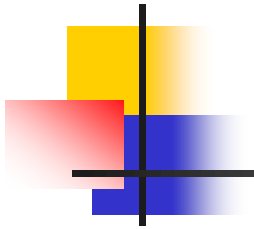
高级算法设计与分析

Advanced Computer Algorithm Design & Analysis

2024.12

吕志鹏 丁俊文 苏宙行

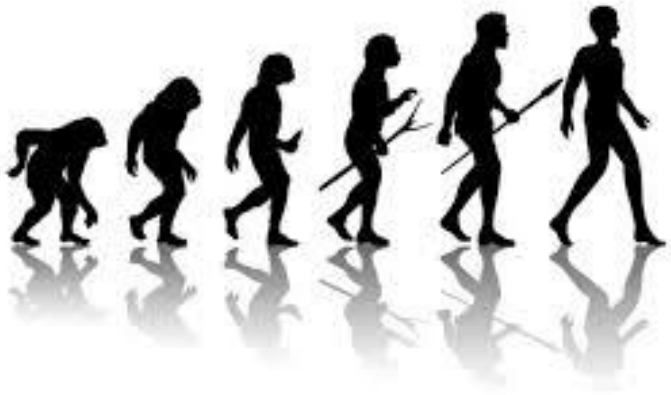
zhipeng.lv@hust.edu.cn

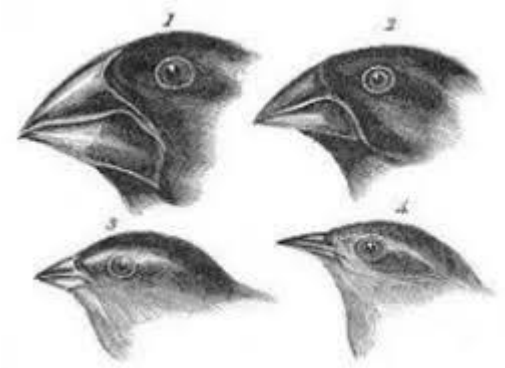


1. 导言——生物的进化

遗传学理论

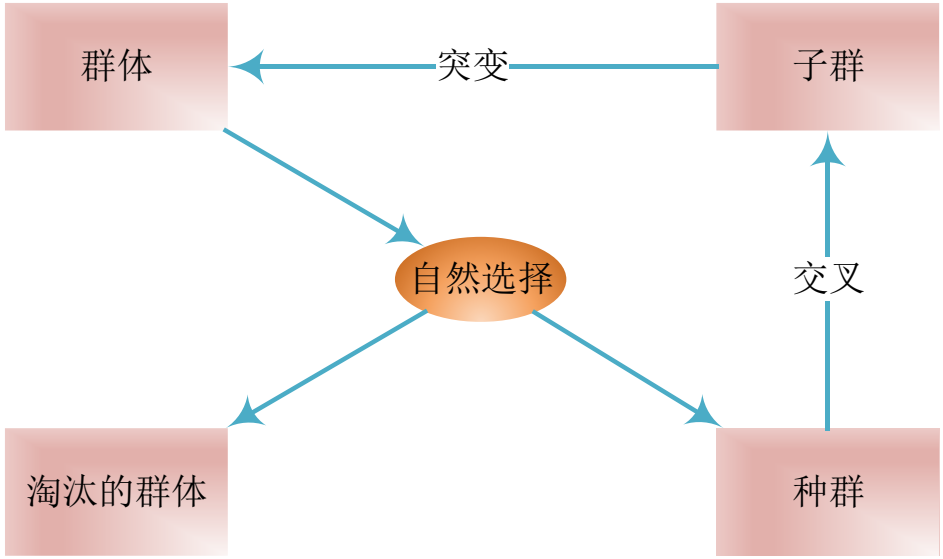
- 达尔文的进化论：物竞天择，适者生存：
  - 具有较强生存能力的生物个体容易存活下来，并有较多的机会产生后代；
  - 具有较低生存能力的个体则被淘汰，或者产生后代的机会越来越少，直至消亡





遗传学理论

- 孟德尔和摩根的遗传学理论：
  - 通过基因交叉和突变可以产生对环境适应性强的后代；
  - 经过优胜劣汰的自然选择，适应值高的基因结构就得以保存下来。



生物进化基本条件

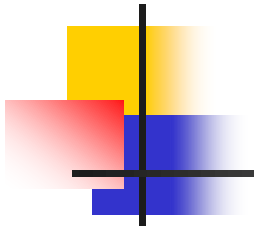
- 生物进化过程的发生需要四个基本条件：
  - 存在由多个生物个体组成的种群；
  - 生物个体之间存在着差异，或群体具有多样性；
  - 生物能够自我繁殖；
  - 不同个体具有不同的环境生存能力，具有优良基因结构的个体繁殖能力强，反之则弱。

生物进化形式

- 生物群体的进化机制可以分为三种基本形式：
  - 自然选择：适应环境变化的生物个体具有更高的生存能力，其所具有的染色体特征会得以保留。
  - 交叉：通过交叉组合来自父代染色体的遗传物质，产生不同的染色体。
  - 突变：随机改变父代个体的染色体上的基因结构。

生物进化的借鉴意义

- 自然界的生物进化是一个不断循环的过程。在这一过程中，生物群体不断地完善和发展。
- 可见，生物进化过程本质上是一种优化过程。
- 因此模拟进化过程，创立新的优化计算方法，可以应用到复杂的工程领域之中。
- 这是GA等模拟自然进化的计算方法的思想源泉。



2. 遗传算法的基本原理

引言

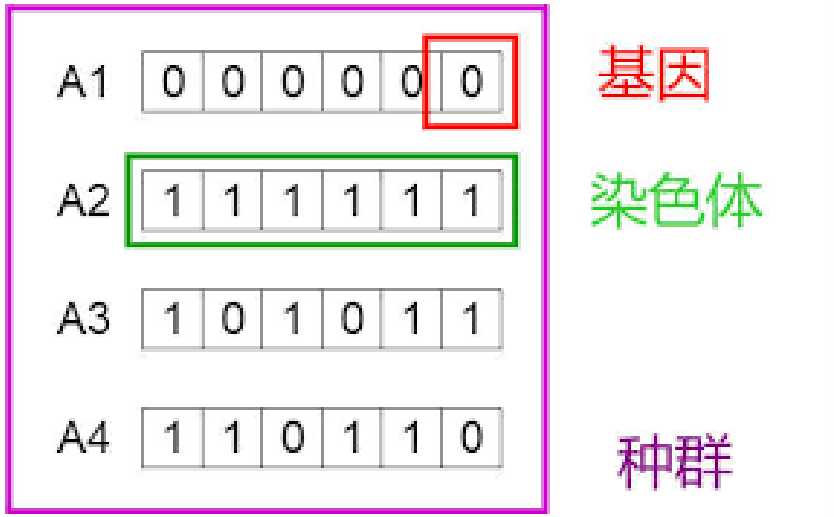
- 遗传算法：
  - 借鉴生物界自然选择和遗传机制的随机搜索算法；
  - 处理传统搜索方法难以求解的复杂和非线性优化问题。
- 遗传算法的本质特征：
  - 群体搜索策略；
  - 简单的遗传算子。
- 遗传算法可广泛应用于：
  - 组合优化、机器学习、自适应控制、规划设计等领域。

生物遗传概念在遗传算法中的应用

生物遗传概念	GA中的应用
个体	解
染色体	解的编码
基因	解的编码中的分量
适应性	适应度函数值
群体	一组解，解的个数为群体的规模
交叉	选择两个解进行交叉产生新的解的过程
突变	解的编码的某一分量发生变化的过程
适者生存	目标值较优的解留存下来的可能性大

初始构造

- 遗传算法从代表问题可能潜在解集的一个种群开始。
- 实现从性状到基因的映射，即编码工作。



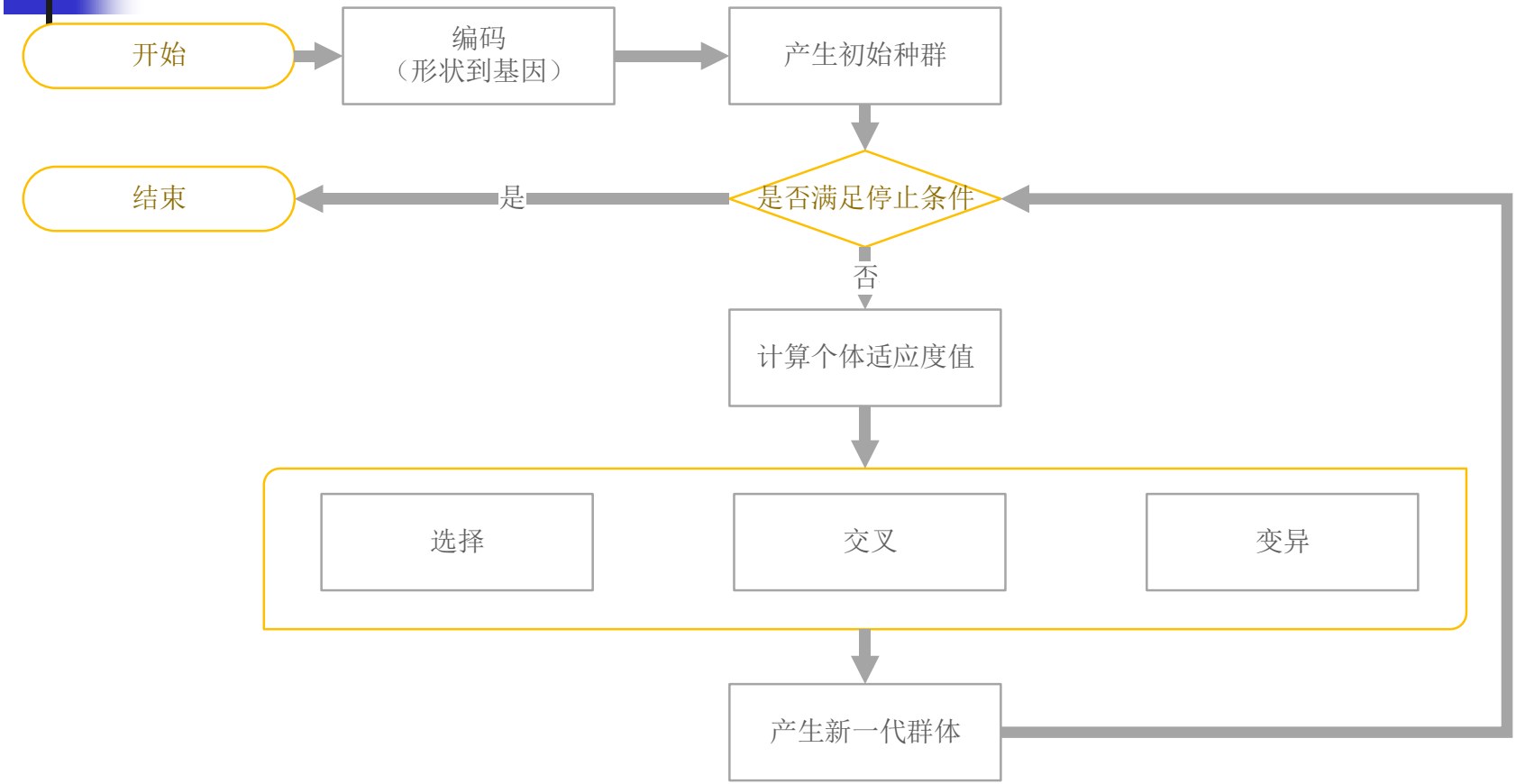
遗传操作

- 种群产生后：
  - 按照优胜劣汰的原则，根据个体适应度大小挑选（选择）个体，进行复制、交叉、变异；
  - 产生出代表新的解集的群体，再对其进行挑选以及一系列遗传操作；
  - 如此往复，逐代演化产生出越来越好的近似解。

输出结果

- 这个过程将导致种群像自然进化一样，后代种群比前代更加适应环境。
- 末代种群中的最优个体经过解码（从基因到性状的映射），可以作为问题近似最优解。

遗传算法构造及基本流程



伪代码

Genetic Algorithm

Objective function $f(\boldsymbol{x})$, $\boldsymbol{x} = (x_1, ..., x_d)^T$

Encode the solutions into chromosomes (strings)
Define fitness F (eg, $F \propto f(\boldsymbol{x})$ for maximization)
Generate the initial population
Initialize the probabilities of crossover (p_c) and mutation (p_m)

初始构造

while (t <Max number of generations)

Generate new solution by crossover and mutation
Crossover with a crossover probability p_c
Mutate with a mutation probability p_m
Accept the new solutions if their fitness increase
Select the current best for the next generation (elitism)
Update $t = t + 1$

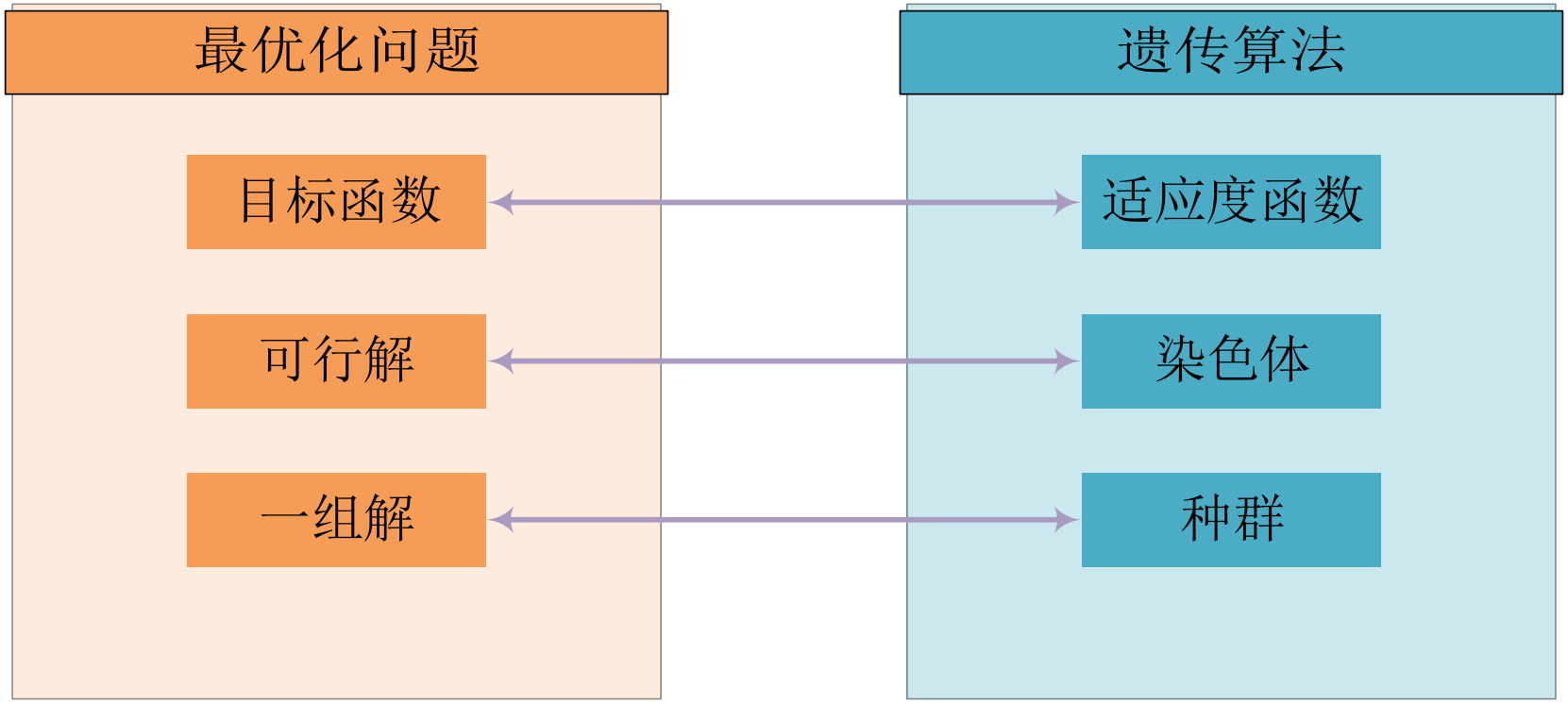
遗传操作

end while

Decode the results and visualization

输出结果

遗传算法与最优化问题的关系

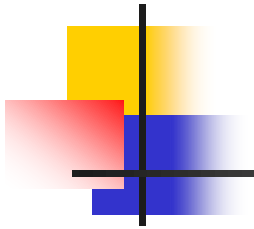


遗传算法的特点

- 同常规算法相比，遗传算法有以下特点：
  - 遗传算法是对参数的编码进行操作，提供的参数信息量大，优化效果好；
  - 遗传算法是从许多点开始并行操作，可以有效地防止搜索过程收敛于局部最优解；
  - 遗传算法通过目标函数来计算适配值，而不需要其他推导和附加信息，从而对问题的依赖性小；
  - 遗传算法的寻优规则是由概率决定的，而非确定性的；

遗传算法的优点

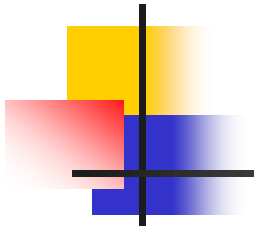
- 同常规算法相比，遗传算法有以下优点：
  - 遗传算法在解空间进行高效启发式搜索，而非盲目地穷举或完全随机搜索；
  - 遗传算法对于待寻优的函数基本无限制，因而应用范围较广；
  - 遗传算法具有并行计算的特点，因而可通过大规模并行计算来提高计算速度；
  - 遗传算法更适合大规模复杂问题的优化；
  - 遗传算法计算简单，功能强。



3. 遗传算法的基本要素

遗传算法的五个基本要素

- 参数编码
- 初始群体的设定
- 适应度函数的设计
- 遗传算子
- 控制参数设定



3.1. 参数编码

编码

- 遗传算法主要是对群体中个体施加操作。
- 遗传算法不能直接处理问题空间的参数，而只能处理以基因链码形式表示的个体。
- 将优化问题的解的参数形式转换成基因链码的表示形式的操作叫做编码。

一维编码

- 遗传算法最常用的编码方法是一维编码。
- 一维编码是指搜索空间的参数转换到遗传空间后，其相应的基因呈一维排列构成基因链码。
- 一维编码主要包括二进制编码、Gray编码、动态编码、实数编码等。

二进制编码

- 二进制编码：
  - 用若干二进制数表示个体，将原问题的解空间映射到位串空间$B = \{0,1\}$上。
- 例如：3编码为011，4编码为100。

二进制编码优点

- 使用二进制编码方法能表达的模式最多。
- 简单易行，编码类似于生物染色体的组成，算法易于用生物遗传理论解释。
- 遗传操作如交叉、变异等易实现。

二进制编码缺点

- 相邻整数的二进制编码可能具有较大的Hamming距离：
  - 例如3和4的二进制编码为011和100，解从3改变到4需要改变所有的位。
  - 这种缺陷将降低遗传算子的搜索效率，这一缺点也被称为Hamming悬崖。
- 需要事先给出解的精度以确定串长，算法缺乏微调功能：
  - 若在算法开始就选取较高的精度，那么位串就会很长，会降低算法效率。
- 在求解高维优化问题时，二进制编码位串将非常长，从而使算法的搜索效率很低。

Gray编码

- Gray编码是将二进制编码通过变换而得到的编码，其目的是克服二进制编码中hamming悬崖的缺点。
- 假设二进制串$B =< b_1, b_2, \ldots, b_n >$，对应的Gray串为$A =< a_1, a_2, \ldots, a_n >$，则从二进制编码到Gray编码的变换为：

$$a_i = \begin{cases} b_i, if\ i = 1 \\ b_{i-1} \oplus b_i, if\ i > 1 \end{cases}$$

即$a_i = \sum_{j=1}^{i} b_j\ (mod 2)$。

- 3的二进制串为011，Gray码为010。
- 4的二进制串为100，Gray码为110。
- 可见相邻的两个整数的编码只有一位不同。

实数编码

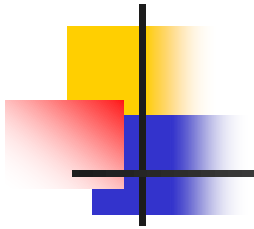
- 对于问题的变量是实向量的情形，可以直接采用十进制进行编码。
- 例如问题解编码为$\{x_1, x_2, \ldots, x_n\}, x_i \in [0,10], x_i \in N$，直接使用实数而不是对应的二进制串表示向量的每个分量。
- 对实数编码的情形，通常需要针对十进制编码的特性，引入特定的遗传算子。
- 对于大部分数值优化问题，通过引入专门设计的遗传算子，采用实数编码比采用二进制编码时算法的平均效率要高。

有序串编码

- 二进制编码用于有序问题困难较多，解的可行性不直观。
  - 假设有编号为A, B, C, D四座城市的旅行商问题。
  - 如果采用二进制编码，用$p_{ij} = 1$表示城市i和j之间的路径被经过，用$p_{AB}p_{AC}p_{AD}p_{BC}p_{BD}p_{CD}$表示解，判断解110011是否合法并不直观。
  - 如果将解编码为$ABCD$表示路线为$A \to B \to C \to D \to A$，就是有序串编码方式，只要每个城市只出现一次就可以保证解合法。
- 有序串编码方案在组合优化问题中使用较多。

编码的重要性

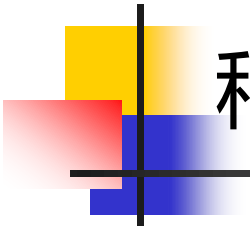
- 由于遗传算法的鲁棒性，它对编码的要求并不苛刻。大多数问题都可以采用二进制编码的形式。
- 编码形式决定了交叉操作，所以编码问题往往又被称为编码-交叉问题。
- 因此作为遗传算法流程中第一步的编码技术对遗传算法的性能好坏起着关键作用。



3.2. 初始群体的设定

个体的产生

- 个体产生的方式：
  - 完全随机的方法：用随机数发生器产生。
  - 基于先验知识：把握最优解所占空间在整个问题空间中的分布范围，然后，在此分布范围内设定初始群体。

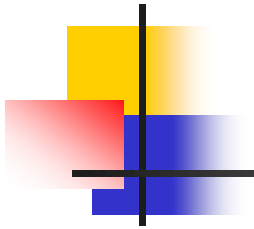


种群的产生

- 种群的产生：
  - 随机产生一定数目的个体，从中挑选最好的个体加到初始群体中。
  - 该过程不断迭代，直到初始群体中个体数目达到了预先确定的规模。

种群规模的确定

- 群体规模太小，遗传算法的优化性能不太好，易陷入局部最优解，出现早熟。
- 群体规模太大，计算量增加，影响算法效率。
- 一般种群规模$N = 20\sim200$。



3.3. 适应度函数的设计

适应度

- 适应度是个体在优化计算中有可能达到或接近于或有助于寻找到最优解的优良程度。
- 适应度较高的个体遗传到下一代的概率较大；而适应度较小的个体遗传到下一代的概率就相对小一些。
- 度量个体适应度的函数称为适应度函数（Fitness Function）。

适应度函数

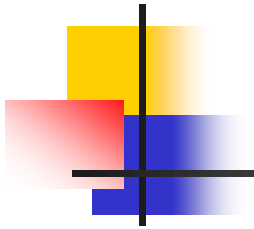
- 遗传算法在优化搜索过程中基本上不用外部信息，仅用适应度函数为寻优依据。
- 遗传算法对适应度函数的唯一要求是该函数不能为负，这使得遗传算法应用范围很广。
- 在具体应用中，适应度函数的设计要根据待求问题而定，而适应度函数的设计直接影响到遗传算法的性能。

目标函数到适应度函数的变换

目标函数$f(x)$	适应度函数$Fit(x)$
$\max f(x), f(x) \geq 0$	$Fit(x) = f(x)$
$\min f(x), f(x) > 0$	$Fit(x) = \frac{1}{f(x)}$
$\max f(x)$	$Fit(x) = \frac{1}{c - f(x)}, c - f(x) > 0$
$\min f(x)$	$Fit(x) = \frac{1}{c + f(x)}, c + f(x) > 0$

适应度函数对停止条件的影响

- 当适应度函数的最大值已知或者次优解适应度的下限可以确定时：
  - 以发现满足最大值或次优解作为遗传算法的迭代停止条件。
- 但在许多组合优化问题中，适应度最大值并不清楚，次优解适应度下限也很难确定：
  - 若发现群体个体进化已趋于稳定状态，即群体中一定比例的个体已完全是同一个体，则终止算法迭代。



3.4. 遗传算子

3个基本遗传算子

- 选择
- 交叉
- 变异

选择算子

- 选择策略对算法性能有至关重要的影响。选择算子分为两个部分：
  - 个体选择概率分配
  - 个体选择方法
- 个体选择概率分配常用方法有适应度比例法、排序方法。
- 个体选择方法常用轮盘赌选择和锦标赛选择方法等。

适应度比例法

- 适应度比例方法也叫做蒙特卡罗法，是最常用的个体选择概率分配方法。
- 在该方法中，每个个体的选择概率和其适应度值成正比。
- 假设群体大小为n，个体i的适应度值为f_i，则被选择的概率p_{si}为$p_{si} = \frac{f_i}{\sum_{j=1}^{n} f_j}$。

排序方法

- 排序方法常见的有线性排序和非线性排序。
- 线性排序：
  - 个体按适应度值从大到小一次排列为$x_1, x_2, \dots, x_N$；
  - 个体x_i分配选择概率为$p_i = \frac{a-bi}{N*(N+1)}$。
- 非线性排序：
  - 个体按适应度值从大到小一次排列为$x_1, x_2, \dots, x_N$；
  - 个体x_i分配选择概率为$p_i = \begin{cases} q*(1-q)^{i-1}, i = 1,2, \dots, N-1 \\ (1-q)^{N-1}, i = N \end{cases}$

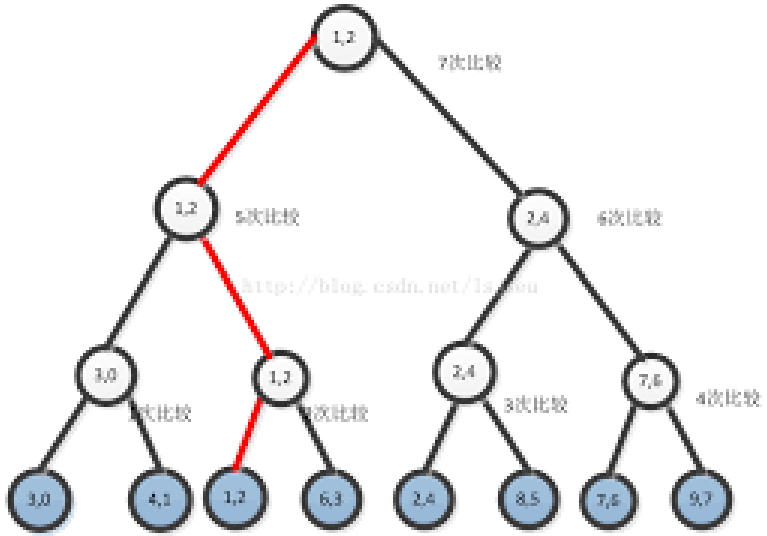
轮盘赌选择

- 假设有n个个体，第i个个体的选择概率为p_{si}，则第i个个体的累积概率$q_i = \sum_{j=1}^{i} p_{si}$，设$q_0 = 0$。
- 从$(0, q_n]$之间产生随机数，如果随机数在区间$(q_{i-1}, q_i]$，则选中个体i。

- 例如上图所示中，如果随机数为0.62，则选中个体2。

锦标赛选择方法

- 锦标赛选择方法：
  - 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定的数量为止。



随机竞争方法

- 随机竞争方法（stochastic tournament）：
  - 每次按轮盘赌选择方法选取一对个体，
  - 让这两个个体进行竞争，适应度高者获胜。
  - 如此反复，直到选满为止。

最佳个体法

- 最佳个体（elitist model）法：
  - 把群体中适应度最高的个体不经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。

基本交叉算子

- 一点交叉（单点交叉）：

p_1	0	1	1	0
p_2	1	1	0	1

⇒

c_1	0	1	1	1
c_2	1	1	0	0

- 二点交叉（两点交叉）：

p_1	0	1	1	0
p_2	1	1	0	1

⇒

c_1	0	1	0	1
c_2	1	1	1	0

- 均匀交叉（一致交叉）：

p_1	0	1	1	0
p_2	1	1	0	1

⇒

c_1	1	1	0	0
c_2	0	1	1	1

部分匹配交叉

- 1. 先随机产生两个交叉点，定义这两点之间的区域为匹配区域。

p_1	2	8	1	3	6	4	0	5	7
p_2	7	8	1	5	0	2	6	3	4

部分匹配交叉

- 2. 交换两个父代的匹配区域

p_1	2	8	1	3	6	4	0	5	7
p_2	7	8	1	5	0	2	6	3	4

c_1	2	8	1	5	0	2	0	5	7
c_2	7	8	1	3	6	4	6	3	4

部分匹配交叉

- 3. 对于匹配区域以外出现的重复编码，要依据匹配区域内的位置逐一进行替换。

c_1	2	8	1	5	0	2	0	5	7
c_2	7	8	1	3	6	4	6	3	4

c_1	4	8	1	5	0	2	6	3	7
c_2	7	8	1	3	6	4	0	5	2

- 本例中$5 \leftrightarrow 3, 0 \leftrightarrow 6, 2 \leftrightarrow 4$

顺序交叉法

- 1. 从父代p_1 (p_2)中随机选一个编码字串，放到子串$c_1(c_2)$的对应位置。

p_1	8	7	2	1	3	9	0	5	4	6
p_2	9	8	3	5	6	7	1	4	2	0

c_1				1	3	9				
c_2				5	6	7				

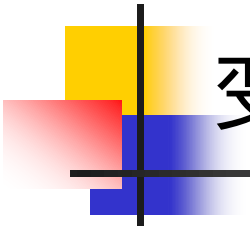
顺序交叉法

- 2. 子代c_1的空余位置从父代p_2中按p_2的顺序选取（与已有编码不重复）。同理可得子代c_2。

p_1	8	7	2	1	3	9	0	5	4	6
p_2	9	8	3	5	6	7	1	4	2	0

c_1				1	3	9				
c_2				5	6	7				

c_1	8	5	6	1	3	9	7	4	2	0
c_2	8	2	1	5	6	7	3	9	0	4



变异

- 变异：将个体染色体编码串中的某些基因值用其他基因值来替换。
- 交叉运算是产生新个体的主要方法，它决定了遗传算法的全局搜索能力；
- 变异运算只是产生新个体的辅助方法，决定了遗传算法的局部搜索能力。

变异算子目的

- 使用变异算子主要有以下两个目的：
  - 改善遗传算法的局部搜索能力；
  - 维持群体的多样性，防止出现早熟现象。
- 常用的变异算子有：
  - 二进制编码：基本位变异算子；
  - 整数编码：两点互换，相邻互换，区间逆转。

基本位变异算子

- 用于二进制编码；
- 依变异概率p_m对个体编码串随机指定的某一位或某几位基因作变异运算。
- 若需要进行变异操作的某一位上的原有基因值为0，则变异操作将其变为1；
- 若原有基因值为1，则变异操作将其变为0。

基本位变异算子举例

- 个体A_1编码如下图所示。
- 只变异一位，假设随机选到最后一位。
- 变异概率为0.1，假设生成的随机数为0.05，执行变异操作。
- 该位原有基因值为0，变为1。

整数编码的变异方法

- 两点互换：
  - 从个体中随机挑选两个逆转点；
  - 将两个逆转点的基因变换。

A_1	1	0	2	3	4	5	6	7
A_1	1	0	2	7	4	5	6	3

整数编码的变异方法

- 相邻互换：
  - 随机选择一个变异位;
  - 将其与其下一位互换。

A_1	1	0	2	3	4	5	6	7

A_1	1	0	3	2	4	5	6	7

整数编码的变异方法

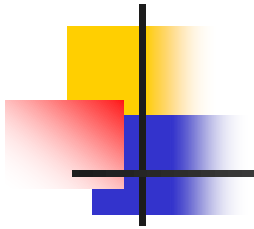
- 区间逆转：
  - 从个体中随机挑选两个逆转点；
  - 将两个逆转点间的基因变换

A_1	1	0	2	3	4	5	6	7
A_1	1	0	2	7	6	5	4	3

整数编码的变异方法

- 单点移动：
  - 随机选择一个变异位；
  - 将其移动到另一个随机选择的变异位。

A_1	1	0	2	3	4	5	6	7
A_1	1	0	3	4	5	2	6	7



3.5. 控制参数设定

控制参数

- 遗传算法中需要选择的运行参数主要有：
  - 个体编码串长度 l；
  - 群体大小M；
  - 交叉概率p_c；
  - 变异概率p_m；
  - 终止代数 T等。

编码长度

- 使用二进制编码来表示个体时，编码串长度的选取与问题所要求的求解精度有关；
- 使用实数编码来表示个体时，编码串长度与决策变量的个数相等；

群体大小

- 群体大小表示群体中所含个体的数量。
- 当M取值较小时，可提高遗传算法的运算速度，但却降低了遗传算法的多样性，有可能会引起早熟现象；
- 而当M取值较大时，又会使得遗传算法的运行效率降低。
- 一般建议的取值范围是$20\sim100$。

交叉概率

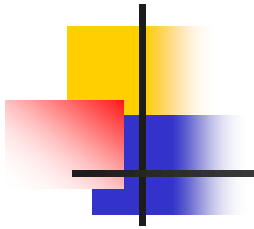
- 将群体内的各个个体随机搭配成对，对每一对个体，以某个概率（称为交叉概率）交换他们之间的染色体：
- 交叉概率一般取值较大；
- 过大容易破坏群体中的优良模式；
- 过小产生新个体的速度较慢；
- 一般建议取值范围是0.4~0.99。
- 也可以使用自适应的思想来确定交叉概率。

变异概率

- 变异概率较大时，可能破坏较好的模式；
- 太小则不利于产生新个体和抑制早熟现象；
- 一般建议范围是0.0001~0.1；
- 也可以使用自适应的思想来确定变异概率。

终止代数

- 一般建议取值范围是100～1000；
- 或者判定出当群体已经进化成熟且不再有进化趋势时就可以终止算法的运行过程。常用的判定准则有：
  - 连续几代个体平均适应度的差异小于某一个极小的阈值；
  - 群体中所有个体的适应度的方差小于某一个极小的阈值。



4. 遗传算法的应用案例

求函数最值

- 设函数$f(x)=x^2$，x为整数，求其在区间$[0,31]$的最大值。

- 使用遗传算法求解问题有哪些要素?
- 如何编码?
- 适应度函数应该如何设计?

求函数最值

- 编码：
  - 采用二进制编码；
  - 由于变量取值在[0,31]，所以编码位数为5位。
- 适应度函数：
  - 由于目标函数值非负，所以适应度函数$Fit(x) = x^2$。
- 初始种群：
  - 为了便于演示，初始种群大小为4，随机产生。

求函数最值

- 遗传算子：
  - 选择算子采用适应度比例法和轮盘赌法选择父代；
  - 交叉算子采用单点交叉和两点交叉，交叉概率为1 ；
  - 变异算子选择基本位变异算子，变异概率为0.1 。
- 子代数：
  - 每次产生4个子代。
- 终止代数：
  - 为了便于演示，终止代数为1代。

初始种群

编号	x	编码	适应度值	选择概率	累积概率
1	13	01101	169	169/1170=0.14	0.14
2	24	11000	576	0.49	0.63
3	8	01000	64	0.06	0.69
4	19	10011	361	0.31	1
和			1170		

- 假设四次的随机数分别是0.12，0.50，0.44，0.72，则1和2，2和4分别进行交叉操作。

交叉

- 对编号1和2的父代进行单点交叉

编号	父代	目标函数值	交换位置	新个体编码	x	目标函数值
1	01101	169	5	01100	12	144
2	11000	576	5	11001	25	625

- 对编号2和4的父代进行二点交叉

编号	父代	目标函数值	交换位置	新个体编码	x	目标函数值
2	11000	576	3，5	11011	27	729
4	10011	361	3，5	10000	16	256

变异

- 假设产生的随机数序列为0.23，0.31，0.75，0.05。
- 由于变异概率是0.1，只对子代4进行基本位变异操作。
- 假设随机选到的位是第2位，则第4个子代变为11000。

编号	新个体编码	目标函数值	变异位	变异后编码	变异后x	目标函数值
4	10000	256	2	11000	24	576

输出结果

- 由于终止条件只执行一代，因此程序终止。
- 最终结果是历史解中目标函数值最大的，即编码为11011的解。
- 解码后x值为27，将其对应结果729输出。

编号	新子代编码	x	目标函数值
1	01100	12	144
2	11001	25	625
3	11011	27	729
4	11000	24	576

求解TSP

- 旅行商问题（Travelling Salesman Problem，TSP）：给定一系列城市和每对城市之间的距离，求解访问每一座城市一次并回到起始城市的最短回路。

- 使用遗传算法求解问题有哪些要素？
- 如何编码？
- 适应度函数应该如何设计？

求解TSP

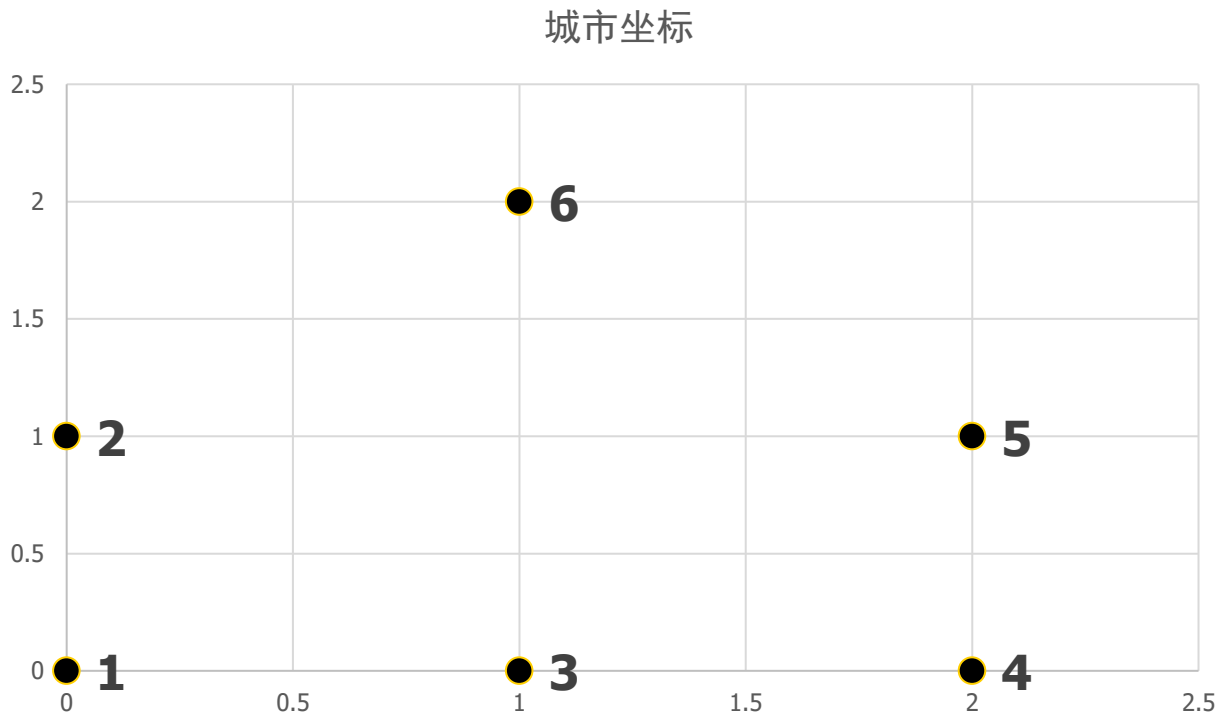
- 编码：
  - 采用有序串编码；
  - 用1到N表示N个城市，编码$1,2,\dots,N$表示从城市1出发，依次经过城市$2,3,\dots,N$，最终回到城市1。
- 适应度函数：
  - 由于目标函数值非负，且求最小值，令$f(x)$表示路线的开销，适应度函数$Fit(x)=\frac{1}{f(x)}$。

求解TSP

- 初始种群：
  - 为了便于演示，初始种群大小为2，随机产生。
- 遗传算子：
  - 选择算子采用适应度比例法和轮盘赌法选择父代；
  - 交叉算子采用顺序交叉法，交叉概率为1；
  - 变异算子选择两点互换，变异概率为0.1。
- 子代数：每次产生2个子代。
- 为了便于演示，终止代数为1代。

问题算例

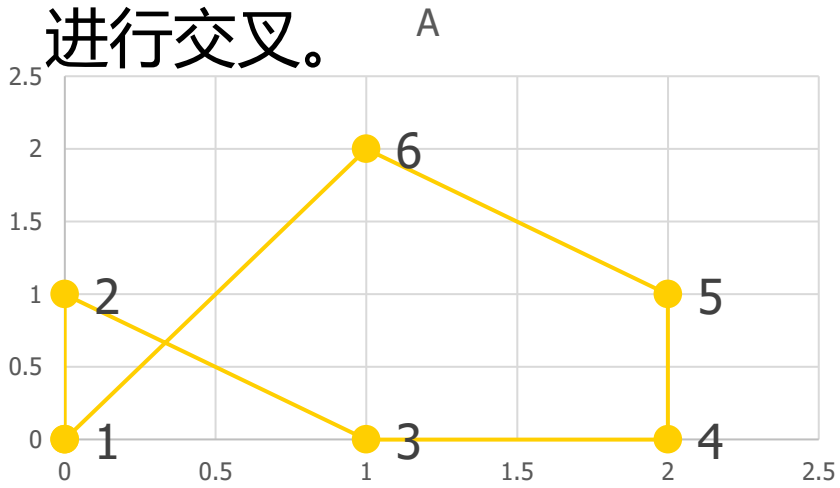
- 有6个城市，编号为1到6

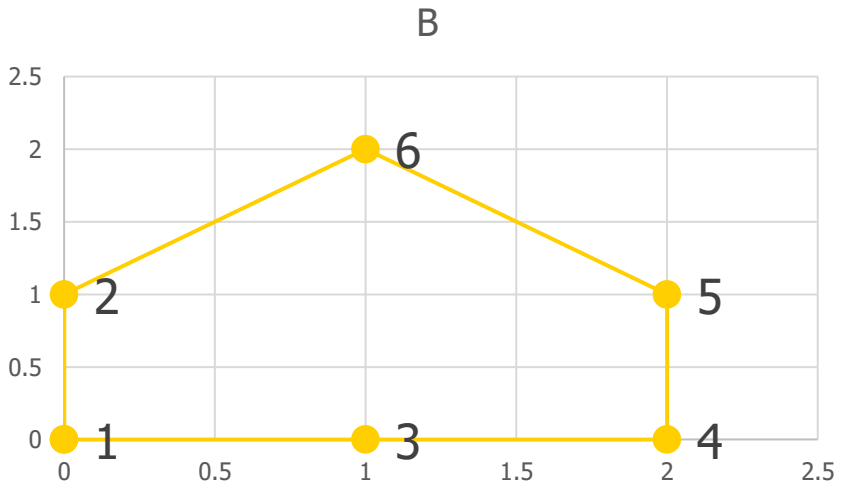


初始种群

编号	编码	目标函数值	适应度值	选择概率	累积概率
A	123456	7	1/7=0.14	0.14/0.31=0.45	0.45
B	134562	6	1/6=0.17	0.55	1

- 假设产生的两个随机数是0.35和0.78，则选择A和B作为父代进行交叉。





交叉

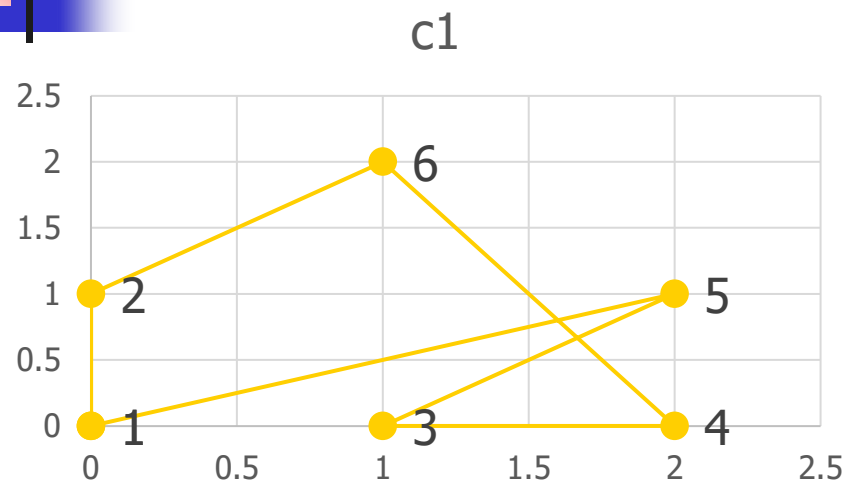
- 对编号A和B的父代进行顺序交叉

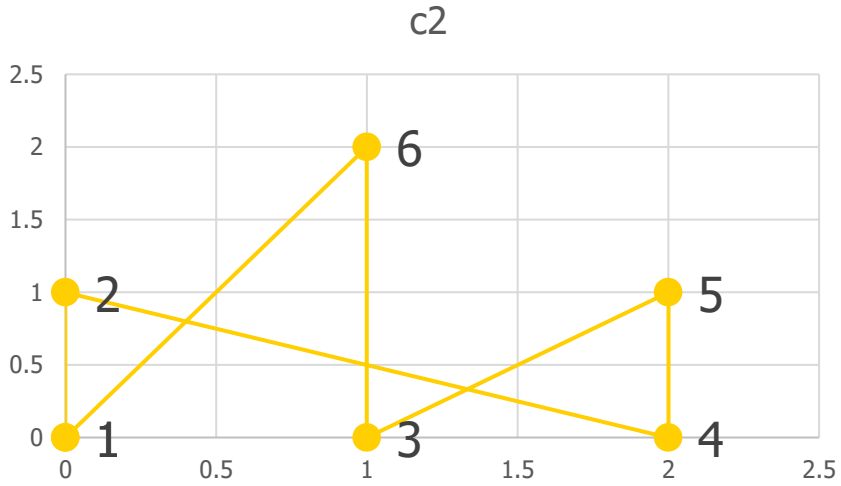
A	1	2	3	4	5	6
B	1	3	4	5	6	2

c_1			3	4		
c_2			4	5		

c_1	1	5	3	4	6	2
c_2	1	2	4	5	3	6

交叉后的子代解



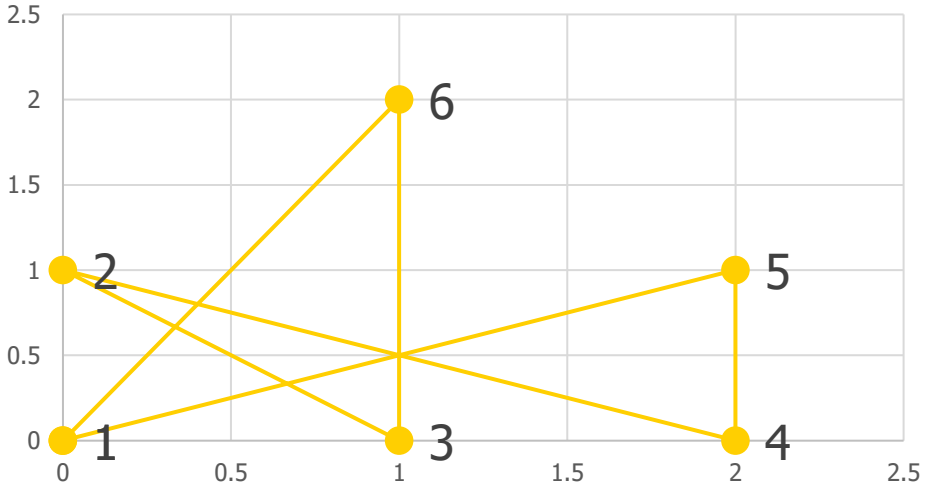


编号	编码	目标函数值
c_1	153462	8
c_2	124536	8

变异

- 假设产生的随机数序列为0.23，0.05。则对子代2进行两点交换操作。
- 假设随机选到的位是第2位和第4位，则第2个子代变为154236，目标函数值为10。

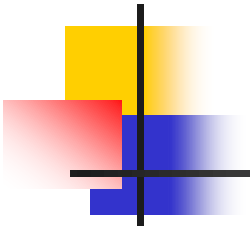
编号	编码	交换位	变异后编码	目标函数值
c_2	124536	2，4	154236	10



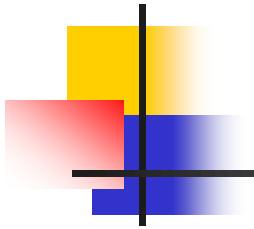
输出结果

- 所以产生的新一代群体为153462和154236。
- 由于终止条件只执行一代，因此程序结束。
- 最终结果是历史解中目标函数值最小的，即编码为134562，目标函数值为6，解码后的路线为$1 \to 3 \to 4 \to 5 \to 6 \to 2 \to 1$，将其输出。

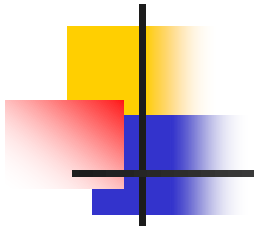
编号	编码	目标函数值
A	123456	7
B	134562	6
c_1	153462	8
c_2	154236	10



回顾



- 1 导言—生物的进化
  - 进化论，遗传理论
  - 进化的基本条件，进化的基本形式
- 2 遗传算法的基本原理
  - 个体，染色体，基因，适应性，群体，交叉，突变，适者生存



- 3 遗传算法的基本要素
  - 参数编码
  - 初始种群的设定
  - 适应度函数的设计
  - 遗传算子
  - 控制参数设定

4 遗传算法构造及基本流程