# 遗传算法

一、绪论  
 1、遗传算法（Genetic Algorithm）是受生物进化学说和遗传学说的启发发展起来的，生物在进化的过程中：1、不断繁殖 2、生存竞争 3、适者生存 4、遗传和变异。

2、{遗传算法不是直接把解求出来，得到的解是不是最优解还不敢肯定，他只是一步一步逼近最优解，他是这样一种寻优的技术}

3、遗传算法的研究对象：研究各种非线性、多变量、多目标、复杂的自适应系统问题。

（这可是一把杀牛刀，对复杂问题特别拿手，像没有特别明确的函数关系的问题，特别拿手）

4、遗传算法的应用

光学专业、生物物理专业

1. 实验数据处理
2. 图像处理
3. 数值拟合
4. 固体材料的模型求数值解
5. 人工什么设计的研究

5、遗传算法的基本特征

（1）智能式搜索：依据适应度（目标函数）进行智能搜索。

（2）渐进式搜索：利用复制、交换、突变等操作，使下一代结果优于上一代。

（3）全局最优解：采用交换和突变操作产生新个体，使得搜索得到的优化结果逼近全局   
 最优解（厉害了！）

（4）黑箱式结构：根据问题的特性进行编码（输入）和确定适应度（输出），具有只考

虑输入与输出关系的黑箱式结构，并不深究输入与输出的关系的原因。

（5）通用性强：不要求明确的数学表达式，只需要一些简单的原则要求，可应用于解   
 决离散问题、函数关系不明确的复杂问题。（因为通用性很强，所以八   
 仙过海各显神通，不同的算子每个人编写的程序，效率是不同的）

（6）并行式运算：每次迭代计算都是对群体中的所有个体同时进行运算，是并行式运算   
 方式，搜索速度快。（同时得到很多解，并行运算。）

{

例子：Michalewicz Function f(x,y)=21.5+xsin(4πx)+ysin(20πy)求他的全局最大值。（0-5）

1. 随机产生一些初始解
2. 经过几代的运行，解就在慢慢聚集，一些离散的解开始看不见了。解开始变优，局部最优解已经开始有了。
3. 循环十次之后全局最优解特别多了，局部开始减少了。
4. 再接着运行，差不多99%的解都聚集在全局最优解那里。
5. 求解完毕！  
   注：这只是简单例子循环几代就可以找到最优解，有的问题循环上万次找不到好的解.

}

6、遗传算法和遗传和遗传规划特别有效的领域  
 1）结构优化设计 2）人工智能 3）复杂问题优化 4）复杂系统分析 5）综合应用

（遗传算法和遗传和遗传规划可以改进）-算子不同结果不同

二、基本运算过程

<一>

1、编码、随机产生初始群体

（编码编的好不好关系到你解决问题的效率，编码后随机产生初始个体，个体就是编码表示的）

2、个体评价、选择、确定是否输出（适应度验算，看解是否到达要求）  
 （有的时候运气好第一代就得到全局最优解，但是这个可能性不大。之后循环选择，把差的筛选掉。）

3、随机交叉运算

（对已经存在下的比较优秀的个体进行随机交叉）

4、随机变异交叉运算

（对已经存在下的比较优秀的个体进行随机变异）

5、转向个体评价，开始新循环

（这里就是第二代的解，重新进行循环，寻找最优解。）

<二>

1. 编码

遗传算法的工作对象是字符串，对字符串的编码要求：

1. 字符串要反映所研究问题的性质
2. 字符串的表达要便于计算机处理
3. 适应度

在遗传算法中，衡量个体优劣的尺度是适应度。根据适应度的大小决定个体繁殖的数量、或者决定其是否消亡。适应度是驱动遗传算法的动力。

1. 复制

复制算子将优良个体在下一代予以繁殖，体现了“适者生存”的自然选择原则。

个体被复制的数量是按照适应度的大小决定的，适应度大的复制得多适应度小的复制得少，群体中个体总数保持不变。（跟生物还是不一样的，群体总数不变。）

[复制方法]

轮盘方式：选择复制对象，直到满足所需要的个体数目。（跟那个购物街转的盘似的，指针停在大奖的区域概率比较小。）

（选择的时候可以分初选阶段，中选阶段，精选阶段）

{

例子：

个体序号I 1 2 3 4 5 6 7 8 9 10

适应度xi 8 2 17 7 2 12 11 7 3 7  
 适应度累计值si 8 10 27 34 36 48 59 66 69 76

随机数R 23 49 76 13 1 27 57  
 被选中的个体号I 3 7 10 3 1 3 7

<

累计适应度最大76，在76之间产生随机数，随机数个数看需求，随机数与si比较累计适应的比随机数大选择出来，依次比较下去选出个体。（选出的个体是比较优秀的）（其实就是转盘指针选中了最左边部分）

>

}

1. 交换（体现生物中的杂交原理）

交换是遗传算法中产生新个体的主要手段，将两个个体的部分字符互相交换。

【~】 执行交换的个体是随机选则的

【~】 交换的概率一般为50% ~ 80%（初选阶段概率可以大点，往后可以依次降低）

{选出的个体不能重复}

1. 变异

变异是遗传算法中产生新个体的另一中方法，借鉴了 生物学中的突变现象，将别选择的个体的部分字符进行补运算。

1. 终止条件

遗传算法是一种反复迭代的搜索方法，他通过多次进化逐渐逼近最优解。

【】最常用的终止方法是迭代的代数

【】当目标函数是最优目标值时，可采用控制偏差的方式实现终止。（满足所需 条件，明确具有最优值）

【】第三种终止方法是检查适应度的变化。（连续100代平均适应度没有变化的话重新编码一般这个用在问题比较复杂的时候）

1. 模式理论(是二进制编码，还是十进制编码，还是其他编码啊就是什么模式)

【】编码的左边数字越小，则其适应度将越大。这种字符串（编码）的形式特征叫做编码的模式。（与排序有关）

【】模式中确定的字符的个数称为模式的阶次（Order），记作O(H).

例如：O(000\*\*)=3。

【】模式中最前面和最后面两个确定字符的距离称为模式的长度（Defining Length）记作∂(H) 例如∂(00\*0\*) = 3 ∂(\*0\*\*\*) = 0

【】若字符串的长度为L，字符的取值为k种，则字符串的模式数目为

n1 =(k+1)^k

【】遗传过程的模式数目及模式定理

1. 复制时的模式数目  
   模式H以常数c繁殖到第t+1代时其数目为：  
   m(H,t+1) = m(H,1)\*(1+c)^t
2. 交换时的模式数目  
   在交换时模式H存活的概率为：  
   P = 1-Pc\*∂(H) /(L-1) (Pc为交换的概率)
3. 突变时的模式数目  
   在突变时模式H存活的概率为：  
   P = (1-Pm)∂(H) ≈ 1-0(H)\*Pm (Pm为突变的概率)
4. 模式定理  
   经过复制、交换、突变，模式H在下一代群体中拥有的个体数目为  
   m(H,t+1)=m(H,t)\*f(H)/(∑fi/n)\*[1-Pc\*∂(H)/(L-1)-0(H)\*Pm]  
    (f为平均适应度)  
   这个定理说明：长度短、阶次低、平均适应度大的模式在遗传算法循环中呈指数增长。（适应性强的在后面就会凸显出来，看这个搜索过程不是盲目的搜索，一定会找出最优的。-理论支持）

三、算子及参数技术

1、遗传算子

（1）选择算子

（2）交叉算子

（3）复制算子

（4）变异算子

（5）重排序算子

（6）倒序算子

（7）生态算子

（8）异性算子

2、参数技术

（1）编码与解码

（2）适应度函数与评估

（3）选择与复制

（4）交换

（5）变异

（6）其他算子（操作）

<编码与解码>

1. 编码原则

原则一：有意义积木块编码原则

应使用能易于产生与所求问题相关的、且具有低阶、段定义长度模式的

编码方案

原则二：最小字符集编码原则

应使用能使问题得到自然表示或描述的具有最小编码字符集的编码方案  
 2）编码方法

十进制编码：直接使用十进制数编码，使用较少。  
 优点：无需解码  
 缺点：突变的可能性太多，有9种；交换较粗略，在多代循环后收敛太慢。

（不仅初选的时候可以变，我们也可以初选的时候用十进制编码，后面的用其他编码）

二进制编码（正整数转换而来）：  
 缺点：需要解码，增大了运算量

优点：突变时取反即可，唯一确定；交换时解可精确变化，在多代循环后收敛速度较快

【】十进制正整数转化为二进制码整数

直接二进制码（一般比较麻烦）

间接二进制码：每一个二进制码全部为正整数

1. 码的位数公式

2^m < (Umax-Umin)/d<=2^m+1

其中d为问题的解精确度，定义域为[Umin,Umax]，则码的位数为m+1

{

例子：

某问题的某个参数取值范围是[-250，250]，要求精确度为0.00001

由于

2^25=33554432=3.3554432\*10^7

2^26=67108864=6.7108864\*10^7

3.35554432\*10^7<(250-(-250)/0.00001)=5\*10^7<6.7108864\*10^7

该参量的二进制码所需位数为26位。（没有符号位没有小数点位）

}

1. 解码公式

x = Umin + (Umax-Umin)/(2^m-1)\*∑(bi\*2^(i-1))

{

例子：

某个问题的某个参量二进制码为10010100001111010100010011，其取值范围为[-250,250],则其真值为

x=-250+500/(2^26-1)∑(i从1到26)(bi\*2^i-1)=39.52994

}

(二进制码相对于十进制码来说精确性强但慢，有没有既快又精确的编码方式呢-有，格雷码)

格雷码：增强遗传算法的局部搜索能力，便于对连续函数进行局部空间搜索。使用非常广泛。解码时，先将格雷码转化为二进制码，再将二进制码转化为十进制真实数；或直接将格雷码转化为十进制真实参数。

G=g1g2……gm-1gm

B=b1b2……bm-1bm

二进制码转化为格雷码

g1 = b1

gi = bi-1异或bi (相同为0不同为1)

{

例子：

将二进制数码1101转化为格雷码

1101B = b1b2b3b4 b1=1 b2=1 b3=0 b4=1 g1=b1

g2=b1异或g1=1异或1 = 0

所以有: 1101B = 1011G

}

格雷码转化为二进制码

b1 = g1   
 bi=bi-1异或gi （i=2,3,….m-1,m）(左边第几位)

{

例子：

将格雷码1011转化为二进制数码

1011G= g1g2g3g4

g1=1 g2=0 g3=1 g4=1

b1=g1=1

b2=b1异或g2=1异或0=1

所以有：1011G= 1101B

}

浮点数编码:个体的每个基因值用某一范围内的一个浮点数表示，个体的编码长度等于其决策变量的个数。这种编码方法使用的是决策变量的真实值，故浮点编码方法也叫真值编码方法。

浮点数编码方法主要用于一些多维、高精度要求的连续函数优化问题。

符号编码：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。（比如：饮食经验方式，旅游方式，机器等（指代的含义不是数值含义））

多参数级联编码：将各个参量分别编码，然后将它们按一定的顺序链接在一起的编码方法。

多参数交叉编码：将各个参量分别编码，然后将它们起主要作用的码集中在一起，并按一定的顺序联接的编码方法。（不把所有鸡蛋放在同一个篮子里面）

（遗传算法不是一成不变的，除了这些编码，我们还可以自己做编码，做更加有效率的编码）

<适应度函数与评估>

通过适应度函数可以评价个体的优劣，并确定是否输出目前最优解，或者继续循环。

对个体编码串解码后，可得到个体值；由个体真值可求出个体的目标函数值；由最优化问题的类型，由目标函数值按一定的转换规则可求出个体的适应度。

{

若求解多元的、高阶的、非线性的方程组，该如何设置适应度？

两边取绝对值，相减为零。 相减的值越接近零，解越接近，为零就是精确解。

}

<交叉>

a)实值重组（real valued recombination）

1)离散重组（discrete recombination）

2)中间重组（intermediate recombination）

3)线性重组（linear recombination）

4)扩展线性重组（extended linear recombination）。

b)二进制交叉（binary valued crossover）

1)单点交叉（single-point crossover）

2)多点交叉（multiple-point crossover）

3)均匀交叉（uniform crossover）

4)洗牌交叉（shuffle crossover）

5)缩小代理交叉（crossover with reduced surrogate）。

最常用的交叉算子为单点交叉(one-point crossover)。具体操作是:在个体串中随机设定一个交叉点，实行交叉时，该点前或后的两个个体的部分结构进行互换，并生成两个新个体。下面给出了单点交叉的一个例子：

个体A：1 0 0 1 ↑1 1 1 → 1 0 0 1 0 0 0 新个体

个体B：0 0 1 1 ↑0 0 0 → 0 0 1 1 1 1 1 新个体

<变异>

变异算子的基本内容是对群体中的个体串的某些基因座上的基因值作变动。依据个体编码表示方法的不同，可以有以下的算法：

a)实值变异

b)二进制变异。

一般来说，变异算子操作的基本步骤如下：

a)对群中所有个体以事先设定的变异概率判断是否进行变异

b)对进行变异的个体随机选择变异位进行变异。