遗传算法

1.基本原理

由生物进化我们知道，种群是生物进化的基本单位，进化的实质其实就是种群基因频率的改变。基因重组、基因突变、自然选择以及隔离是物种形成的基本环节，通过这些环节，种群产生分化最终新物种形成。

2.作用

计算机中使用遗传算法为了什么呢？你可以先思考一下

3.遗传算法的实现

3.1 遗传算法的一些名词（相关术语）

基因型:性状染色体内部的表现

表现型：染色体决定的性状的外在表现

进化：种群逐渐适应环境，品质不断改良，进化是以种群的形式进行的

适应度：度量某个物种对于生存环境的适应程度

选择：以一定概率从种群中选择若干个体，一般是基于适应度的优胜劣汰

复制：新的细胞继承旧细胞的基因

交叉：基因重组或杂交

变异：复制的时候可能出现差错，产生新的染色体，表现出新的性状

编码：遗传编码是表现型基因型到基因型的映射

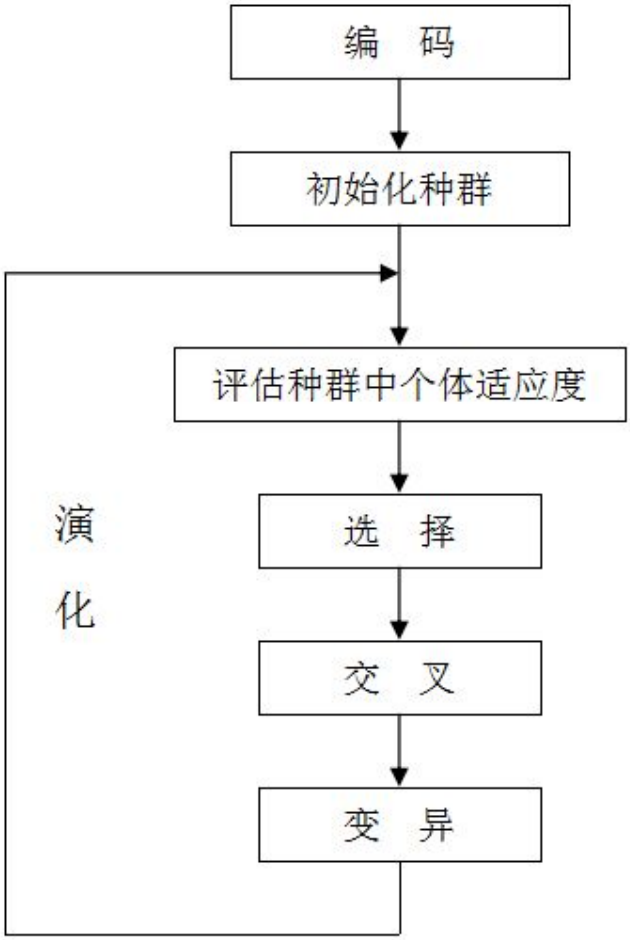
解码：基因型到表现型的映射

个体：染色体带有特征的实体

种群：个体的集合，集合内个体的个数称为种群的大小

3.2 实现步骤

- 1.编码
- 2.解码
- 3.交配
- 4.突变
- 5.倒位
- 6.个体适应度评估
- 7.复制



由此我们可以得出遗传算法的一般步骤：

1. 随机产生种群。
2. 根据策略判断个体的适应度，是否符合优化准则，若符合，输出最佳个体及其最优解，结束。否则，进行下一步。
3. 依据适应度选择父母，适应度高的个体被选中的概率高，适应度低的个体被淘汰。
4. 用父母的染色体按照一定的方法进行交叉，生成子代。
5. 对子代染色体进行变异。

由交叉和变异产生新一代种群，返回步骤2，直到最优解产生。

04 开始我们的进化(具体实现细节)

4.1 先从编码说起

编码是应用遗传算法时要解决的首要问题，也是设计遗传算法时的一个关键步骤。编码方法影响到交叉算子、变异算子等遗传算子的运算方法，大很大程度上决定了遗传进化的效率。

迄今为止人们已经提出了许多种不同的编码方法。总的来说，这些编码方法可以分为三大类：二进制编码法、浮点编码法、符号编码法。下面分别进行介绍：

4.1.1 二进制编码法

就像人类的基因有AGCT 4种碱基序列一样。不过在这里我们只用了0和1两种碱基,然后将他们串成一条链形成染色体。一个位能表示出2种状态的信息量，因此足够长的二进制染色体便能表示所有的特征。这便是二进制编码。如下： 1110001010111

它由二进制符号0和1所组成的二值符号集。它有以下一些优点：

1. 编码、解码操作简单易行
2. 交叉、变异等遗传操作便于实现
3. 合最小字符集编码原则
4. 利用模式定理对算法进行理论分析。

二进制编码的缺点是：对于一些连续函数的优化问题，由于其随机性使得其局部搜索能力较差，如对于一些高精度的问题（如上题），当解迫近于最优解后，由于其变异后表现型变化很大，不连续，所以会远离最优解，达不到稳定。

4.1. 2 浮点编码法

二进制编码虽然简单直观，但明显地。但是存在着连续函数离散化时的映射误差。个体长度较短时，可能达不到精度要求，而个体编码长度较长时，虽然能提高精度，但增加了解码的难度，使遗传算法的搜索空间急剧扩大。

所谓浮点法，是指个体的每个基因值用某一范围内的一个浮点数来表示。在浮点数编码方法中，必须保证基因值在给定的区间限制范围内，遗传算法中所使用的交叉、变异等遗传算子也必须保证其运算结果所产生的新个体的基因值也在这个区间限制范围内。如下所示：

1.2-3.2-5.3-7.2-1.4-9.7

浮点数编码方法有下面几个优点：

1. 适用于在遗传算法中表示范围较大的数
2. 适用于精度要求较高的遗传算法
3. 便于较大空间的遗传搜索
4. 改善了遗传算法的计算复杂性，提高了运算交率
5. 便于遗传算法与经典优化方法的混合使用
6. 便于设计针对问题的专门知识的知识型遗传算子
7. 便于处理复杂的决策变量约束条件

4.1.3 符号编码法

符号编码法是指个体染色体编码串中的基因值取自一个无数值含义、而只有代码含义的符号集如｛A,B,C...｝。 符号编码的主要优点是：

1. 符合有意义积术块编码原则
2. 便于在遗传算法中利用所求解问题的专门知识
3. 便于遗传算法与相关近似算法之间的混合使用。

4.2 为我们的袋鼠染色体编码（具体的例子来看看）

在上面介绍了一系列编码方式以后，那么，如何利用上面的编码来为我们的袋鼠染色体编码呢？首先我们要明确一点：编码无非就是建立从基因型到表现型的映射关系。这里的表现型可以理解为个体特征（比如身高、体重、毛色等等）。那么，在此问题下，我们关心的个体特征就是：袋鼠的位置坐标（因为我们要把海拔低的袋鼠给杀掉）。无论袋鼠长什么样，爱吃什么。我们关心的始终是袋鼠在哪里，并且只要知道了袋鼠的位置坐标（位置坐标就是相应的染色体编码，可以通过解码得出），我们就可以：

1. 在喜马拉雅山脉的地图上找到相应的位置坐标，算出海拔高度。（相当于通过自变量求得适应函数的值）然后判读该不该射杀该袋鼠。
2. 可以知道染色体交叉和变异后袋鼠新的位置坐标。

回到3.1中提的求一元函数最大值的问题。在上面我们把极大值比喻为山峰，那么，袋鼠的位置坐标可以比喻为区间[-1, 2]的某一个x坐标（有了x坐标，再通过函数表达式可以算出函数值 <==> 得到了袋鼠染色体编码，解码得到位置坐标，在喜马拉雅山脉地图查询位置坐标算出海拔高度）。这个x坐标是一个实数，现在，说白了就是怎么对这个x坐标进行编码。下面我们以二进制编码为例讲解，不过这种情况下以二进制编码比较复杂就是了。（如果以浮点数编码，其实就很简洁了，就一浮点数而已。）

我们说过，一定长度的二进制编码序列，只能表示一定精度的浮点数。在这里假如我们要求解精确到六位小数，由于区间长度为2 - (-1) = 3 ,为了保证精度要求，至少把区间[-1,2]分为3 × 10^6等份。又因为

> 2^21 = 2097152 < 3*10^6 < 2^22 = 4194304

所以编码的二进制串至少需要22位。

把一个二进制串(b0,b1,....bn)转化为区间里面对应的实数值可以通过下面两个步骤：

1. 将一个二进制串代表的二进制数转化为10进制数：

$$(b_0 \cdots b_{20} b_{21})_2 = (\sum_{i=0}^{21} b_i \cdot 2^i)_{10} = x'$$

2. 对应区间内的实数：

$$x = -1 + x' \frac{(2-(-1))}{2^{22}-1}$$

例如一个二进制串(1000101110110101000111)2通过上面换算以后，表示实数值0.637197。

4.3 评价个体的适应度--适应度函数（fitness function）

前面说了，适应度函数主要是通过个体特征从而判断个体的适应度。在本例的袋鼠跳中，我们只关心袋鼠的海拔高度，以此来判断是否该射杀该袋鼠。这样一来，该函数就非常简单了。只要输入袋鼠的位置坐标，在通过相应查找运算，返回袋鼠当前位置的海拔高度就行。

适应度函数也称评价函数，是根据目标函数确定的用于区分群体中个体好坏的标准。适应度函数总是非负的，而目标函数可能有正有负，故需要在目标函数与适应度函数之间进行变换。

评价个体适应度的一般过程为：

1. 对个体编码串进行解码处理后，可得到个体的表现型。
2. 由个体的表现型可计算出对应个体的目标函数值。
3. 根据最优化问题的类型，由目标函数值按一定的转换规则求出个体的适应度。

4.4 射杀一些袋鼠--选择函数（selection）

遗传算法中的选择操作就是用来确定如何从父代群体中按某种方法选取那些个体，以便遗传到下一代群体。选择操作用来确定重组或交叉个体，以及被选个体将产生多少个子代个体。前面说了，我们希望海拔高的袋鼠存活下来，并尽可能繁衍更多的后代。但我们都知道，在自然界中，适应度高的袋鼠越能繁衍后代，但这也是从概率上说的而已。毕竟有些适应度低的袋鼠也可能逃过我们的眼睛。

那么，怎么建立这种概率关系呢?

下面介绍几种常用的选择算子:

1. 轮盘赌选择（Roulette Wheel Selection）：是一种回放式随机采样方法。每个个体进入下一代的概率等于它的适应度值与整个种群中个体适应度值和的比例。选择误差较大。
2. 随机竞争选择（Stochastic Tournament）：每次按轮盘赌选择一对个体，然后让这两个个体进行竞争，适应度高的被选中，如此反复，直到选满为止。
3. 最佳保留选择：首先按轮盘赌选择方法执行遗传算法的选择操作，然后将当前群体中适应度最高的个体结构完整地复制到下一代群体中。
4. 无回放随机选择（也叫期望值选择Excepted Value Selection）：根据每个个体在下一代群体中的生存期望来进行随机选择运算。方法如下:

   （1） 计算群体中每个个体在下一代群体中的生存期望数目N。

   （2） 若某一个体被选中参与交叉运算，则它在下一代中的生存期望数目减去0.5，若某一个体未 被选中参与交叉运算，则它在下一代中的生存期望数目减去1.0。

   （3） 随着选择过程的进行，若某一个体的生存期望数目小于0时，则该个体就不再有机会被选中。
5. 确定式选择：按照一种确定的方式来进行选择操作。具体操作过程如下:

   （1） 计算群体中各个个体在下一代群体中的期望生存数目N。

   （2） 用N的整数部分确定各个对应个体在下一代群体中的生存数目。

   （3） 用N的小数部分对个体进行降序排列，顺序取前M个个体加入到下一代群体中。至此可完全确定出下一代群体中M个个体。
6. 无回放余数随机选择：可确保适应度比平均适应度大的一些个体能够被遗传到下一代群体中，因而选择误差比较小。
7. 均匀排序：对群体中的所有个体按期适应度大小进行排序，基于这个排序来分配各个个体被选中的概率。
8. 最佳保存策略：当前群体中适应度最高的个体不参与交叉运算和变异运算，而是用它来代替掉本代群体中经过交叉、变异等操作后所产生的适应度最低的个体。
9. 随机联赛选择：每次选取几个个体中适应度最高的一个个体遗传到下一代群体中。
10. 排挤选择：新生成的子代将代替或排挤相似的旧父代个体，提高群体的多样性。

下面以轮盘赌选择为例给大家讲解一下:

假如有 5 条染色体，他们的适应度分别为 5 、 8 、 3 、 7 、 2 。

那么总的适应度为：F = 5 + 8 + 3 + 7 + 2 = 25。

那么各个个体的被选中的概率为:

α1 = (5 / 25) * 100% = 20%

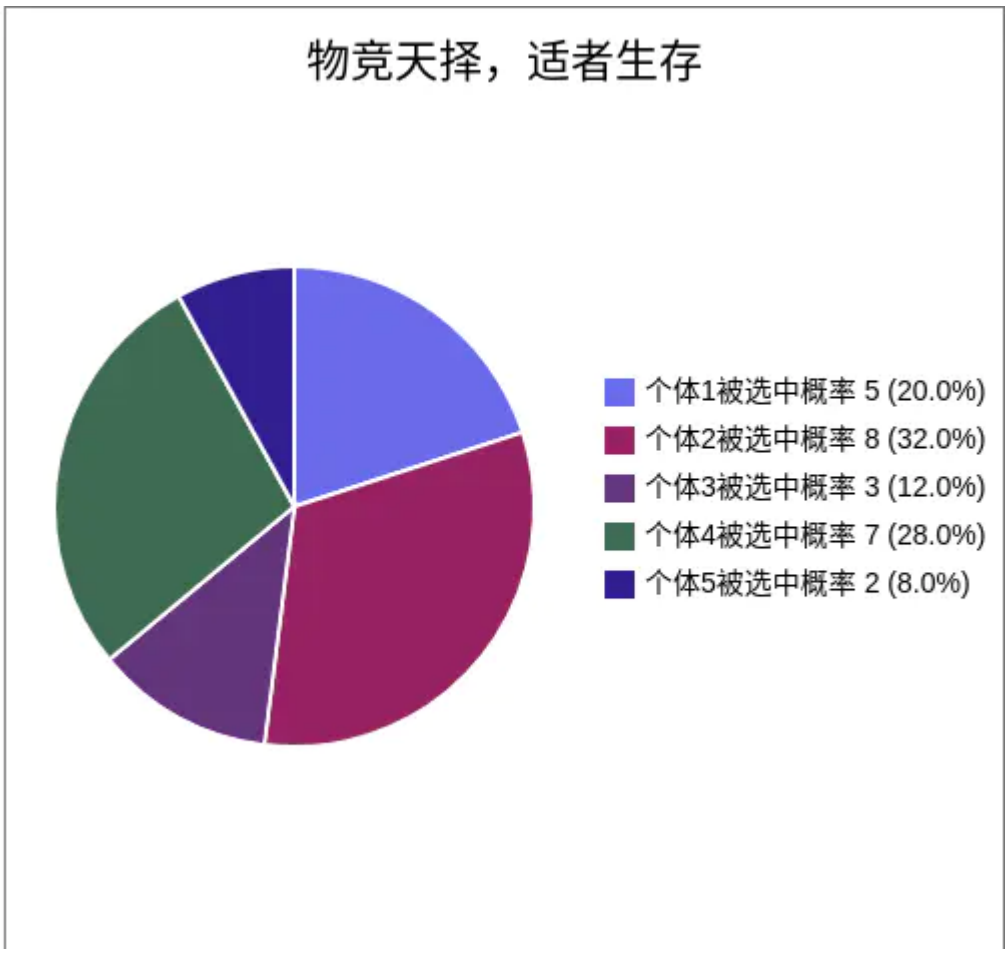
α2 = (8 / 25) * 100% = 32%

α3 = (3 / 25) * 100% = 12%

α4 = (7 / 25) * 100% = 28%

α5 = (2 / 25) * 100% = 8%

所以转盘如下：



当指针在这个转盘上转动，停止下来时指向的个体就是天选之人啦。可以看出，适应性越高的个体被选中的概率就越大。

4. 5 遗传--染色体交叉(crossover)

遗传算法的交叉操作，是指对两个相互配对的染色体按某种方式相互交换其部分基因，从而形成两个新的个体。

适用于二进制编码个体或浮点数编码个体的交叉算子：

1. 单点交叉（One-point Crossover）：指在个体编码串中只随机设置一个交叉点，然后再该点相互交换两个配对个体的部分染色体。
2. 两点交叉与多点交叉：

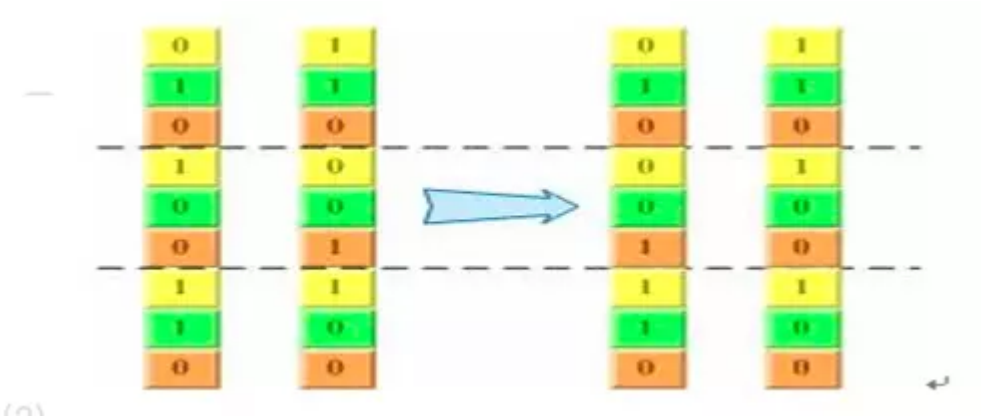
   (1) 两点交叉（Two-point Crossover）：在个体编码串中随机设置了两个交叉点，然后再进行部分基因交换。

   (2) 多点交叉（Multi-point Crossover）
3. 均匀交叉（也称一致交叉，Uniform Crossover）：两个配对个体的每个基因座上的基因都以相同的交叉概率进行交换，从而形成两个新个体。
4. 算术交叉（Arithmetic Crossover）：由两个个体的线性组合而产生出两个新的个体。该操作对象一般是由浮点数编码表示的个体。

咳咳，根据国际惯例。还是抓一个最简单的二进制单点交叉为例来给大家讲解讲解。

二进制编码的染色体交叉过程非常类似高中生物中所讲的同源染色体的联会过程——随机把其中几个位于同一位置的编码进行交换，产生新的个体。

来看看二进制的交叉



4.6 变异--基因突变(Mutation)

遗传算法中的变异运算，是指将个体染色体编码串中的某些基因座上的基因值用该基因座上的其它等位基因来替换，从而形成新的个体。

例如下面这串二进制编码：

101101001011001

经过基因突变后，可能变成以下这串新的编码：

00110101**1**1011001

以下变异算子适用于二进制编码和浮点数编码的个体：

1. 基本位变异（Simple Mutation）：对个体编码串中以变异概率、随机指定的某一位或某几位仅因座上的值做变异运算。
2. 均匀变异（Uniform Mutation）：分别用符合某一范围内均匀分布的随机数，以某一较小的概率来替换个体编码串中各个基因座上的原有基因值。（特别适用于在算法的初级运行阶段）
3. 边界变异（Boundary Mutation）：随机的取基因座上的两个对应边界基因值之一去替代原有基因值。特别适用于最优点位于或接近于可行解的边界时的一类问题。
4. 非均匀变异：对原有的基因值做一随机扰动，以扰动后的结果作为变异后的新基因值。对每个基因座都以相同的概率进行变异运算之后，相当于整个解向量在解空间中作了一次轻微的变动。
5. 高斯近似变异：进行变异操作时用符号均值为 P 的平均值，方差为P**2的正态分布的一个随机数来替换原有的基因值。

5 一个更简单易懂的文章

点击这里去看文章

当然，我等下也会去写个python实现的例子给你看看，代码难度不大，关键是理解其中的原理和过程。