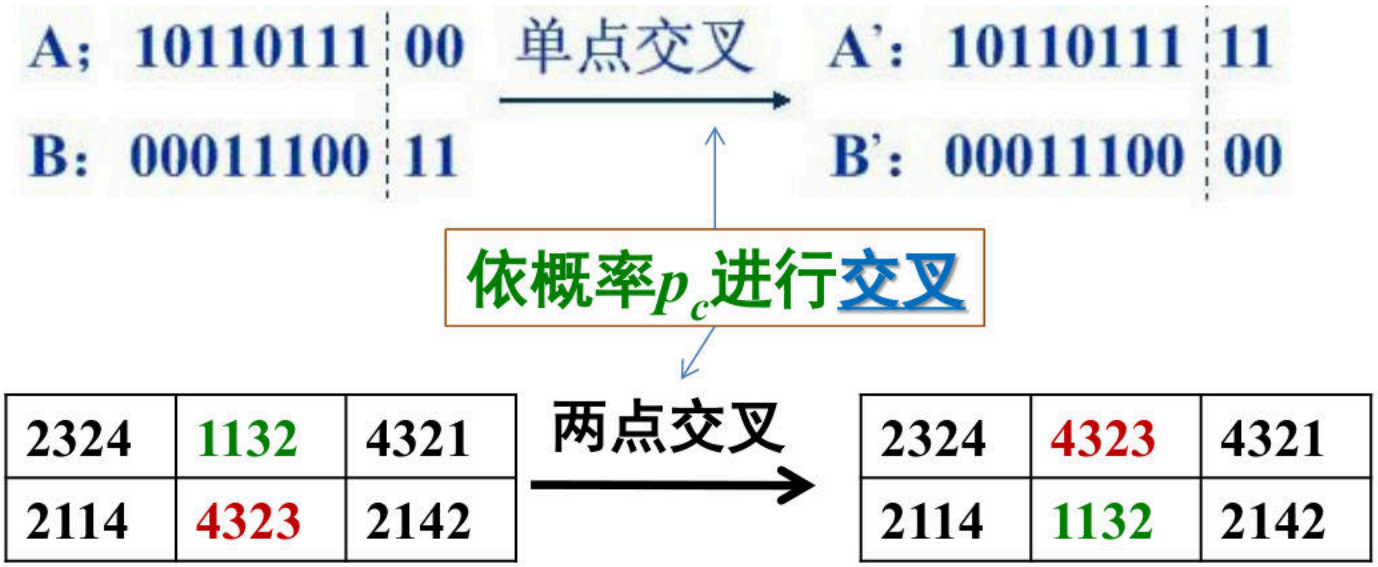
	$P(x_i)$	$e(x_i)$	$\lfloor e(x_i) \rfloor$	$e(x_i)-\lfloor e(x_i) \rfloor$	选中次数
x_1	0.35	0.35×3 =1.15	1	0.15	1+0=1
x_2	0.15	0.15×3 =0.45	0	0.45	0+0=0
x_3	0.5	0.5×3 =1.5	1	0.5	1+1=2

Σ=N=3

遗传算法的交叉算子

交叉算子

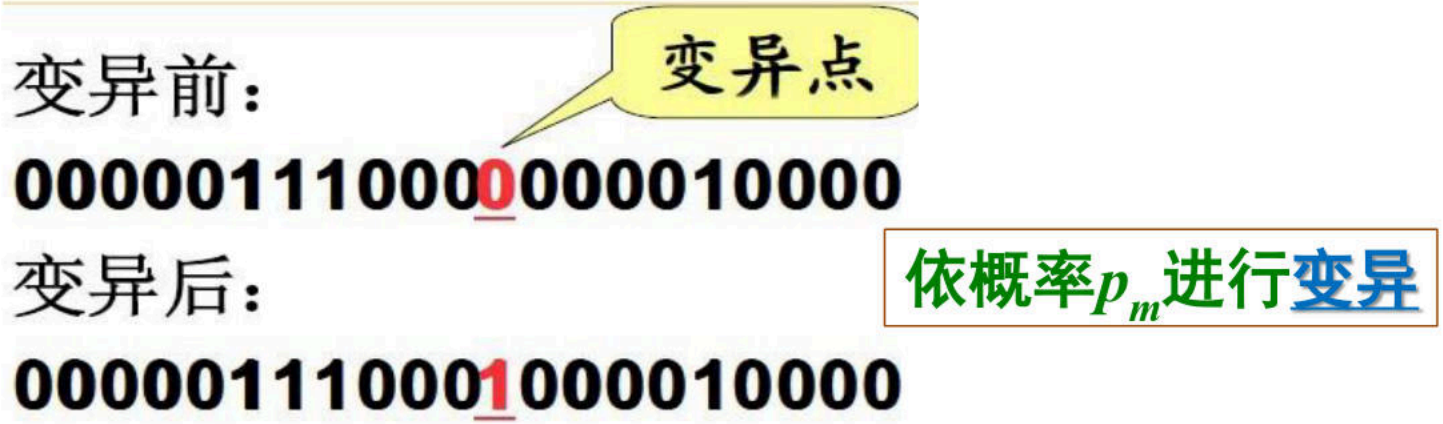
■模仿自然界有性繁殖的基因重组过程，将上代优良基因遗传给下一代



遗传算法的变异算子

变异算子

- 模仿自然界有性繁殖的基因突变现象，保持多样性



遗传算法控制参数选择-1

位串长度L（染色体长度）

- 影响算法的计算量和交叉变异操作的效果

- 与优化问题密切相关，一般由问题定义的解的形式和选择的编码方法决定

- 二进制编码：L根据解的值域和精度要求选择
  原问题变量$x \in [a, b]$，码长n，编码精度为 $(b-a)/(2^n-1)$

- 浮点数编码：L一般与问题定义的解的维数D相同。

- 除了定长染色体编码，Goldberg等人还提出了一种变长度染色体遗传算法Messy GA。

遗传算法控制参数选择-2

- **群体规模N**
- 影响算法的搜索能力和运行效率
- N较大则采样范围广，可防早熟，但计算量大；N太小，计算量小但多样性不足，可能导致早熟。
- 建议N=20~200，为便于交叉，通常取为偶数

遗传算法控制参数选择-3

交叉概率P_c

- P_c较高则结构变化导入较快，优良基因丢失的可能性上升
- P_c过低可能导致搜索阻滞
- 一般$P_c \geq 0.6$
- 也可采用自适应方法在算法运行过程中调整交配概率。

自适应变异：
纵向—随着迭代次数的变化而变
横向—不同的个体变异概率不同

变异概率P_m

- P_m过高则趋向于随机搜索
- P_m过低可能导致基因位过早丢失信息，多样性难恢复
- 一般取0.005~0.05，与字长L相关
- 也可采用自适应方法在算法运行过程中调整P_m值

如何设计自适应方案，依据是什么？

遗传算法实现细节及说明-1

依$p(x_i)$，结合精英策略从群体中选择N个染色体得到种群，规模依然为N

方法1:

1)对N个个体$x(i), i = 1,2,\cdots,N$，执行3个算子后最终产生$N-1$个新个体$y(j), j = 1,2,\cdots,N-1$;

2)$y(N) \leftarrow x(\arg \underset{1\le i\le N}{Max}\{F(x_i)\})$;精英策略

方法2:

1)对N个个体$x(i), i = 1,2,\cdots,N$执行3个算子后最终产生N个新个体$y(j), j = 1,2,\cdots,N$;

2)$x(\arg \underset{1\le i\le N}{Max}\{F(x_i)\})$替代$y(j)$中适应值最小的;精英策略

1)精英策略还有其他实现形式，如多精英保留、增扩群体规模等

2)精英策略是保证遗传算法收敛的不可或缺的步骤

遗传算法实现细节及说明-2

从种群中选择染色体对，依概率p_c进行交叉，其子代进入新群体；种群中未进行交叉的染色体，直接复制到新群体中，规模依然为N

1)在$y(j), j = 1,2,\cdots, N$中随机选出2个个体

2)依概率P_C,轮盘赌方式决定是否交叉

3)若交叉，用随机数选出一个交叉位,2个个体交叉后成为下一代新个体；若不交叉,2个个体直接复制到下一代

4)在余下的$y(j)$中重复上述过程，直至遍历完成

遗传算法实现细节及说明-3

对群体中的每个染色体x_i的每个位，依概率p_m进行变异，用 变异后的染色体代替原染色体

1)对每一个个体，每一个位，依概率P_m用轮盘赌方式决定是否发生变异

2)一次迭代中变异与否的判断次数 $= L \cdot N$，约为$(20 \sim 200) \times L$

3)遗传算法的很多应用场景L的值很大。如用于神经网络的训练时，一个个体对应整个网络的一组可学习的权值，以*LeNet*为例，大约60000个，编码后L的长度更大

4)*LeNet*仅有7层，现今深度神经网络的层数甚至达到1000多层。为减少计算量，有时会对变异操作进行简化

设计变异操作简化方案
并给出依据和结果分析

遗传算法举例—求函数的极大值

例：求函数 $f(x)=x^2$ 的最大值，$x \in [0, 31]$，整数

编码：二进制

适应函数：$f(x)$

选择算子：确定选择法

位串长度：L=5（染色体长度）

用5位二进制数表示

00000: x＝0，10101: x＝21，11111: x＝31

群体规模：N=4

交叉概率：P_c =1.0

变异概率：P_m =0.01

初始群体：（13,24,8,19）

01101，11000，01000，10011

遗传算法举例—求函数的极大值

第0代情况表（选择运算）

序号	群体	适应值	选择概率（%）	期望次数	选中次数
1	01101	169	14.44	4×14.44% ≈0.58	0+1=1
2	11000	576	49.23	4×49.23% ≈ 1.97	1+1=2
3	01000	64	5.47	4× 5.47% ≈ 0.22	0+0=0
4	10011	361	30.85	4× 30.85% ≈ 1.23	1+0=1

遗传算法举例—求函数的极大值

第0代种群的交叉情况（无变异发生）

序号	种群	交叉对象	交叉位	子代	适应值
1	01101	2	4	01100	144
2	11000	1	4	11001	625
3	11000	4	2	11011	729
4	10011	3	2	10000	256

遗传算法举例—求函数的极大值

第1代情况表（选择运算）

序号	群体	适应值	选择概率（%）	期望次数	选中次数
1	01100	144	8.21	0.33	0+0=0
2	11001	625	35.62	1.42	1+0=1
3	11011	729	41.56	1.66	1+1=2
4	10000	256	14.60	0.58	0+1=1

遗传算法举例—求函数的极大值

第1代种群的交叉情况（无变异则发生早熟）

序号	种群	交叉对象	交叉位	子代	适应值
1	11001	2	3	11011	729
2	11011	1	3	11001	625
3	11011	4	1	10000	256
4	10000	3	1	11011	729

遗传算法举例—求函数的极大值

第1代种群的变异情况

序号	种群	是否变异	变异位	新群体	适应值
1	11011	N		11011	729
2	11001	Y	3	11101	841
3	10000	N		10000	256
4	11011	N		11011	729

遗传算法举例—求函数的极大值

第2代情况表（选择运算）

序号	群体	适应值	选择概率（%）	期望次数	选中次数
1	11011	729	28.53	1.14	1+0=1
2	11101	841	32.92	1.31	1+0=1
3	10000	256	10.02	0.40	0+1=1
4	11011	729	28.53	1.14	1+0=1

2025/4/15
37

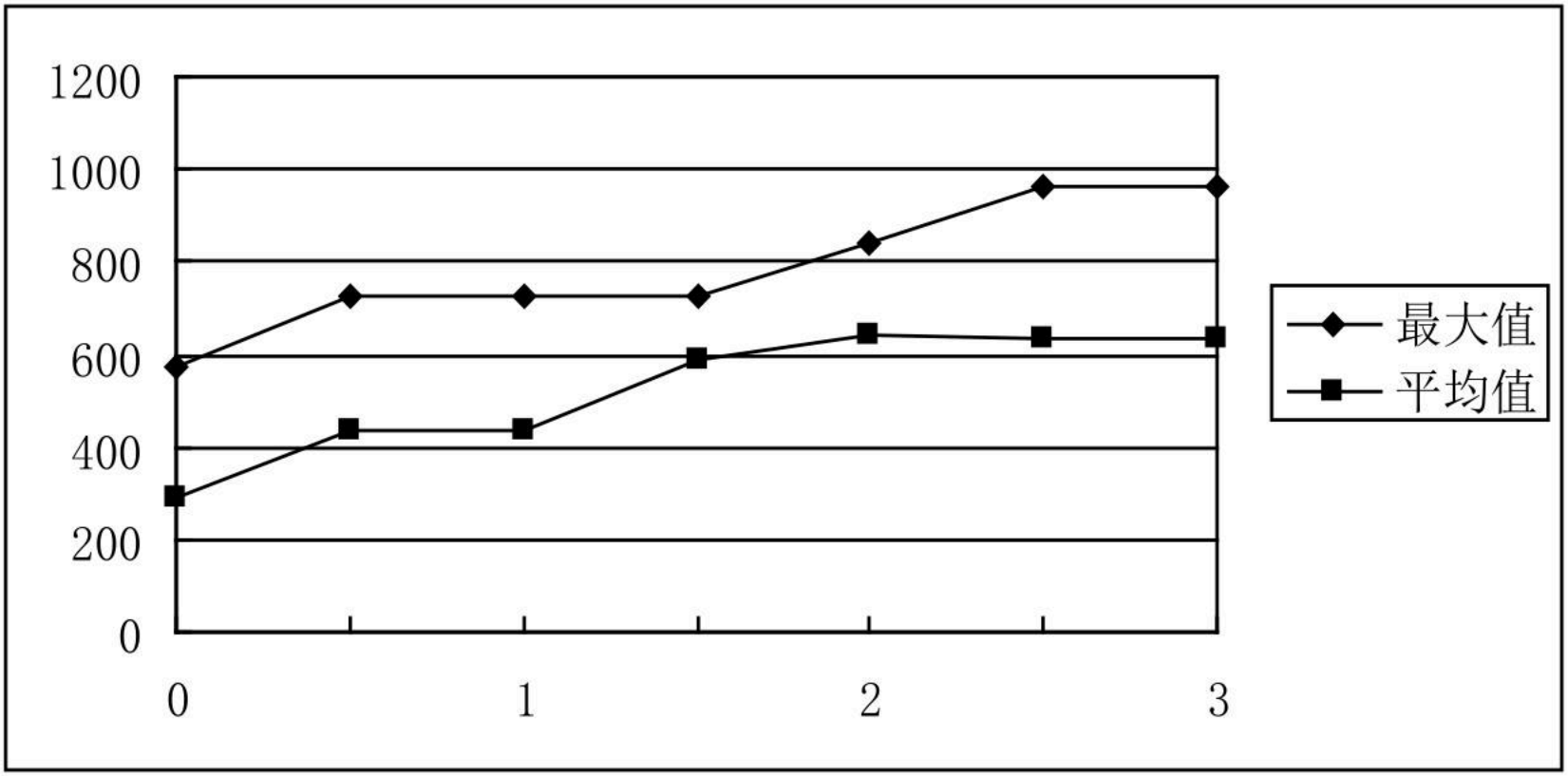
遗传算法举例—求函数的极大值

第2代种群的交叉情况

序号	种群	交叉对象	交叉位	子代	适应值
1	11011	2	3	11001	625
2	11101	1	3	11111	961
3	10000	4	2	10011	361
4	11011	3	2	11000	576

遗传算法举例—求函数的极大值

最大适应值、平均适应值进化曲线



4.2.2 应用举例

- 例4.1 $y = f(x_1, x_2, x_3, x_4) = \dfrac{1}{x_1^2 + x_2^2 + x_3^2 + x_4^2 + 1}$

$-5 \le x_1, x_2, x_3, x_4 \le 5$ ，用遗传算法求解y的最大值。

运行步骤

步骤6：重新评价各染色体适应值，更新 $Best$

计算每个染色体的适应值：

$Eval(C_1) = \ldots = 0.0558585$

$Eval(C_2) = \ldots = 0.0230112$

$Eval(C_3) = \ldots = 0.0210019$

$Eval(C_4) = \ldots = 0.0789962$

$Eval(C_5) = \ldots = 0.0245465$

因为 $Max(0.0558585, 0.0230112, 0.0210019, 0.0789962, 0.0245465) = 0.0789962$，所以更新 $Best$：

$Best = \ldots$

步骤5：变异

对每个染色体的每个基因随机生成[0,1]的随机数，如果小于0.1，则改变基因的值，否则不改变。

步骤1：初始化

假设种群大小为5，使用浮点数编码方式构造染色体，即每个染色体以形式 (x_1, x_2, x_3, x_4) 表示

$Eval(C_1) = f(\ldots) = 0.0373603$

$Eval(C_2) = f(\ldots, -0.9008, -4.3712, \ldots, 3.7535) = 0.0255988$

$Eval(C_3) = f(\ldots, -0.1503, 0.0023) = 0.0448529$

$Eval(C_4) = f(\ldots, -0.9008, -4.3712) = 0.0238214$

$Eval(C_5) = f(0.2073, 2.9932, -4.0802, 1.8794) = 0.0331319$

因此 $Best = C_3$，$Eval(Best) = 0.0448529$

步骤3：选择

采用轮盘赌选择算法，计算群体适应值总和为0.0373603+0.0255988+0.0448529+0.0238214+0.0331319=0.164765

$C_1' = (1.0152, -3.9811, -2.6638, 3.7535)$

$C_2' = (4.0589, 2.1904, -0.1503, 0.0023)$

$C_3' = (1.0152, -3.9811, -2.6638, 1.8794)$

$C_4' = (1.0152, -3.9811, -2.6638, 3.7535)$

$C_5' = (0.2073, 2.9932, -4.0802, 1.8794)$

步骤4：交配

假设交配概率为0.88，下面是对每个染色体生成的[0,1]的随机数，决定染色体是否参加交配。

$C_2'' = (4.0589, 2.1904, -2.6638, 3.7535)$

$C_3'' = (1.0152, -3.9811, -2.6638, 3.7535)$

$C_4'' = (1.0152, -3.9811, -2.6638, 1.8794)$

$C_5'' = (0.2073, 2.9932, -4.0802, 3.7535)$

精英策略？

步骤1：初始化。
假设群体规模为5；使用浮点数编码方式构造染色体，即每个染色体以形式(x_1, x_2, x_3, x_4)表示。
初始化群体的染色体，得到
C_1 = (-2.1351 , 2.0917 , -0.1327 , -4.1006)
C_2 = (1.0152 , -3.9811 , -2.6638 , 3.7535)
C_3 = (4.0589 , 2.1904 , -0.1503 , 0.0023)
C_4 = (-3.4098 , -3.0714 , -0.9008, -4.3712)
C_5 = (0.2073 , 2.9932 , -4.0802 , 1.8794)

步骤2：适应值评价
选择评估函数

$$Eval(C) = y = f(x_1, x_2, x_3, x_4) = \frac{1}{x_1^2 + x_2^2 + x_3^2 + x_4^2 + 1}$$

计算每个染色体的适应值如下：
$Eval(C_1) = f$ (-2.1351 , 2.0917 , -0.1327 , -4.1006) = 0.0373603
$Eval(C_2) = f$ (1.0152 , -3.9811 , -2.6638 , 3.7535) = 0.0255988
$Eval(C_3) = f$ (4.0589 , 2.1904 , -0.1503 , 0.0023) = 0.0448529
$Eval(C_4) = f$ (-3.4098 , -3.0714 , -0.9008, -4.3712) = 0.0238214
$Eval(C_5) = f$ (0.2073 , 2.9932 , -4.0802 , 1.8794) = 0.0331319
因此 $Best = C_3$ ， $Eval(Best)$ = 0.0448529

步骤3：选择
采用轮盘赌选择算法，计算群体适应值总和为0.0373603+0.0255988+0.0448529+0.0238214+0.0331319 = 0.164765
分别计算每个染色体适应值同群体适应值总和的比：
C_1：0. 226749
C_2：0. 155365
C_3：0. 272223
C_4：0. 144578
C_5：0. 201085
下面是5次选择产生的[0,1]的随机数和选中的染色体：
1： 0.278756 C_2
2： 0.604389 C_3
3： 0.230964 C_2
4： 0.376263 C_2
5： 0.858791 C_5
因此得到种群为
C_1' = (1.0152 , -3.9811 , -2.6638 , 3.7535)
C_2' = (4.0589 , 2.1904 , -0.1503 , 0.0023)
C_3' = (1.0152 , -3.9811 , -2.6638 , 3.7535)
C_4' = (1.0152 , -3.9811 , -2.6638 , 3.7535)
C_5' = (0.2073 , 2.9932 , -4.0802 , 1.8794)

步骤4：交配
假设交配概率为0.88。下面是对每个染色体生成的[0，1]的随机数，决定染色体是否参加交配。
C_1' 0.341044 < 0.88 参加交配
C_2' 0.6137971< 0.88 参加交配
C_3' 0.963042> 0.88 不参加交配
C_4' 0.347545< 0.88 参加交配
C_5' 0.593677 < 0.88 参加交配
即C_1'和C_2'、C_4'和C_5'进行交配，每对染色体交配时随机生成0至3之间的自然数作为交配位。以下是各对染色体的交配位和得到的子代染色体。
1. C_1' = (1.0152 , -3.9811 , -2.6638 , 3.7535)
和C_2' = (4.0589 , 2.1904 , -0.1503 , 0.0023)
交配位为1；子代染色体为：
C_1'' = (1.0152 , -3.9811 , -0.1503 , 0.0023)
C_2'' = (4.0589 , 2.1904 , -2.6638 , 3.7535)
2. C_4' = (1.0152 , -3.9811 , -2.6638 , 3.7535)
和C_5' = (0.2073 , 2.9932 , -4.0802 , 1.8794)
交配位为2；子代染色体为：
C_4'' = (1.0152 , -3.9811 , -2.6638 , 1.8794)
C_5'' = (0.2073 , 2.9932 , -4.0802 , 3.7535)
故交配后的新种群为：
C_1'' = (1.0152 , -3.9811 , -0.1503 , 0.0023)
C_2'' = (4.0589 , 2.1904 , -2.6638 , 3.7535)
C_3'' = (1.0152 , -3.9811 , -2.6638 , 3.7535)
C_4'' = (1.0152 , -3.9811 , -2.6638 , 1.8794)
C_5'' = (0.2073 , 2.9932 , -4.0802 , 3.7535)

步骤5：变异
假设变异概率为0.1。对于每个染色体的每个基因随机生成[0,1]的随机数，若该随机数小于0.1，则改变基因的值，否则不改变基因的值。以下是发生变异的染色体和基因改变的过程。

C_3'' = (***1.0152*** , -3.9811 , -2.6638 , 3.7535) →
C_3''' = (***3.0953*** , -3.9811 , -2.6638 , 3.7535)
C_4'' = (1.0152 , ***-3.9811*** , -2.6638 , 1.8794) →
C_4''' = (1.0152 , ***0.0153*** , -2.6638 , 1.8794)

故得到的新群体为
C_1''' = (1.0152 , -3.9811 , -0.1503 , 0.0023)
C_2''' = (4.0589 , 2.1904 , -2.6638 , 3.7535)
C_3''' = (3.0953 , -3.9811 , -2.6638 , 3.7535)
C_4''' = (1.0152 , 0.0153 , -2.6638 , 1.8794)
C_5''' = (0.2073 , 2.9932 , -4.0802 , 3.7535)

步骤6：重新评价染色体适应值，更新*Best*
计算每个染色体的适应值如下：
$Eval(C_1''') = f$ (1.0152 , -3.9811 , -0.1503 , 0.0023) = 0.0558585
$Eval(C_2''') = f$ (4.0589 , 2.1904 , -2.6638 , 3.7535) = 0.0230112
$Eval(C_3''') = f$ (3.0953 , -3.9811 , -2.6638 , 3.7535) = 0.0210019
$Eval(C_4''') = f$ (1.0152 , 0.0153 , -2.6638 , 1.8794) = 0.0789962
$Eval(C_5''') = f$ (0.2073 , 2.9932 , -4.0802 , 3.7535) = 0.0245465
因为*Max*(0.0558585，0.0230112，0.0210019，0.0789962，0.0245465)= 0.0789962 > $Eval(Best)$,故更新*Best*：
$Best = C_4'''$， $Eval(Best)$ = 0.0789962。

步骤7：判断结束。
如果满足算法终止条件,则输出找到的最优解*Best*并退出程序。否则返回步骤3继续执行 。

- 上面只是给出了遗传算法的基本框架，不同的编码方式和交叉、变异方式会产生出不同的遗传算法；

遗传算法算子特点总结

- 选择：优胜劣汰

—为种群提供进化的方向；

- 交叉：优势组合

—将散布于不同个体的、好的局部模式组合在一起；

- 变异：寻找新模式

—种群向外扩展的触角，好的变异保留；

遗传算法的特点

- 随机搜索算法，适用于数值求解具有多参数、多变量、多目标等复杂的最优化问题；
- 对指标函数无特殊要求，不要求诸如连续性、导数存在、单峰值假设等，甚至无需显式的写出指标函数；
- 经编码后，遗传算法除了适应值的计算外，几乎不需要任何与问题有关的知识；无需使用者对问题有深入的了解和求解技巧，是一个较通用的优化算法；
- 具有天然的并行性，适用于并行求解。

  —运算过程可并行实现；

  —隐含的模式并行机制。
- 求得的是近似解
- 编码方式有时不方便

模式

指群体中编码的某些位置具有相似结构的染色体集合

模式的阶

指模式中具有确定取值的基因个数

模式的定义长度

指模式中第一个具有确定取值的基因到最后一个具有确定取值的基因的距离

模式

- 字串中用通配符“*”，表示该位可以为0或1；模式表示的字母表{0，1，*}；
- 因此用三元素字母表{0，1，*}可构造出任意一种模式；
- 一个模式与一个特定位串相匹配是指：该模式中的1与位串中的1相匹配，模式中的0与位串的0相匹配，模式中的“*”可以匹配位串中的0或1。

模式

- 例如：
  - 模式00*00匹配了：{00100, 00000}
  - 模式*111*可以和{01110，01111，11110，11111}中的任何一个位串匹配，即与长度为5中间三位为“1”的四个位串匹配；
  - 模式0*1**则匹配了长度为5、第一位为0、第三位为1的8个位串{00100，00101，00110，00111，01100，01101，01110，01111}

阶数2　　定义长度2

模式定理的结论

- 结论：定义长度短的、确定位数少的、平均适配值高的模式数量将随着代数的增加呈指数增长；

- 上述结论称为模式理论（*Schema Theory*）或遗传算法的基本定理（*The Fundamental of Genetic Algorithms*）；

- 根据模式理论，随着遗传算法一代一代的进行，那些定义长度短的、位数少的、高适配值的模式将越来越多，因而可期望最后得到的位串（即这些模式的组合）的性能越来越得到改善，并最终趋向全局的最优。

模式定理的分析

- 在产生新代的过程中，遗传算法处理的模式数的数量级为$O(n^3)$，这是遗传算法的一个重要特征；
- 尽管只完成了$O(n)$的计算量，但处理的模式数却为$O(n^3)$。这就是遗传算法隐含的计算上的并行机制；
- 遗传算法通过定义长度短、确定位数少、适配值高的模式反复抽样、组合来寻找最佳点。

遗传算法收敛性结论

如果在代的进化过程中，遗传算法每次保留到目前为止的最好解（精英策略），且算法以交叉和变异为其随机化操作，则对于一个全局最优化问题，当执行次数趋于无穷时，遗传算法找到全局最优解的概率为1。

与GA相关的重要学术期刊与国际会议

⑩ 重要学术期刊

- Evolutionary Computation
- IEEE Transactions on Evolutionary Computation
- ……

⑩ 重要国际会议

- International Conference on Genetic Algorithm
- ACM Genetic and Evolutionary Computation Conference
- Workshop on Foundations of Genetic Algorithms and Classifier Systems
- Genetic Programming Conference
- International Workshop on Artificial Life
- ……

整数编码遗传算法问题—TSP

TSP的一种二进制编码：

- ABCD4个城市，考虑状态空间的二进制模型化方法
- 若用矩阵表示TSP问题的一个可能解，如下图，按行将可能解用一个n×n的二进制串表示：

	1	2	3	4
A	0	1	0	0
B	1	0	0	0
C	0	0	0	1
D	0	0	1	0

0100100000010010 （n=4）

行：城市
列：访问顺序
路径：BADCB

整数编码遗传算法问题—TSP

二进制编码存在的问题：

- 编码数远超原问题的状态数，计算量大

  位串长度：$n \times n$；

  状态空间（编码个数）：$2^{n \times n}$

  对称的n城市TSP问题的可能解：$n!/2$

  例：$n=10$，编码个数为可能解个数的 7.0×10^{23}倍

- 可行解难判断、适应值函数难设计

可行解：该解是否遍历所有城市一遍且仅一遍

适应值函数：该解的路径长度

整数编码： 整数1~n的一个排列作为一个可行解，长度n

问题： 需要重新定义遗传操作……

一种整数编码遗传算法-交叉算子设计1

交叉算子—常规交叉法

保留父代序的关系进行交叉

- 随机选取一个交叉位，子代1交叉位之前的基因选自父代1交叉位之间的基因，交叉位之后的基因，从父代2中按顺序选取那些没有出现过的基因。

	交叉位		交叉位
父代1：	1 2 3 4 5	6 7 8	
父代2：	5 2 1 7 3	8 4 6	
子代1：	1 2 3 4 5	7 8 6	
子代2：	5 2 1 7 3	4 6 8	

一种整数编码遗传算法-交叉算子设计2

交叉算子—基于次序的交叉法

保留父代序的关系进行交叉

父代1:　1　2　3　4　5　6　7　8　9　10

父代2:　5　9　2　4　6　1　10　7　3　8

所选位置:　*　*　　*　　　*

1）父代1中与所选位置相对应的数字为：2、3、5、8。

2）从父代2中找出这些数字，并去除它们，有：

父代2: b　9　b　4　6　1　10　7　b　b

3）用2、3、5、8依次填入上述父代2的空位置b中，得到

子代1: 2　9　3　4　6　1　10　7　5　8

一种整数编码遗传算法-交叉算子设计3

交叉算子—基于位置的交叉法

保留父代序的关系进行交叉

- 先随机产生一组位置。对于这些位置上的基因，子代1从父代2中直接得到，子代1的其他位置的基因，按顺序从父代1中选取那些不相重的基因。子代2也类似处理。如：

父代1：1 2 3 4 5 6 7 8 9

父代2：5 9 2 4 6 1 7 3 8

* * * *

子代1：1 9 2 4 6 5 7 3 8

子代2：9 2 3 4 5 6 1 8 7

一种整数编码遗传算法-交叉算子设计4

交叉算子—基于部分映射的交叉法

- 对父代1和父代2，随机产生两个位置，两个父代在这两个位置之间的基因产生对应对，然后用这种对应对分别去替换两个父代的基因，从而产生两个子代。

父代1：2 6 4 3 8 1 5 7 9

父代2：8 5 1 7 6 2 4 3 9

子代1：1 8 4 7 6 2 5 3 9

子代2：6 5 2 3 8 1 4 7 9

一种整数编码遗传算法-变异算子设计

变异算子

改变序的关系

- 基于位置：随机产生两个变异位，将第二变异位上的基因移动到第一变异位之前。

  2 1 3 6 4 5 7　　2 4 1 3 6 5 7

- 基于次序：随机产生两个变异位，然后交换这两个变异位上的基因。

  2 1 3 6 4 5 7　　2 4 3 6 1 5 7

- 基于置乱：随机选取染色体上的一段，然后打乱在该段内的基因次序

  2 1 3 6 4 5 7　　2 4 6 3 1 5 7

逆序是一种特例

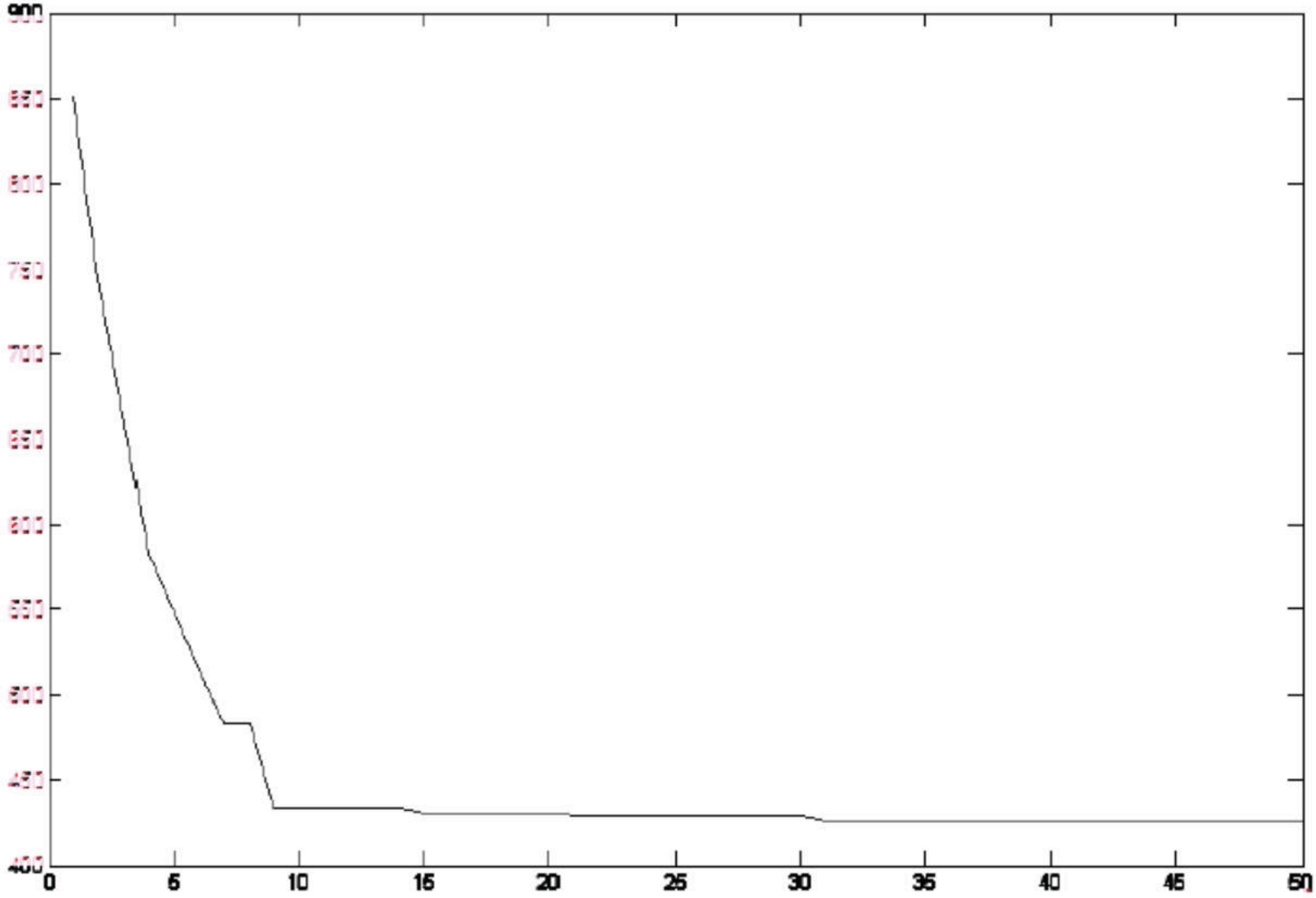
30 个城市的 TSP 问题

City = [64, 60; 68, 58; 83, 69; 87, 76; 74, 78;71,71; 58, 69; 54, 62; 51, 67; 37, 84; 41, 94; 2, 99; 7, 64; 22, 60; 25, 62; 18, 54; 87, 7; 91, 38; 83, 46; 71, 44; 4, 50; 13, 40; 18, 40; 24, 42; 25, 38; 41, 26; 45, 21; 44, 35; 58, 35; 62, 32];

参数设定为:
初始种群大小s=250
交叉概率=0.8
变异概率=0.02
最大遗传代数=50

50代后得到最短距离为：424.87；最优结果423.7601。
最佳图路径为：21,22, 23, 24, 25, 28, 26, 27, 29, 30, 17, 18, 19, 20, 2, 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 14, 16

30 城市最短路径长度收敛图



前10代下降快速，30代后基本无变动。

50 个城市的 TSP问题

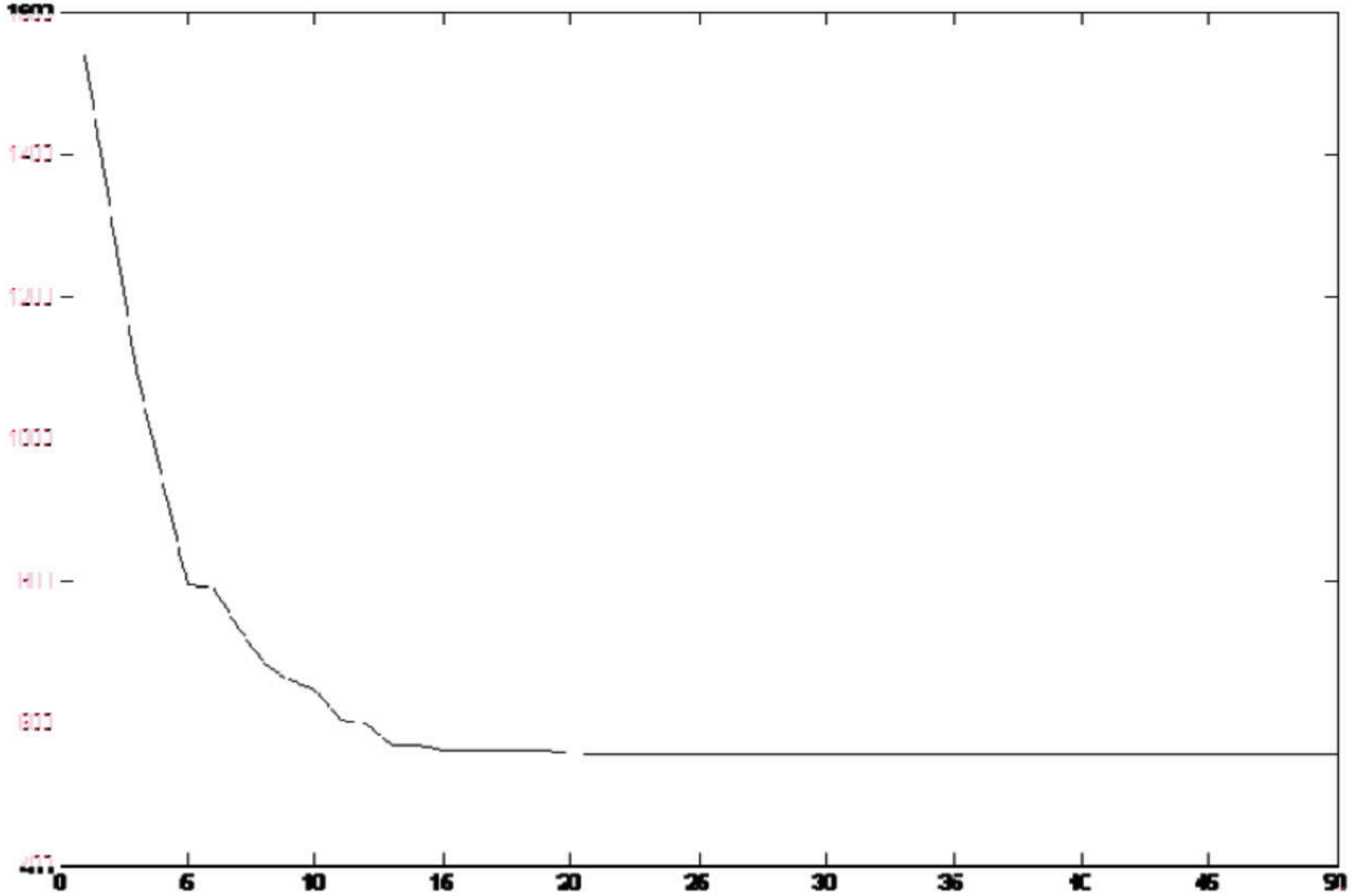
50 个城市的坐标：

x	*y*	*x*	*y*	*x*	*y*	*x*	*y*	*x*	*y*
10	75	74	98	14	18	15	70	20	20
36	9	91	16	26	80	37	91	39	2
91	78	92	75	49	13	63	54	29	90
54	53	63	73	24	94	46	42	87	46
8	51	95	16	5	92	66	8	62	87
78	51	21	84	66	43	52	28	70	92
73	14	4	23	48	26	61	34	19	74
23	56	41	29	13	33	70	13	32	10
15	40	36	74	11	53	91	55	81	84
26	60	56	15	47	65	36	60	73	90

参数设定为：
初始种群大小s=1000
交叉概率=0.8
变异概率=0.02
最大遗传代数=50

经过50 代后得到最短距离为：554.2。
最佳图路径为：50, 46, 11, 45, 14, 30, 40, 19, 22, 16, 43, 32, 24, 25,1, 31, 47, 10, 8, 29, 5, 9, 28, 17, 21, 41, 48, 2, 42, 23, 20, 35,38, 7, 12, 15, 37, 36, 27, 18, 34, 4, 33, 26, 6, 44, 39, 13, 3, 49

50 城市最短路径长度收敛图



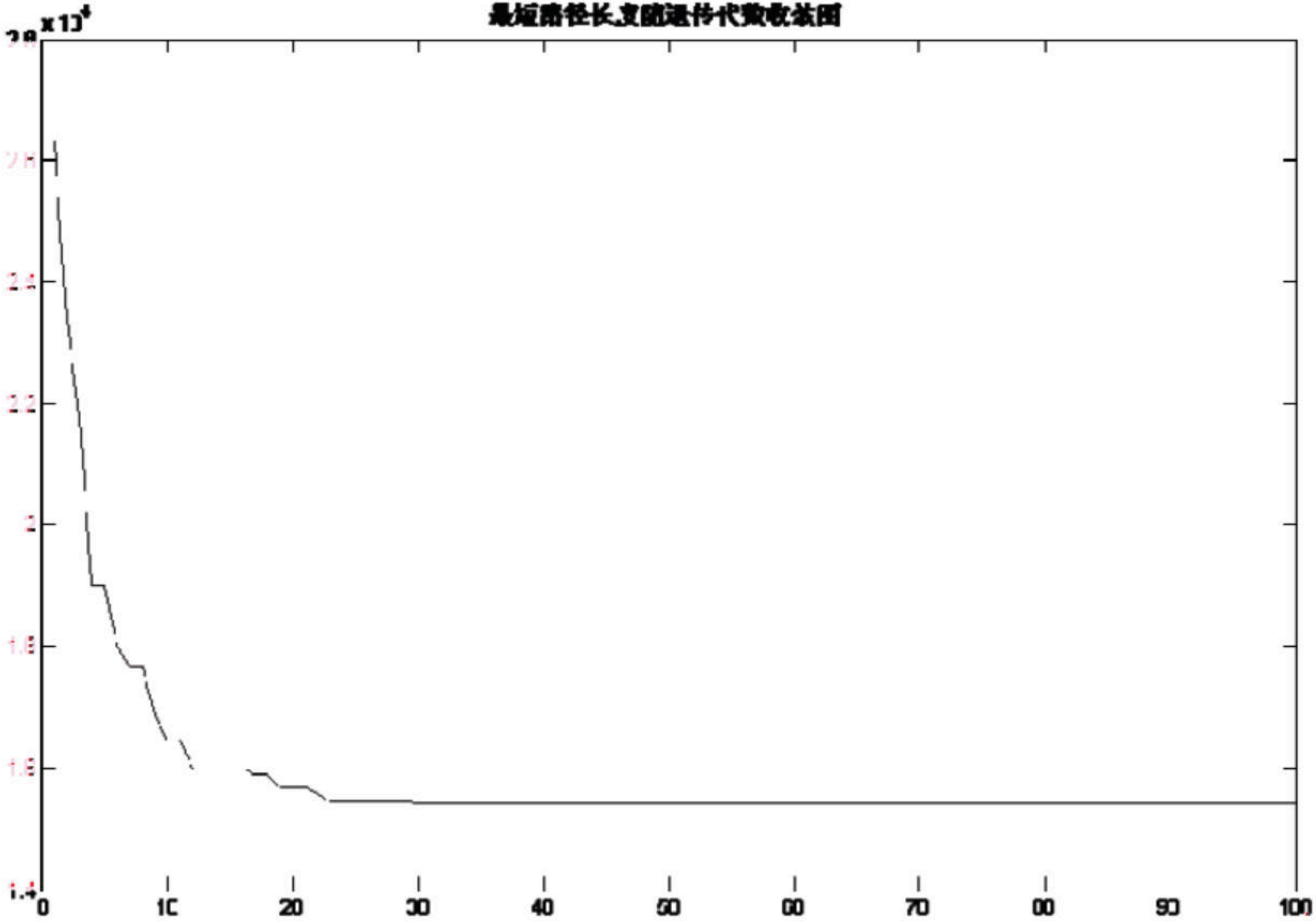
中国31 城市问题

城市坐标：

```
china = [1304 2312；3639 1315；4177 2244；3712 1399；3488 1535；3326 1556；…
    3238 1229；4196 1004；4312 790；4386 570；3007 1970；2562 1756；…
    2788 1491；2381 1676；1332 695；3715 1678；3918 2179；4061 2370；…
    3780 2212；3676 2578；4029 2838；4263 2931；3429 1908；3507 2367；…
    3394 2643；3439 3201；2935 3240；3140 3550；2545 2357；2778 2826；2370 2975]；
```

在经过50代后得到最短距离为：15468，最优值15309。
最佳图路径为：10, 2, 4, 5, 6, 7, 13, 12, 14, 15, 1, 29, 31, 3, 27, 28, 26, 21, 22, 18, 3, 17, 19, 20, 25, 24, 11, 23, 16, 8

中国31 城市最短路径长度收敛图



进化规划算法简介
（Evolution Programming, EP）

- 二十世纪60年代提出，90年代形成并推广
- 模拟自然进化过程的一种随机搜索算法
- 与遗传算法相比，没有交叉过程，变异和选择过程也不同
- **变异**：高斯变异算子，

- **选择**：父代、子代一同竞争，依概率、锦标赛、精英策略

基本流程：

种群初始化（随机分布个体）→评价个体（适应度计算）
→个体更新（变异）→群体更新（选择）

- 锦标赛选择算子：

1）种群记为I

2）$\forall x_i \in I$，从I中随机选择 q 个个体，将q个个体的适应值与x_i的适应值比较，得到比x_i的适应值差的个体数目w_i，记为x_i的得分

3）遍历种群I的所有个体，将其按得分排序，选择得分最高的N个个体作为下一代种群。

进化策略算法简介
（Evolution Strategy, ES）

- 二十世纪60年代提出，德国柏林工业大学
- 1980's形成两种经典的进化策略算法
- 编码方式：十进制实数
- μ+λ-进化策略：μ个父代重组变异→ λ个子代，父代、子代共μ+λ个一同参加竞争，选出μ个下一代种群
- (μ,λ)-进行策略：μ个父代重组变异→ λ个子代，λ(λ>μ)个子代参加竞争，选出μ个下一代种群

基本流程：

种群初始化（随机分布个体）→评价个体（适应度计算）→个体更新（重组+变异）→群体更新（选择）