

第四讲 遗传算法





- ✓ 遗传算法基本概念
发展历史，基本概念，遗传操作 ...
- ✓ 遗传算法基本步骤
基本步骤、算法参数
- ✓ 基本遗传算法应用举例
函数求极值
- ✓ 旅行商 问题求解
遗传算法参数设定...
- ✓ 总结
遗传算法的特点与优势



- ✓ **遗传算法基本概念**
发展历史，基本概念，遗传操作 ...
- ✓ **遗传算法基本步骤**
基本步骤、算法参数
- ✓ **基本遗传算法应用举例**
函数求极值
- ✓ **旅行商 问题求解**
遗传算法参数设定...
- ✓ **总结**
遗传算法的特点与优势

遗传算法发展历史

遗传算法的诞生

- 1、1967年，Holland学生J. D. Bagley在博士论文中首次提出“遗传算法（Genetic Algorithms）”；
- 2、1971年，R.B.Hollstien在他的博士论文中首次把遗传算法用于函数优化；
- 3、1975年，Holland出版了他的著名专著《自然系统和人工系统的自适应》
（Adaptation in Natural and Artificial Systems），这是第一本系统论述遗传算法的专著；
- 4、K.A.De Jong完成了他的博士论文《一类遗传自适应系统的行为分析》
（An Analysis of the Behavior of a Class of Genetic Adaptive System）

基本概念

个体

- 个体就是模拟生物个体，对问题中对象（一般就是问题的解）的一种称呼。
- 一个个体也就是搜索空间中的一个点。

种群

- 种群(population)就是模拟生物种群, 由若干个体组成的群体.
- 它一般是整个搜索空间的一个很小的子集.

基本概念——适应度与适应度函数

适应度(fitness)

- 借鉴生物个体对环境的适应程度, 对问题中的个体对象所设计的表征其优劣的一种测度。

适应度函数(fitness function)

- 是问题中全体个体与其适应度之间的一个对应关系。
- 通常为实值函数。
- 该函数就是遗传算法中指导搜索的评价函数。

基本概念——染色体与基因

染色体（chromosome）与基因（gene）

- 染色体是问题中个体的某种字符串形式的编码表示。
- 字符串中的字符称为基因。

例如：

- 个体 染色体
- 9 ---- 1001
- (2, 5, 6) ---- 010 101 110

基本概念——遗传操作

遗传操作亦称遗传算子(genetic operator)

关于染色体的运算。

遗传算法中有三种遗传操作:

- 选择-复制(selection-reproduction)
- 交叉(crossover, 亦称交换、交配或杂交)
- 变异(mutation, 亦称突变)

选择-复制

对于一个规模为 N 的种群 S ,按每个染色体 $x_i \in S$ 的选择概率 $P(x_i)$ 所决定的选中机会,分 N 次从 S 中随机选定 N 个染色体,并进行复制。

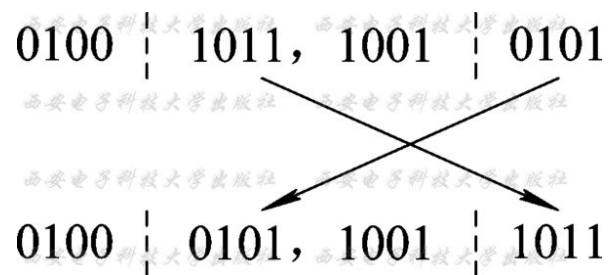
这里的选择概率 $P(x_i)$ 的计算公式为:

$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^N f(x_j)}$$

交叉

交叉: 互换两个染色体某些位上的基因。

例如, 设染色体 $s1=01001011$, $s2=10010101$, 交换其后4位基因, 即



$s1'=01000101$, $s2'=10011011$

可以看做是原染色体 $s1$ 和 $s2$ 的子代染色体。

变异

变异: 就是改变染色体某个(些)位上的基因。

例如, 设染色体 $s = 11001101$, 将其第三位上的0变为1, 即

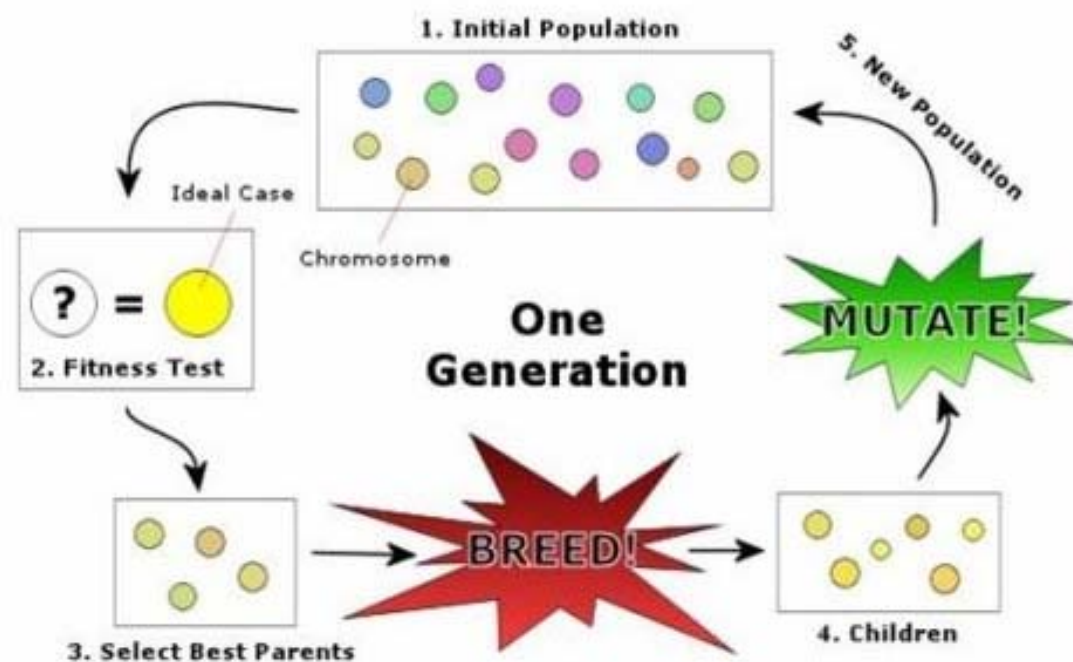
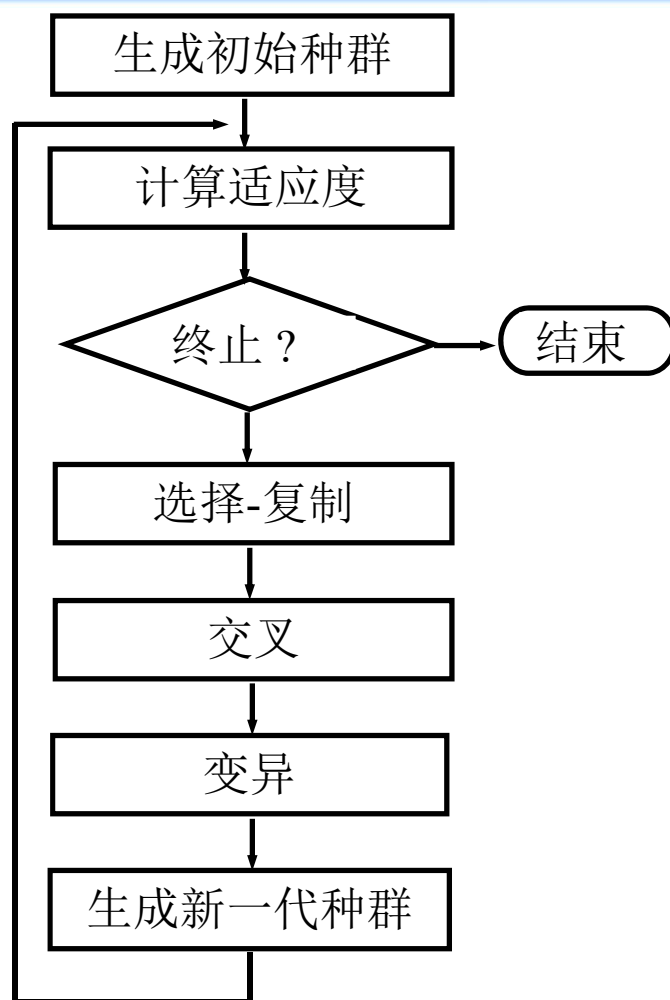
$$s = 11\underline{0}01101 \rightarrow 11\underline{1}01101 = s'.$$

s' 也可以看做是原染色体 s 的子代染色体。



- ✓ 遗传算法基本概念
发展历史，基本概念，遗传操作 ...
- ✓ 遗传算法基本步骤
基本步骤、算法参数
- ✓ 基本遗传算法应用举例
函数求极值
- ✓ 旅行商 问题求解
遗传算法参数设定...
- ✓ 总结
遗传算法的特点与优势

遗传算法基本流程框图



基本遗传算法步骤

步1: 在搜索空间 U 上定义一个适应度函数 $f(x)$, 给定种群规模 N , 交叉率 P_c 和变异率 P_m , 代数 T ;

步2: 随机产生 U 中的 N 个个体 s_1, s_2, \dots, s_N , 组成初始种群 $S=\{s_1, s_2, \dots, s_N\}$, 置代数计数器 $t=1$;

步3: 计算 S 中每个个体的适应度 f ;

步4: 若终止条件满足, 则取 S 中适应度最大的个体作为所求结果, 算法结束。

基本遗传算法步骤 2

步5: 按选择概率 $P(x_i)$ 决定选中机率, 每次从 S 中随机选定1个体并将其染色体复制, 共做 N 次, 然后将复制所得的 N 个染色体组成群体 S_1 ;

步6: 按交叉率 P_c 决定参加交叉的染色体数 c , 从 S_1 中随机确定 c 个染色体, 对进行交叉操作, 并用产生的新染色体代替原染色体, 得群体 S_2 ;

步7: 按变异率 P_m 决定变异次数 m , 从 S_2 中随机确定 m 个染色体, 分别进行变异操作, 并用产生的新染色体代替原染色体, 得群体 S_3 ;

步8: 将群体 S_3 作为新一代种群, 即用 S_3 代替 S , $t = t+1$, 转步3;

遗传算法中的一些控制参数

种群规模

最大换代数

交叉率(crossover rate)

- 参与交叉运算的染色体个数占全体染色体总数的比例，记为 P_c ，取值范围一般为0.4~0.99。

变异率(mutation rate)

- 指发生变异的基因位数所占全体染色体的基因总位数的比例，记为 P_m ，取值范围一般为0.0001~0.1。

遗传算法中交叉与变异的作用

复制的作用：

保留优良个体，但不会产生新个体。

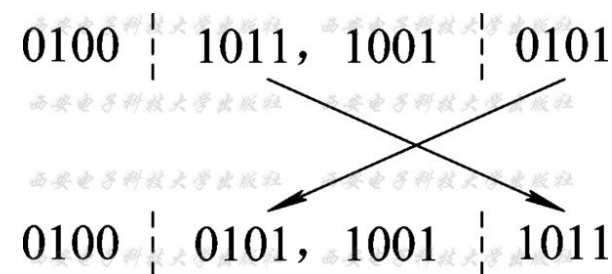
控制进化的方向，而交叉、变异等算子不能控制进化方向。

交叉的作用：

会产生新个体，但子代个体与父代的差异不大。

如右图，只能产生 0100,xxxx,xxxx,xxxx 这个范围内新个体。

与父代的差异较小，进化速度慢，且易于陷入局部最小。



变异的作用：

会产生突变的新个体，会产生差异较大的新个体

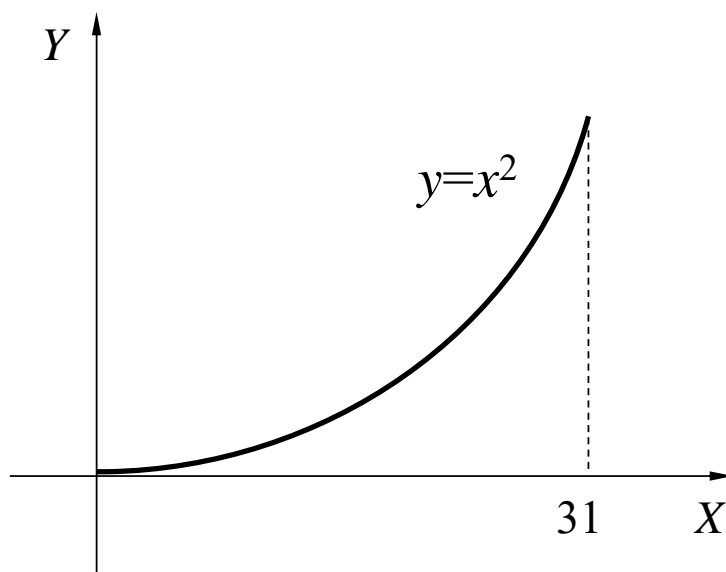
若变异率太大，使得进化变成了随机算法。



- ✓ 遗传算法基本概念
发展历史，基本概念，遗传操作 ...
- ✓ 遗传算法基本步骤
基本步骤、算法参数
- ✓ **基本遗传算法应用举例**
函数求极值
- ✓ 旅行商 问题求解
遗传算法参数设定...
- ✓ 总结
遗传算法的特点与优势

遗传算法应用举例

利用遗传算法求解区间 $[0,31]$ 上的二次函数 $y=x^2$ 的最大值。



分析

原问题可转化为在区间 $[0, 31]$ 中搜索能使 y 取最大值的点 a 的问题。

$[0, 31]$ 中的每个点 x 就是个体。

函数值 $f(x)$ 恰好就可以作为 x 的适应度。

区间 $[0, 31]$ 就是一个(解)空间。

问题：如何给出个体 x 的适当染色体编码？使得该问题可用遗传算法来解决。

求解步骤1：随机选取初始种群

(1)设定种群规模, 编码染色体, 产生初始种群。

- 假定将种群规模设定为4;
- 采用5位二进制数编码染色体;
- 随机选取4个个体, 组成初始种群 S_1 :
 - $s_1 = 13$ (01101)
 - $s_2 = 24$ (11000)
 - $s_3 = 8$ (01000)
 - $s_4 = 19$ (10011)

求解步骤2： 定义适应度函数

(2) 定义适应度函数

- 取适应度函数： $f(x)=x^2$
- 适应度函数选取是具体问题具体分析。

求解步骤3： 计算适应度函数

(3) 计算当前种群中的各个体的适应度。

- 以初始种群 S_1 中各个体的适应度 $f(s_i)$ 。

- $s_1=13(01101)$, $s_2=24(11000)$,

- $s_3=8(01000)$, $s_4=19(10011)$

- 有： $f(s_1)=f(13)=13^2=169$

$$f(s_2)=f(24)=24^2=576$$

$$f(s_3)=f(8)=8^2=64$$

$$f(s_4)=f(19)=19^2=361$$

求解步骤4： 计算选择概率

(4) 计算当前种群中各个体的选择概率。

- 选择概率的计算公式为

$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^N f(x_j)}$$

- 以初始种群 S_1 为例：

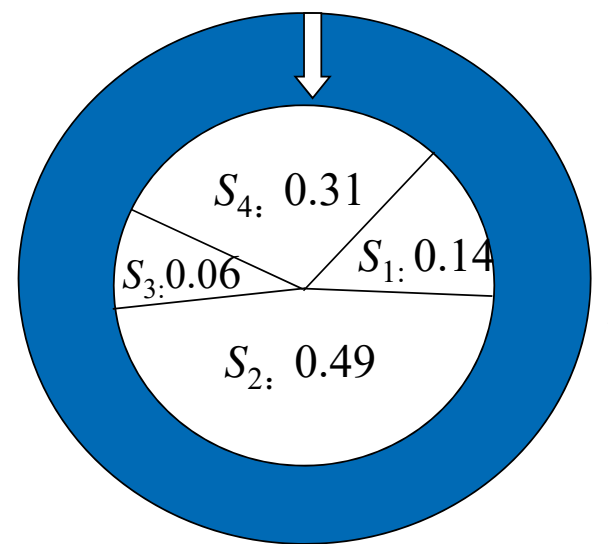
- $P(s1) = P(13) = 169/1170 = 0.14$
- $P(s2) = P(24) = 576/1170 = 0.49$
- $P(s3) = P(8) = 64 /1170 = 0.06$
- $P(s4) = P(19) = 361/1170 = 0.31$

求解步骤5：选择-复制

选择下一代种群的个体，选择-复制：

- 赌轮选择法
- 赌轮选择法过程：
 - ① 在 $[0,1]$ 区间内产生一个均匀分布的随机数 r 。
 - ② 若 $r \leq q_1$ ，则染色体 x_1 被选中。
 - ③ 若 $q_{k-1} < r \leq q_k$ ($2 \leq k \leq N$)，则染色体 x_k 被选中。

其中的 q_i 称为染色体 x_i ($i=1, 2, \dots, n$)的积累概率，其计算公式为 $q_i = \sum_{j=1}^i P(x_j)$



求解步骤5举例:

设从区间[0,1]中产生4个随机数如下:

- $r1 = 0.450126$, $r2 = 0.110347$
- $r3 = 0.572496$, $r4 = 0.98503$

染色体	适应度	选择概率	积累概率	选中次数
$s_1=01101(13)$	169	0.14	0.14	1 ($r2$)
$s_2=11000(24)$	576	0.49	0.63	2 ($r1$ 和 $r3$)
$s_3=01000(8)$	64	0.06	0.69	0
$s_4=10011(19)$	361	0.31	1.00	1 ($r4$)

求解步骤5举例的结果:

经复制得下一代新种群的初始个体:

- $s1' = 11000$ (24) , $s2' = 01101$ (13)
- $s3' = 11000$ (24) , $s4' = 10011$ (19)

结束了吗?

求解步骤6：交叉

根据交叉率从新种群选取参与交叉运算的个体。

若交叉率 $p_c=100\%$ ，则全体染色体都参加交叉运算。

随机选择两对染色体参与交叉。

随机选择交叉运算时的交换位数。

举例：

- 设在选择-复制获得新种群后，随机选择 $s1'$ 与 $s2'$ 配对交叉， $s3'$ 与 $s4'$ 配对交叉。

求解步骤6举例

$$s1' = 11000 \text{ (24)}, s2' = 01101 \text{ (13)}$$

随机确定最后两位交叉，分别交换后两位基因，得新染色体：

- $s1'' = 11001 \text{ (25)}, s2'' = 01100 \text{ (12)}$

同样： $s3' = 11000 \text{ (24)}, s4' = 10011 \text{ (19)}$

- 最后两位交叉，得新染色体：

- $s3'' = 11011 \text{ (27)}, s4'' = 10000 \text{ (16)}$

求解步骤6举例的结果

新种群中各个体的染色体:

- $s1''=11001$ (25) , $s2''=01100$ (12)
- $s3''=11011$ (27) , $s4''=10000$ (16)

求解步骤7：变异

根据变异率 p_m ，对新种群中个体的染色体基因进行变异。

新种群中各个体染色体的各个基因，产生随机数，若小于 p_m ，则变异（反转）。

举例：

- 设变异率 $p_m=0.001$
- 从总体看，共有 $5 \times 4 \times 0.001=0.02$ 位基因发生变异。0.02位显然不足1位，所以不会有变异。

求解步骤8：下一代种群的产生

第二代种群S2:

- $s_1=11001$ (25) , $s_2=01100$ (12)
- $s_3=11011$ (27) , $s_4=10000$ (16)

计算新种群中最大的适应度值

- $s_3=11011$ $f(s_3) = 729$
- 大于前一代种群中的最大适应度值

新种群“适者生存”，新种群作为当前种群。为第二代种群S2。

返回步骤3，迭代

迭代求解步骤3、4

当前种群中染色体 S_2	适应度	选择概率	积累概率	估计的选中次数
$s_1=11001(25)$	625	0.36	0.36	1
$s_2=01100(12)$	144	0.08	0.44	1
$s_3=11011(27)$	729	0.41	0.85	1
$s_4=10000(16)$	256	0.15	1.00	1

迭代求解步骤5

选择-复制

- 这一轮假设选择-复制操作中，经随机选择，将种群 S_2 中的4个染色体都选中，则得到新群体：
- $s_1' = 11001$ (25), $s_2' = 01100$ (12)
- $s_3' = 11011$ (27), $s_4' = 10000$ (16)

迭代求解步骤6、7

$s1' = 11\underline{001}$ (25), $s2' = 01\underline{100}$ (12), $s3' = 11\underline{011}$ (27), $s4' = 10\underline{000}$ (16)

交叉运算,

- 经随机确定, 让 $s1'$ 与 $s2'$, $s3'$ 与 $s4'$ 分别交换后三位基因, 得:

- $s1'' = 11\underline{100}$ (28), $s2'' = 01\underline{001}$ (9)

- $s3'' = 11\underline{000}$ (24), $s4'' = 10\underline{011}$ (19)

变异运算

- 随机确定, 没有变异发生。

新种群产生

得到新一代种群：

- $s_1=11100$ (28) , $s_2=01001$ (9)
- $s_3=11000$ (24) , $s_4=10011$ (19)

计算新种群中最大的适应度值

- $s_1=11100$ $f(s_1) = 784$
- 大于前一代种群中的最大适应度值

新种群“适者生存”，新种群作为当前种群，为第三代种群 S_3 。

第三代种群S3中各染色体情况

染色体	适应度	选择概率	积累概率	估计的选中次数
$s_1=11100$ (28)	784	0.44	0.44	2
$s_2=01001$ (9)	81	0.04	0.48	0
$s_3=11000$ (24)	576	0.32	0.80	1
$s_4=10011$ (19)	361	0.20	1.00	1

第三代的遗传运算

设这一轮的选择-复制结果为:

- $s1'=11100$ (28) , $s2'=11100$ (28)

- $s3'=11000$ (24) , $s4'=10011$ (19)

做交叉运算, 让 $s1'$ 与 $s4'$, $s2'$ 与 $s3'$ 分别交换后两位基因, 得

- $s1''=11111$ (31) , $s2''=11100$ (28)

- $s3''=11000$ (24) , $s4''=10000$ (16)

这一轮仍然不会发生变异。

获得第四代种群S4

第四代种群S4:

- $s1=11111$ (31) , $s2=11100$ (28)
- $s3=11000$ (24) , $s4=10000$ (16)

在这一代种群中已经出现了适应度最高的染色体 $s1=11111$ 。

之后遗传进化所得的各代种群，最大个体适应度值均小于等于 $s1=11111$ 。

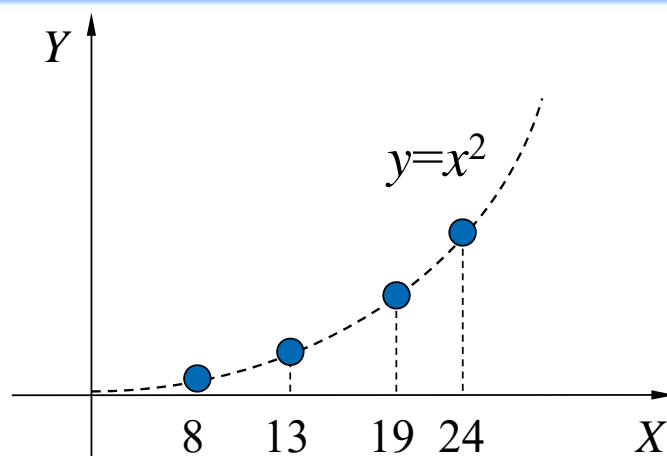
于是，遗传操作终止，将染色体“11111”作为最终结果输出。

遗传算法数值结果

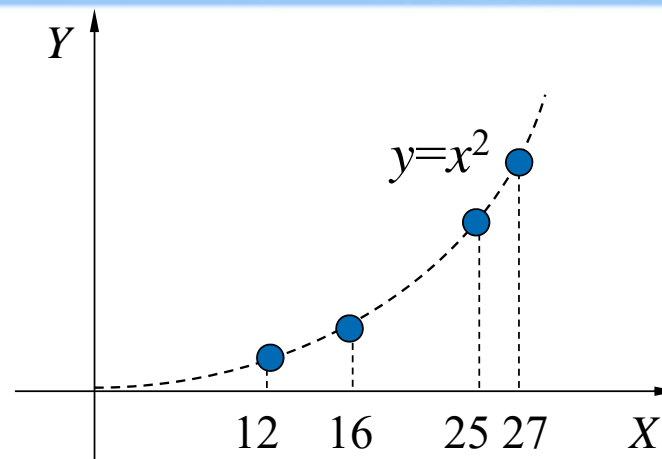
染色体“11111”解码为表现型，即得所求的最优解：31。

将31代入函数 $y=x^2$ 中得原问题的解，即函数 $y=x^2$ 的最大值为961。

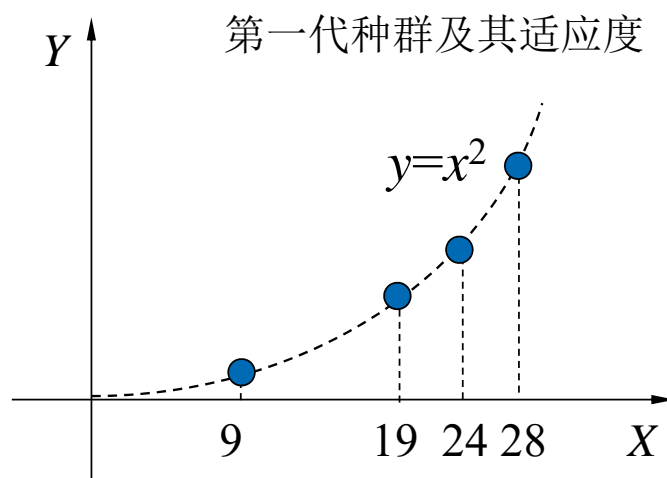
遗传算法迭代图示



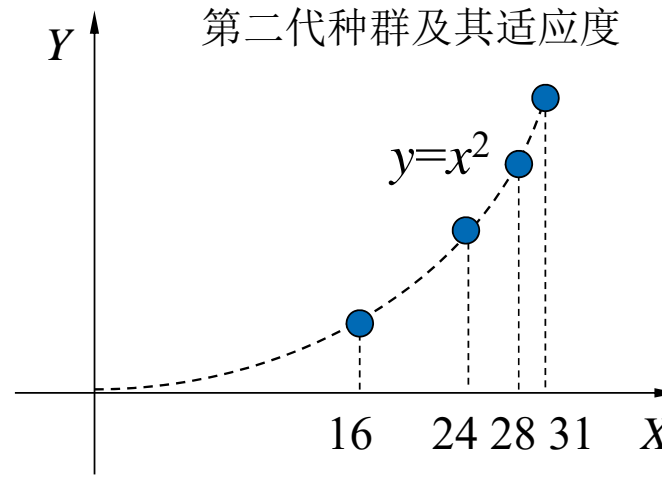
第一代种群及其适应度



第二代种群及其适应度



第三代种群及其适应度



第四代种群及其适应度

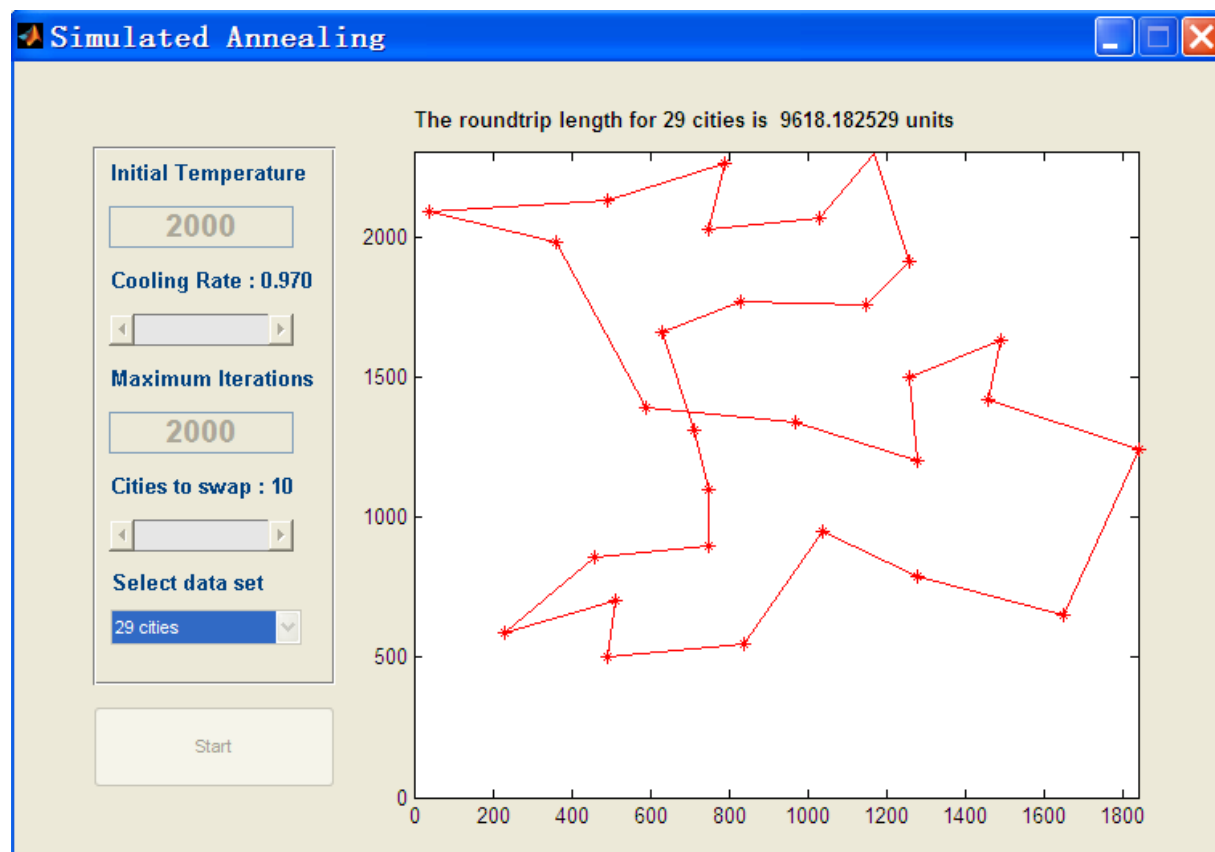


- ✓ 遗传算法基本概念
发展历史，基本概念，遗传操作 ...
- ✓ 遗传算法基本步骤
基本步骤、算法参数
- ✓ 基本遗传算法应用举例
函数求极值
- ✓ **旅行商 问题求解**
遗传算法参数设定...
- ✓ 总结
遗传算法的特点与优势

用遗传算法求解TSP

Traveling Salesman Problem

假设有一个旅行商人要拜访 n 个城市，他必须选择所要走的路径，路径的限制是每个城市只能拜访一次，而且最后要回到原来出发的城市。路径的选择目标是要求得的路径路程为所有路径之中的最小值。



分析

如何构建染色体？

任一可能解：一个合法的城市序列，即 n 个城市的一个排列。

因此，可直接在解空间（所有合法的城市序列）中搜索最佳解。

这正适合用遗传算法求解。

一个合法的城市序列 $s = (c_1, c_2, \dots, c_n, c_{n+1})$ (c_{n+1} 就是 c_1)作为一个个体。

步骤1：定义适应度函数

城市序列中相邻两城之间的距离之和的倒数可作为相应个体 s 的适应度。

即适应度函数就是：

为什么这么定义？

$$f(s) = \frac{1}{\sum_{i=1}^n d(c_i, c_{i+1})}$$

如何编码和交叉

虽然选定一个合法的城市序列 $(c_1, c_2, \dots, c_n, c_{n+1})$ (c_{n+1} 就是 c_1) 作为一个个体。

还需对个体进行编码。

如果编码不当，就会在实施交叉或变异操作时出现非法城市序列即无效解。

例如，对于5个城市的TSP，用符号A、B、C、D、E代表相应的城市，用这5个符号的序列表示可能解即染色体。

出现的问题？

设 $s1=(A, C, B, E, D, A)$, $s2=(A, E, D, C, B, A)$ 进行遗传操作:

- 实施常规的交叉操作，如交换后三位，得
 - $s1'=(A, C, B, C, B, A)$, $s2'=(A, E, D, E, D, A)$
- 实施常规的变异操作，将染色体 $s1$ 第二位的 C 变为 E ，得
 - $s1''=(A, E, B, E, D, A)$
- 可以看出，上面得到的 $s1'$, $s2'$ 和 $s1''$ 都是非法的城市序列。

求解方法

对TSP必须设计合适的染色体和相应的遗传运算。

针对TSP已提出了许多编码方法和相应的特殊化的交叉、变异操作，以巧妙地用遗传算法解决TSP。

如顺序编码或整数编码、随机键编码。

如部分映射交叉、顺序交叉、循环交叉、位置交叉。

如反转变异、移位变异、互换变异等等。

编码方式

二进制编码

(000 001 010 011 100 101)

- 对TSP而言没有意义，因为城市号就是基本单元。

路径表示

(6 3 5 4 1 2)

- 最常用的编码方式
- 有许多特殊的遗传操作算子

Matrix representation

(2 3 1 4)



$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

交叉方式——部分映射交叉

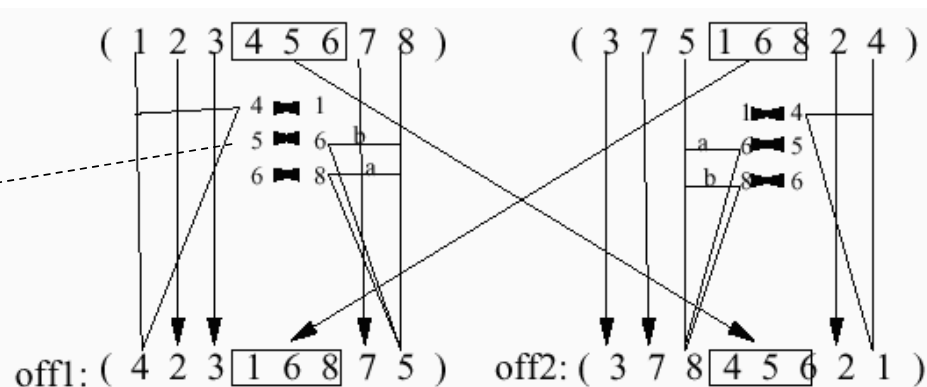
部分映射交叉PMX(Partial-Mapped Crossover)

2 parents

(1 2 3 4 5 6 7 8)

(3 7 5 1 6 8 2 4)

1. 随机取两点,例如4、6
2. 两点内部分先行交换
3. 其余依次填入,但重复者必须替换



- 1) 第四位交换, 4和1交换

原来 1 2 3 4 5 6 7 8 3 7 5 1 6 8 2 4

左边: 4 2 3 1 5 6 7 8 右边: 3 7 5 4 6 8 2 1

- 2) 第五位交换, 5和6交换

原来 4 2 3 1 5 6 7 8 右边: 3 7 5 4 6 8 2 1

左边: 4 2 3 1 6 5 7 8 右边: 3 7 6 4 5 8 2 1

- 3) 第六位交换, 6和8交换

原来: 4 2 3 1 6 5 7 8 右边: 3 7 6 4 5 8 2 1

左边: 4 2 3 1 6 8 7 5 右边: 3 7 8 4 5 6 2 1

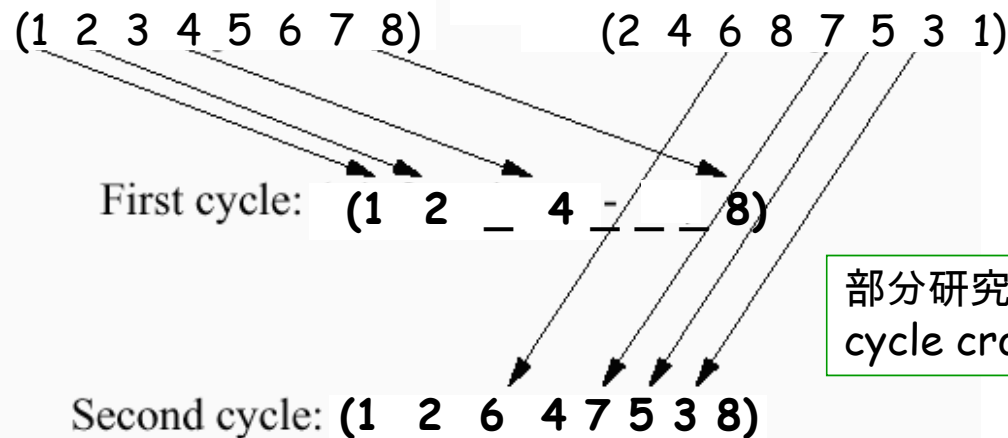
交叉方式——循环交叉 Cycle Crossover

2 parents 產生一個offspring

(1 2 3 4 5 6 7 8)

(2 4 6 8 7 5 3 1)

1. 从第一个位置开始填起, 2 选 1, 下例为选到左边parent的情形
2. 填好第一个位置之後后, 看另一个parent同样number(此处為 1)的位置何在(位置8)
3. 再將原parent同位置的number填入offspring, 其餘依此類推
4. 左邊parent沒有number可供填入, 則找右邊的parent, 從offspring最前面的空位填起



部分研究实验显示
cycle crossover较PMX效果佳

变异方式——移位变异

Displacement Mutation(DM)

the parent

(1 2 3 4 5 6 7 8)

1. 随机选出一段sub-tour, 例如选到(3 4 5)
2. 先将其余cities重新排好,如(1 2 6 7 8)
3. 将(3 4 5)随机插入任一个位置

(1 2 3 4 5 6 7 8)

(1 2 6 7 3 4 5 8)

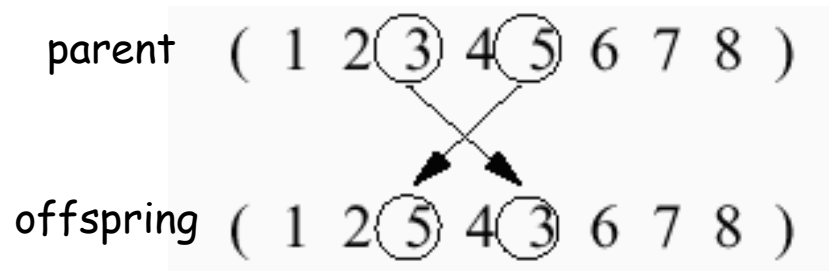
变异方式——互换变异

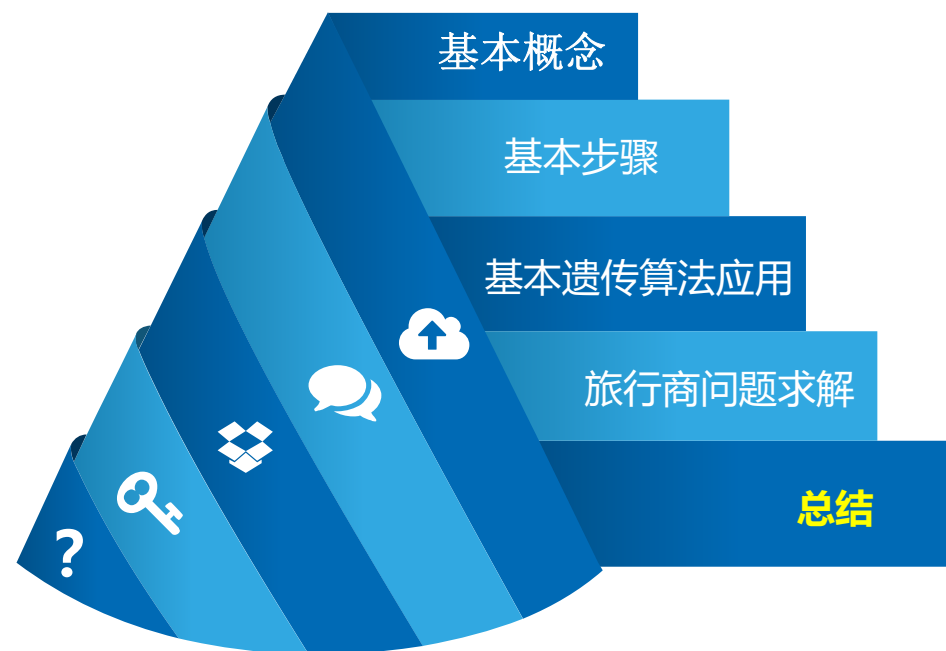
Exchange Mutation(EM) 或 swap mutation

the parent

(1 2 3 4 5 6 7 8)

1. 任意选择两个位置, 将之交换





- ✓ 遗传算法基本概念
发展历史，基本概念，遗传操作 ...
- ✓ 遗传算法基本步骤
基本步骤、算法参数
- ✓ 基本遗传算法应用举例
函数求极值
- ✓ 旅行商 问题求解
遗传算法参数设定...
- ✓ **总结**
遗传算法的特点与优势

遗传算法的特点与优势

遗传算法与图搜索方法的主要区别：

- 遗传算法一般是直接在解空间搜索,
- 图搜索是在问题空间搜索, 最后才找到解。
- 遗传算法搜索是随机始于搜索空间中的一个点集, 因而它是一种随机搜索算法。
- 图搜索始于搜索空间的初始节点或终止节点。

遗传算法与图搜索方法的主要区别2

遗传算法旨在寻找优解，是一种优化搜索算法。

图搜索并非总是要求优解，只为了尽快找到解。

遗传算法的搜索过程是从空间的一个点集(种群)到另一个点集(种群)的搜索。

因而它实际是一种并行搜索，适合大规模并行计算，且这种种群到种群的搜索有能力跳出局部最优解。

图搜索是从空间的一个点到另一个点地搜索。

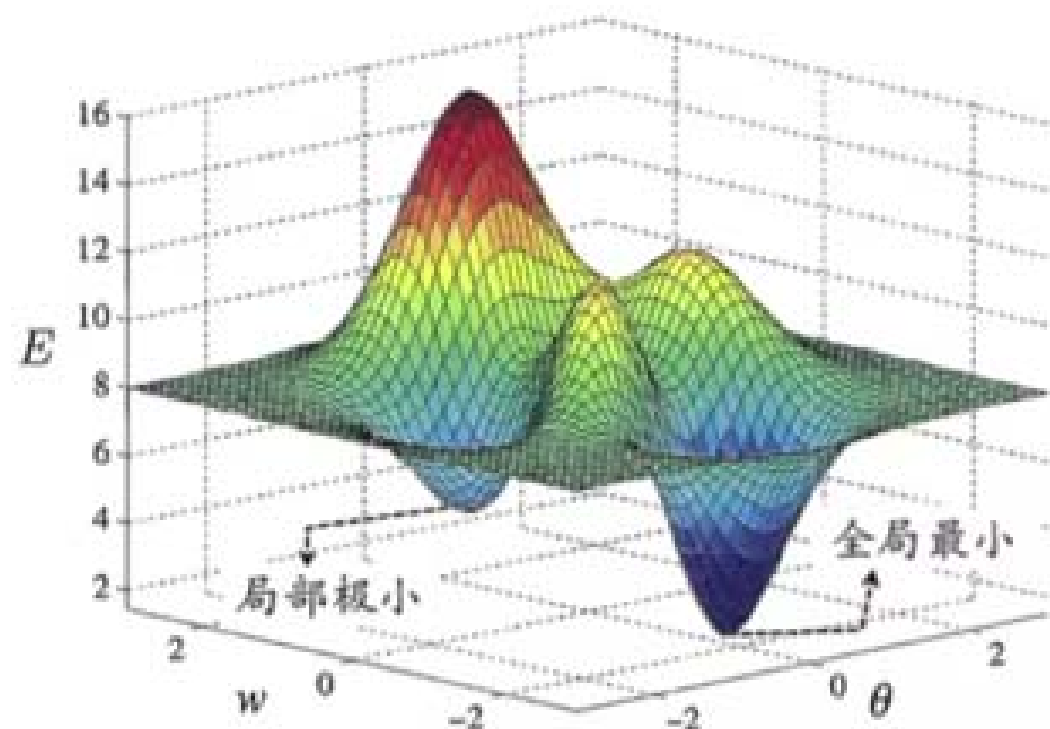
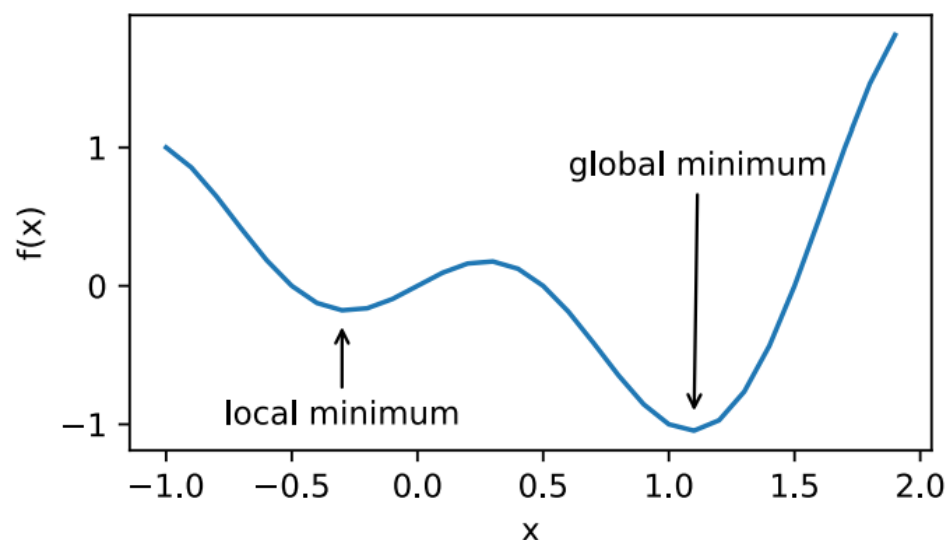
遗传算法与图搜索方法的主要区别3

遗传算法的适应性强, 除需知适应度函数外, 几乎不需要其他的先验知识。

遗传算法适于全局搜索, 它不受搜索空间的限制性假设的约束, 不要求连续性, 能以很大的概率从离散的、多极值的、含有噪声的高维问题中找到全局最优解。但时间无穷大。

实现全局最优解的前提是接受下一代种群是否允许存在比前一代种群适应度低的情况。

局部最小问题



跳出局部最小的策略:

- 1、优化初始值
- 2、模拟退火
- 3、随机扰动

课堂实验

遗传算法解决旅行商问题

讨论:

- 1、遗传算法的优缺点
- 2、跳出局部最小问题

```
Editor - /Users/sunshine/Desktop
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment % Find Indent Breakpoints Run Run and Advance
FILE NAVIGATE EDIT BREAKPOINTS RUN

main.m x +
1 close all; clear all; clc;
2
3 % the number of cities
4 prompt = 'What is the number of cities in this TSP program? [30]';
5 numberOfCities = input(prompt);
6 if (isempty(numberOfCities))
7     numberOfCities = 30;
8 end
9 global gNumber;
10
11 % the number of population size
12 prompt = 'What is the number of population size? [500]';
13 popSize = input(prompt);
14 if (isempty(popSize))
15     popSize = 500;
16 end
```

MATLAB R2014b

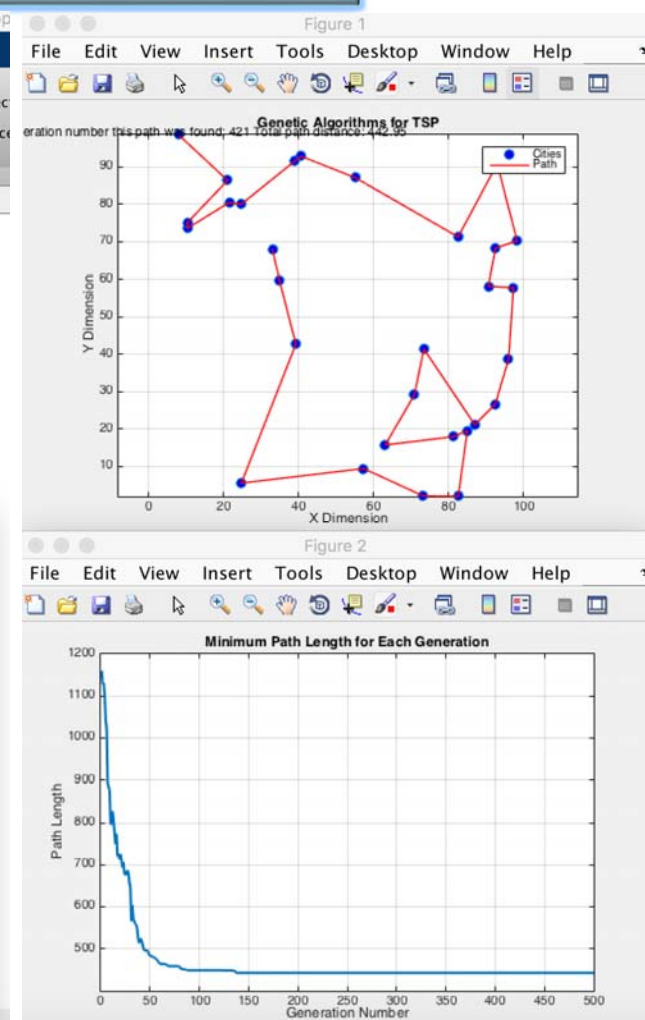
HOME PLOTS APPS Search Documentation

FILE

Command Window

Minimum path in 492. generation: 442.948940
Minimum path in 493. generation: 442.948940
Minimum path in 494. generation: 442.948940
Minimum path in 495. generation: 442.948940
Minimum path in 496. generation: 442.948940
Minimum path in 497. generation: 442.948940
Minimum path in 498. generation: 442.948940
Minimum path in 499. generation: 442.948940
Minimum path in 500. generation: 442.948940
Elapsed time: 0 minutes and 57.416616 seconds.

fx >>



END !