第五章 遗传算法

5.1 概述

进化算法（Evolutionary Computation）主要涵盖一下三方面的研究：遗传算法（GA Genetic Algorithms）、进化策略（ES Evolution Strategies）和进化规划（EP Evolutions Programming）。作为进化计算的一个重要分支，遗传算法提供了一种有效的随机搜索和优化方法，因而它是影响最大的进化算法之一。

进化算法是一门新兴的科学，它是研究仿照生物进化自然选择过程中所表现出来的优化规律和方法，对复杂的工程技术领域或其它领域提出传统优化理论和方法难以解决的优化问题，进行优化计算、预测和数字寻优等方面的

计算方法。

遗传算法是20世纪60年代由美国Michigan大学的J.H.Holland教授提出的。1975年Holland出版了第一本系统论述遗传算法和人工智能自适应系统的专著《自然系统和人工智能的自适应性》，标志着遗传算法的诞生。

遗传算法是建立在自然选择和自然遗传学机理基础上的迭代自适应概率性搜索算法，达尔文的“适者生存”基本理论贯彻于整个算法。它采用选择、复制、交叉和变异等基因操作作为其模型，并将这些算子应用于抽象的染色体上，使得（父代）优良品质得到保留并重新组

合得到更好的染色体。随着这些操作不断进行，好的特征被（后代）继承下来，差的特征逐渐被淘汰，这样，整个群体就得到了进化，以此向最优解收敛，从而达到对最优化问题的求解。

遗传优化与传统优化

优化问题是指在满足等式或不等式表示的约束条件下的多变量函数的最小化和最大化问题。

传统的优化方法主要有三种类型：解析法、枚举法和随机法。其中解析法是传统方法中研究得最多的一种。解析法又可以分为直接法和间接法。间接法是通过令目标函数的梯度为零，进而求解一组非线性方程来寻找局部极值的方法；直接法是按照梯度提供的信息确定一个寻优方向，并按该方向逐步进行寻优的方法，如最速下降法。具体描述如下：

考虑无约束最小化问题

$$\min_{x\in D} f(x) \qquad (5.1)$$

其中 $D\in\mathbf{R}^n$, $f:\mathbf{R}^n\to\mathbf{R}$ 。定义

$$\nabla f(x)=\left(\frac{\partial f(x)}{\partial x_1}\quad\frac{\partial f(x)}{\partial x_2}\quad\cdots\quad\frac{\partial f(x)}{\partial x_n}\right)^T$$

称 $\nabla f(x)$为函数f(x)沿着向量x的梯度。如果已知函数f(x)在定义域D内达到最小值，则最小值必在据点，即D中满足如下方程组的解集中达到：

$$\nabla f(x)=0$$

该方法称为间接法。

直接法的代表是最速下降法，它是一种迭代方法，极值点按梯度的负方向反复迭代而得到，具体算法如下：

最速下降法算法：

Step1:选取初始点x_0，给定允许误差$\varepsilon > 0$，令

k=0。

Step 2: 计算 $d_k = -\nabla f(x_k)$，若 $\|d_k\| < \varepsilon$，则迭代终止，x_k即为最优问题（5.1）的近似最优解；否则，转入Step3。

Step 3：进行一维搜索，求取 λ_k 和下一步迭代点x_{K+1}。

$$f(x_k + \lambda_k d_k) = \min_{\lambda \geq 0} f(x_k + \lambda d_k)$$

$$x_{k+1} = x_k + \lambda_k d_k \qquad (5.2)$$

令k:=k+1，返回Step 2.

定理:如果将最速下降法应用于正定二次型函数的最优化问题

$$\min_{\lambda\geq 0} f(x)=\frac{1}{2}x^TQx+b^Tx+c$$

其中 $x\in\mathbf{R}^n, Q\in\mathbf{R}^{n\times n}$ 为正定矩阵，而 $b\in\mathbf{R}^n, c\in\mathbf{R}$。则

$$\lambda_k=\frac{\nabla^T f(x_k)\nabla f(x_k)}{\nabla^T f(x_k)Q\nabla f(x_k)} \qquad (5.3)$$

例：用最速下降法求解问题

$$\min f(x)=4x_1^2+x_2^2$$

其中 $x=(x_1 \quad x_2)^T$，取初始点为 $x_0=(1,1)^T$,允许误差为 $\varepsilon=0.1$

解：该问题是正定二次函数的最下优化问题，其

中

$$Q=\begin{bmatrix}8 & 0\\ 0 & 2\end{bmatrix},\quad b=\begin{bmatrix}0\\ 0\end{bmatrix},\quad c=0$$

f在点$x=(x_1,x_2)^T$的梯度是 $\nabla f(x)=(8x_1 \quad 2x_2)^T$

第一次迭代：令搜索方向 $d_0=-\nabla f(x_0)=(-8,-2)^T$，

$$\|d_0\|=\sqrt{64+4}=2\sqrt{7}>\varepsilon$$

从x_0出发沿着x_0按（5.2）作一维搜索，由于是对二次型函数求解最优化问题，故可以直接按计算得到 $\lambda_0=\frac{68}{520}=0.130769$，

$$x_1=(1,1)^T+0.130769(-8,)-2)^T=(-0.046152,\ 0.738462)^T$$

第二次迭代：令 $d_1=-\nabla f(x_1)=(0.369216,\ -1.476924)^T$

则$\|d_1\| = \sqrt{2.18305} = 1.522375 > \varepsilon$，从点$x_1$出发沿着$d_1$作一维搜索，按（5.3）和（5.2）计算得

$$x_2 = (0.101537, 0.147682)^T$$

第三次迭代：令 $d_2 = -\nabla f(x_2) = (-0.812296, -0.295364)^T$

$$\|d_2\| = \sqrt{0.747056} = 0.864329 > \varepsilon$$

按（5.3）和（5.2）计算得

$$x_3 = (-0.09747, 0.107217)^T$$

第四次迭代：令 $d_3 = -\nabla f(x_3) = (0.077967, -0.214434)^T$

$$\|d_3\| = \sqrt{0.052062} = 0.228171 > \varepsilon$$

$$x_4 = (0.019126, 0.027816)^T$$

第五次迭代：令 $d_4 = -\nabla f(x_4) = (-0.153008, -0.055632)^T$

$$\|d_4\| = \sqrt{0.026506} = 0.162807 > \varepsilon$$

按（5.3）和（5.2）计算得

$$x_5 = (-0.001835, 0.020195)^T$$

此时，$\|\nabla f(x_5)\| = \sqrt{0.001847} < \varepsilon$，已满足要求，故$x_5$是该优化问题的近似最优解。事实上，真正最优解为$x^* = (0, 0)^T$。

从上面可以看出，解析法求解优化问题，要求目标函数必须连续、光滑和可导。通过解析法寻优只能得到获得局部最优解。

遗传算法的特点：

1)遗传算法以决策变量的编码作为运算对象。

2)遗传算法对解空间的搜索是基于一组可能解组成的种群而进行的，而初始种群通常由随机个体产生。

3)遗传算法采用关于每个可能解的评价来引导搜索。

4)遗传算法的各种操作是基于概率性规则而不是确定性规则进行的

5)遗传算法对目标函数的类型和性质基本没有要求，对函数的连续性、可微性和光滑不作要求。目标函数可以是解析表达的显函数，也可以是映射矩阵或神经网络等各种隐函数，所以，具有很强的通用性。

5.2 遗传算法的基本原理

遗传算法的基本思想是，由一个个体（由编码表示，如二进制编码）（Individual）组成一个群体（Population)。每一个个体代表问题的一个潜在的解（可行解）。每一个个体均受到评价并得到相应的适应值（Fitness），以反映该个体对环境的适应度。种群中的部分个体需要经过遗传操作（Genetic Operation)并由此产生新个体。遗传操作其实际是

随机变换，主要有选择、复制、交叉和变异。受遗传算子直接操作的个体称为父代（Parent）个体，新产生的个人称为后代或子代（Offspring）个体。从父代种群中选取比较优秀的个体就形成了新的种群，新的种群称为新的父代。这样的过程循环往复，经过若干代后，算法就收敛到一个最优个体，即优化问题的最优解或者次优解。

遗传算法的基本操作

遗传算法有五个基本组成部分（1）问题可行解的遗传表示；(2)创建解的初始种群的方法；（3）根据个体适应值对其进行优劣判断的评价函数；（4）用来改变复制过程中产生的子代个体的遗传算子；（5）遗传算法 的参数。

可行解是由很多个体所组成。遗传算法的个体也称为染色体，遗传算法中的个体以串位式的编码表示（如二进制编码），编码中的每一位称为基因。在下面的阐述中，将不加区别地使用个体、染色体和串位这三个术语。下面结合一个例子来简单介绍遗传算法的基本操作。

设待求解的优化问题是

$$\max_{x\in D} f(x)$$

其中 $f(x)=x^2$ ，而可行解为 $D=\{1,2,\cdots,31\}$ 。

第一步：根据求解精度，选5位二进制来表示可行解。如 $x=12\rightarrow 01100,\ x=31\rightarrow 11111$ 。

第二步：遗传算法中“代”的概念是通过种群来表示的。若干个

个体组成一个种群，遗传算法的操作是从某个种群开始的，而该种群就称为初始种群。初始种群通常是随机产生。对本例，假设随机产生4个个体：01101,11000,01000,10011组成初始种群。

第三步：选择和复制：首先要确定出一个适应度函数，用于判断个体对环境的适应能力。适应度函数一般由目标函数构造而成，本例直接以目标函数$f(x)=x^2$作为适应度函数。对种群中的个体进行逐个解码并根据适应度函数计算出它们的适应值，以此为根据确定出各个个体被复制的概率。如果优化问题是使得适应值最大化，并且各个染色体的适应值均为正，那么

就采用轮盘赌的方式进行选择。基本原理是：根据每个染色体适应值在整个种群中所占的比例来画出一个圆盘，然后转动圆盘若干次来选择个体。实际计算时，可以按下式计算各个个体的期望复制数，然后通过取整得到实际复制数：

$$N_i = N \frac{f_i}{\sum_i f_i}$$

其中，N是种群规模（即种群的个体个数），f_i 是第i个个体的适应值，$\sum_i f_i$ 是当前种群所有个体适应值之和，N_i是第i个个体的复制数。新一代种群产生过程见表5.1

序号	当前种群	x值	$f(x)=x^2$	选择概率 $f_i/\sum_i f_i$	期望复制数 $Nf_i/\sum_i f_i$	实际复制数	产生的新种群
1	01101	13	169	14.4%	0.576	1	01101 11000 11000 10011
2	11000	24	576	49.2%	1.968	2	
3	01000	8	64	5.5%	0.22	0	
4	10011	19	361	30.9%	1.236	1	
总计			1170	100	4.0	4	
平均值			293	25	1.0	1	
最大值			576	49.2	1.968	2	

第四步：交叉：交叉分两步进行，首先对经过复制生成的新的种群中的部分或全部染色体随机地进行两两配对；然后对配对的染色体进行部分的基因位的交叉操作，即首先选择一个交叉点，并把交叉点以后的部分互换而产生两个子代染色体，这种交叉方式称为单点交叉。

交叉的作用是实现优势互补，通过交叉，种群的总体适应值和平均适应性能得到改善，当然，不排除个体性能变差的情况。

表5.2 交叉操作

序号	当前种群	配对情况	交叉点	新种群	x值	$f(x)=x^2$
1	01101	1与2	4	01100	12	144
2	11000			11001	25	625
3	11000	3与4	2	11011	27	729
4	10011			10000	16	256
总值						1754
平均值						439
最大值						729

第五步：变异：变异是随机地将染色体的某些基因的状态变成其它状态而产生新的染色体的过程。对于二进制位串，变异就是对随机选取的位进行1→0或0→1的操作。变异的概率通常很小，其经典取值为0.001~0.01。对于本例，种群共有4×5＝20位，如果变异的概率是0.001，那么变异的位数是20×0.01＝0.2位，所有的基因均不参与变异。而如果变异的概率是0.05，则变异的概率是20×0.05＝1位，变异位随机选取。如果变异为刚好是新种群的第3位，则个体11011　111-11，则通过变异达到最优点（最大适应值）。

5.3 遗传算法的实现

根据前面的介绍，实现遗传算法一步包括如下的几个步骤：

（1）问题的优化表述

①根据具体问题确定待寻优参数和适应值函数；

②根据参数的变化范围，把参数编码成串；

③把各参数的位串串连起来形成一个染色体。

（2）初始种群的生成原则

（3）遗传算法的各种操作，遗传算法的基本算子是选择、复制和交叉。

1. 编码

（1）二进制编码

二进制编码是遗传算法中最常用的一种编码方法，二进制编码符号串的长度与问题所要求的求解精度有关。

假设某一个参数的取值范围是$[x_{min},x_{max}]$，采用长度为L的二进制编码符号串来表述该参数，则它总共能产生2^L种不同的编码。若产生编码的对应关系为

$$00\cdots000 = 0 \to x_{\min}$$
$$00\cdots001 = 1 \to x_{\min} + \delta$$
$$00\cdots010 = 2 \to x_{\min} + 2\delta$$
$$\vdots \qquad\qquad \vdots$$
$$11\cdots110 = 2^L - 2 \to x_{\max} - \delta$$
$$11\cdots111 = 2^L - 1 \to x_{\max}$$

其中

$$\delta = \frac{x_{\max} - x_{\min}}{2^L - 1}$$

是二进制表示法的精度。如果个体**x**的二进制编码是$b_L b_{L-1}\cdots b_2 b_1$，则其解码公式为

$$x = x_{\min} + \frac{x_{\max} - x_{\min}}{2^L - 1} \cdot (\sum_{i=1}^{L} b_i \times 2^{i-1}) = x_{\min} + \delta \cdot (\sum_{i=1}^{L} b_i \times 2^{i-1})$$

例如，对 $x \in [1, 1024]$，若用10位二进制编码来表述该产生的话，则有

$$\delta = \frac{1024-1}{2^{10}-1} = 1$$

于是

$$0000000000 \to 1$$

$$0000000010 \to 2$$

$$\vdots$$

$$0010101111 \to 1 + \frac{1024-1}{2^{10}-1}(2^0 + 2^1 + 2^2 + 2^3 + 2^5 + 2^7) = 176$$

$$\vdots$$

如果用11位的二进制来表述，则表述精度为

$$\delta = \frac{1024-1}{2^{11}-1} = 0.4998$$

这时，有

$$00000000000 \to 1$$

$$00000000010 \to 1 + 0.4998 = 1.4998$$

$$\vdots$$

$$00010101111 \to 1 + \frac{1024-1}{2^{11}-1}(2^0 + 2^1 + 2^2 + 2^3 + 2^5 + 2^7) = 88.4573$$

$$\vdots$$

二进制编码的有点：

①编码、解码等操作简单易行；

②交叉、变异等操作便于实现；

③符合最小字符集编码原则；

④便于用模式定理对算法进行分析。

（2）格雷码（Gray code）

二进制编码有如下的缺点：

①不便于反映所求问题的结构特征；

②局部搜索能力差。

格雷码的特点是:其连续的两个整数所对应的编码值之间只有一个码值不同；任意两个整数的差是这两个整数所对应的格雷码之间的海明距离（Hamming distance）。例如，十进制数0~15的二进制码和格雷码如表5.3所示。

表5.3 二进制码和格雷码

十进制码	二进制码	格雷码
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

二进制码和格雷码两者间的转换公式

假设有一个二进制编码为$B = b_L b_{L-1} \cdots b_2 b_1$，其对应的格雷码为 $G = g_L g_{L-1} \cdots g_2 g_1$。由二进制到格雷码的转换为公式为

$$\begin{cases} g_L = b_L \\ g_i = b_{i+1} \oplus b_i \qquad i = L-1, L-2, \cdots, 2, 1 \end{cases}$$

由格雷码到二进制的转换公式为

$$\begin{cases} b_L = g_L \\ b_i = g_{i+1} \oplus g_i \qquad i = L-1, L-2, \cdots, 2, 1 \end{cases}$$

上面两个公式中的 $\oplus$ 表示异或运算。例如，由二进制数x=1011向格雷码转换，计算如下：

$$g_4 = b_4 = 1$$
$$g_3 = b_4 \oplus b_3 = 1 \oplus 0 = 1$$
$$g_2 = b_3 \oplus b_2 = 0 \oplus 1 = 1$$
$$g_1 = b_2 \oplus b_1 = 1 \oplus 1 = 0$$

遗传算法的局部搜索能力不强，由于格雷码编码串之间只有一位之差，因而对应的参数值也只有微小的差别，这样就相当于增强了局部搜索能力。

例如，对于区间$x \in [0, 1023]$中的两个临近的整数x_1＝175和x_2＝176，若使用长度为10的二进制码，它们分别表示为

x1：0010101111；x2：0010110000

而使用同样长度的格雷码，它们分别表示为

x_1：0011111000；x_2：0011101000

格雷码的主要有点是：

①便于提高遗传算法的局部搜索能力；

②交叉、变异等遗传操作便于实现；

③符合最小字符集编码原则；

3)浮点数编码方法

浮点数编码是指个体的每个基因值用某一范围的一个浮点数来表示，其个体的编码长度取决于决策变量的个数。浮点数编码方法也称为十进制数编码方法。

例如，某一优化问题含有5个变量：x_i（i=1,2,3,4,5），每个变量$x_i \in [x_{min}, x_{max}]$,则浮点数表示如下表

x_1	x_2	x_3	x_4	x_5
1.80	6.70	3.60	2.80	9.00

表示一个体的基因型，其对应的表现型是

$$x=[1.80 \ \ 6.70 \ \ 3.60 \ \ 2.80 \ \ 9.00]^T$$

浮点数编码方法的优点：

①适合于在遗传算法中表示范围较大的数；

②适合于精度要求较高的遗传算法；

③便于遗传算法与经典优化问题的混合使用；

2. 遗传算法的模式理论

一个染色体（位串）由若干个基因（位）组成，每个基因（位）的状态是每个集合A的某一个元素。如对于二进制，则A＝{0,1}。如果一个位串的某些位固定而其它位可以取集合A中的任何元素，则该位串就构成了一个模式（Schema）用符号“*”表示可取A中的任意元素的统配符，并将状态集扩充为A’={0,1,*}.

例如，模式{01**}包括位串｛0100}, {0101}, {0110}, 和{0111}.

与位串{0000}相关的模式有多个，如{0***}, {*0**}, {*00*}和{*0*0}等。

模式S的阶次：指模式中确定位的个数，也就是非通配符的个数，模式S的阶次用O(s)表示。

例如，对模式S_1=***1**0**,S_2=0001***11,则$O(S_1)=2, O(S_2)=6$.

模式S的定义长度：是指模式中第一个和最后一个确定位（非通配符）之间的距离，用 $\delta(S)$ 表示。例如，$\delta(S_1)=7-4=3$，$\delta(S_2)=9-1=8$。而对于模式S_3=***0*****,$\delta(S_3)=4-4=0$ 。

各算子对模式的影响

1) 复制对模式的影响

设时刻k的种群P(k)中特定的模式S的个数是m=m(S,k)，则有如下复制公式：

$$\frac{m(S,k+1)}{m(S,k)}=\frac{f(S)}{\bar{f}} \qquad (5.4)$$

其中$m(S,k+1)$是在k+1时刻经过复制后种群S的数量。$f(S)$是k时刻对应于模式S的个体的平均适应值，而$\bar{f}$是种群的平均适应值。

上式表明，在选择复制后，特定模式数量的变化正比于它们的平均适应值与种群的平均适应值的比值，如果模式的适应值大于种群的适应值，那么该模式在下一代的数量就将增加，反之则减少。

例：在前面的例子中，记S_1=01**,S_2=1***,则具体的有

S_1={01101, 01000}; S_2={11000, 10011}

m(S_1,0)=2; m(S_2,0)=2

由表5.1知，$f(S_1)=(169+64)/2=116.5<\bar{f}=293$

故经过复制后在1时刻，模式S1的数量将减少，这由表5.1反应出。由公式（5.4），

$$m(S_1,1)=\frac{f(S)}{\bar{f}}m(S_1,0)=\frac{116.5}{293}\times 2=0.7952\approx 1$$

这与表5.1的结果相符合。

同理：对模式S_2，由于

$f(S_2)=(576+361)/2=468.5>\bar{f}=293$

故经过复制后在1时刻，模式S1的数量将增加，这由表5.1反应出。由公式（5.4）

$$m(S_2,1)=\frac{f(S)}{\bar{f}}m(S_2,0)=\frac{468.5}{293}\times 2=3.19\approx 3$$

同样与表5.1的结果相符合。

不是一般性，对模式平均适应值于高于种群平均值的情况，有

$$\frac{m(S,k+1)}{m(S,k)}=\frac{f(S)}{\bar{f}}=1+a,\quad a>0$$

既有

$$m(S,k+1)=(1+a)m(S,k)$$

假设a是常数，则模式S在k时刻的数量可以表示为

$$m(S,k)=(1+a)^k m(S,0)$$

所以可以认为，高于平均适应值的模式将以指数方式进行增长。

值得注意的是，选择复制操作只是保留高适应值模式，丢弃低适应值模式，而不产生新的模式。

2) 交叉对模式的影响

考虑一个长度为8的位串C=10110100及其所包含的两个模式：S_1=1*****0*和S_2=****01**。假设位串C与其它个体发生交叉的交叉点为4，用分隔符“|”表示。

C=1011|0100; S_1=1***|**0*; S_2=****|01**

一般情况下，交叉模式将破坏模式S1而保留模式S2。原因在于，对于模式S1，它的第一位状态“1”在交叉后还会保留在其后代S_1‘中，而第7

位状态“0”进而交叉对象。除非交叉对象第7位的状态与S_1相同（属于小概率事件），否则，模式S_1与其后代S_1'将不一样。

例：设D=0111|1111，C和D交叉后所产生的后代是：

C'=1011|1111和D'=0111|0100

显然，交叉后所产生的后染色体C'和D'均不属于模式S_1'，但D'属于模式S_2'，即模式S_1被破坏而模式S_2被保留。

所以，通过位于染色体两端的位置所表示的模式更容易因交叉而被破坏。

3 适应度函数F的确定

遗传算法通常是考虑如下的优化问题

$$\min_{x\in D} J(x) \qquad \text{或者} \qquad \max_{x\in D} J(x)$$

其中J(x)称为目标函数。

适应度函数F是遗传算法与实际优化问题的接口。在遗传算法中，通常要求：

1. 适应度函数是非负的;

2. 且认为一个个体的F值越大，则该个体适应环境的程度就越好。

F通常是基于J构造而出，F的构造方法具体有：

①如果是最大化优化问题，且有 $J(x) \geq 0,\ \forall x \in D$，则可以取

F=J

②将适应度函数表示为目标函数J的线性形式，

既有

$$F(x)=a\times J(x)+b$$

其中a和b是常系数，可以根据具体问题的特征来确定。

③对于最小化问题，可采用如下转换形式

$$F(x)=\begin{cases} c_{\max}-J(x) & \text{当} J(x)<c_{\max} \\ 0 & \text{其它} \end{cases}$$

其中，c_{max}或为理论上的最大值，或是当前进化代及其所有父代中的最大值（此时c_{max} 随着进化代的变化而变化）。

④对于最大化问题，可采用如下的转换

$$F(x)=\begin{cases} J(x)-c_{\min} & \text{当} J(x)>c_{\min} \\ 0 & \text{其它} \end{cases}$$

其中，c_{min}或为理论上的最小值，或是当前进化代及其所有父代中的最小值（此时c_{min} 随着进化代的变化而变化）。

⑤采用如下的指数函数形式

$$F(x)=c^{J(x)}$$

对于最大化问题，c一般取1.618或2；对于最小化问题，c可取0.618，这样可以保证适应值最大方向和目标函数优化方向一致。

适应度函数的变换

在GA初期，各个个体的性态明显不同，其适

大小差别很大，个别优良的个体的适应度有可能远远高于其它个体，从而增加了被复制的次数，并在下一代群体中占有很高的比例；而个别适应度低的个体，尽管本身含有部分有益的基因（字符），却会过早地被抛弃，结果会使种群的多样性降低，导致GA的早熟现象，即过早地收敛到局部最优解。

另一方面，在GA后期，种群中所有个体的平均适应度可能接近于种群中最佳个体的适应度，这时各个个体的适应度差别不大，都会以1概率的可能被复制到下一代，从而使进化无竞争可言。

为了克服以上的不足，需对适应度函数进行缩放变换。

目前常用的适应度尺度变换的主要方法有三种：线性尺度变换、乘幂变换和直属尺度变换。

（1）线性尺度变换：线性尺度变换公式如下

$$F'=aF+b$$

其中，F为原适应度，F’为变换后适应度，a和b为常系数。

一般期望该变换满足如下的两个条件：

条件一：缩放变换后全部个体的新适应度的平均值F'_{avg}要等于原适应度的平均值F’，即

$$F'_{avg}=F_{avg}$$

条件二：缩放后种群中新的最大适应度F'_{max}要等于原平均适应度F_{avg}的指定倍数，即

$$F'_{avg}=CF_{avg}$$

其中，C为最佳个体的期望复制数量。

条件三：F'为非负。

为达到以上三个条件，可按如下方式却定a和b。

①若

$$F_{\min}=\frac{CF_{avg}-F_{\max}}{C-1}$$

则转②，否则转③

②

$$a=\frac{C-1}{F_{\max}-F_{avg}}F_{avg}$$

$$b=\frac{(F_{\max}-CF_{avg})F_{avg}}{F_{\max}-F_{avg}}$$

③

$$a = \frac{F_{avg}}{F_{avg} - F_{\min}}$$

$$b = -\frac{F_{\min} F_{avg}}{F_{avg} - F_{\min}}$$

（2）乘幂缩放：

$$F' = F^{k}$$

（3）指数缩放：

$$F' = e^{-\beta F}$$

4 复制和选择

在遗传算法的搜索过程中，对生存环境适应度高的物种将有更多的机会遗传到下一代；而对生

存环境适应度较低的物种遗传到下一代的机会相对较小。遗传算法是用选择和复制算子来模拟这个过程的。

（1）比例选择法

其基本思想是依据个体的适应度占总适应度的比例来确定被复制的可能性的，这在前面已经有详细的介绍。

（2）随机联赛选择法

设种群的规模是M，该方法的基本操作是：

①从第t代（当前代）中随机的选取k个个体，其中1<k<M.

②比较k个个体的适应度，选择适应度最大

者进行复制，进入第t+1代，同时被复制的个体仍保留在第t代。

③ 重复以上的①和②共M次，得到t+1代的含有M个个体的新种群。

此外还有：分级选择法，玻尔兹曼选择法等。但最常用的还是比例选择法，它符合遗传算法的模式理论。

5 交叉运算

交叉运算是遗传算法产生新个体的主要手段。所谓交叉运算，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，从而形成两个新的染色体(个体)。

交叉运算的操作过程是先对种群中的个体随机地进行两两配对，即将种群中的M个个体随机的方式组成[M/2]对，然后每对均按某种方式进行交叉。主要交叉方法有：

（1）单点交叉：是指在个体编码串中随机地设计一个交叉点，然后进行部分基因交叉。如

$$A:\ xxxxxxx|xxx \qquad A':\ xxxxxxx|yyy$$

单点交叉

$$B:\ yyyyyyy|yyy \quad \rightarrow \quad B':\ yyyyyyy|xxx$$

（2）双点交叉与多点交叉：是指在个体编码中随机设计两个或两个以上的交叉点，然后进行部分基因交换。如

$$A:\ xx\,|\,xxxxx\,|\,xxx \qquad A':\ xx\,|\,yyyyy\,|\,xxx$$

双点交叉

$$B:\ yy\,|\,yyyyy\,|\,yyy \quad \rightarrow \quad B':\ yy\,|\,xxxxx\,|\,yyy$$

（3）均匀交叉：是指两个配对个体的每个基因座的基因都以相同的交叉概率进行交换，从而形成两个新的个体。主要操作过程如下：

①随机长生一个与个体编码串等长的屏蔽字

$$W = w_1 w_2 \cdots w_i \cdots w_L$$

其中L是个体的长度。

②若i=0，则第i位不进行交叉；若i=1，则第i位进行交叉，如：

$$\begin{array}{lcl} A: xxxxxxxxxx & \overset{\text{均匀交叉}}{\longrightarrow} & A': yxyxyxyxyx \\ B: yyyyyyyyyy & w = 1010101010 & B': xyxyxyxyxy \end{array}$$

6 变异

遗传算法中的变异运算，是指将染色体中的某些基因位的状态变成其它状态而产生新的染色体。对于二进制，变异就是将某些基因从0变成1，或者从1变成0。

7 举例

考虑最大化优化问题

$$\max f(x_1,x_2)=21.5+x_1\sin(4\pi x_1)+x2\sin(20\pi x_2)$$

$$-3.0\le x_1 \le 12.1$$

$$-4.1 \le x_2 \le 5.8$$

第一步，确定决策变量和约束条件

约束条件如上，决策变量是x_1和x_2

第二步，建立优化表达式

如题目所示

第三步，确定编码方法

采用二进制，设对变量x1和x2，要求精精确到小数点后4位，则按如下确定每一个变量的对于的二进制的长度：

$$\delta_1 = \frac{12.1-(-3.0)}{2^{L_1}-1} \le 10^{-4} \Rightarrow L_1 = 18$$

$$\delta_2 = \frac{5.8-(-4.1)}{2^{L_2}-1} \le 10^{-4} \Rightarrow L_2 = 15$$

这样，一个染色体的长度=$L_1+L_2=18+15=33$

变量	二进制	十进制	编码所表示的数
x_1	000001010100101001	5417	-2.687969
x_2	101111011111110	24318	5.361653

其中，编码所表示的数计算如下：

$$x_1 = -3.0 + \frac{12.1-(-3.0)}{2^{18}-1} \times 5417 = -2.687969$$

$$x_2 = 4.1 + \frac{5.8-(-4.1)}{2^{15}-1} \times 24318 = 5.361653$$

于是编码

000001010100101001 101111011111110表示二维向量

$$x = \begin{pmatrix} -2.687969 \\ 5.361653 \end{pmatrix}$$

从$2^{33}=8.5899\times10^9$个染色体中随机地选取10生成初始种群如下：

U1=[00000101010010101001101111101111111110]

U2=[00111010111001100000001010100100 1000]

U3=[11100011100000100001010100100 0110]

U4=[10011011010010110100000010111001]

U5=[00001011110110001000111000110 1000]

U6=[11111010101101100000001011001 1001]

U7=[11010001001111110001001100111 01101]

U8=[00101101010000110001011001100 1100]

U9=[11111000101111011000111010001 11101]

U10=[11110100111010101000001010110 1010]

该种群中的10个染色体解码后的实际值为

U1=[-2.687969, 5.361653]

U2=[0.474101, 4.170144]

U3=[10.419457, 4.66461]

U4=[6.159951, 4.109598]

U5=[-2.301286, 4.477282]

U6=[11.788084, 4.174346]

U7=[9.342067, 5.121702]

U8=[-0.330256, 4.694977]

U9=[11.671267, 4.873501]

U10=[11.446273, 4.171908]

第四步，确定个体的评价方案（确定适应度函数）

由于该例是最大化优化问题，且目标函数在定义域内是非负的，所以可以取目标函数为适应度函数，即取

$$F=f$$

计算初始种群各染色体的适应度如下：

F(U1)=f(-2.687969, 5.361653)=19.805119

F(U2)=f(0.474101, 4.170144)=17.370896

F(U3)=f(10.419457, 4.66461)=9.590546

F(U4)=f(6.159951, 4.109598)=29.406122

F(U5)=f(-2.301286, 4.477282)=15.686091

F(U6)=f(11.788084, 4.174346)=11.900541

F(U7)=f(9.342067, 5.121702)=17.958717

F(U8)=f(0.330256, 4.694977)=19.763190

F(U9)=f(11.671267, 4.873669)=26.401669

F(U10)=f(11.466273, 4.171908)=10.252480

从以上的数据可以看出，该初始种群中，染色体U4最健壮，U3最虚弱。

第五步，设计遗传算子

用轮盘赌选择法进行复制：

①计算各染色体U_i的适应度值$F(U_i)$

②计算种群的适应度总和

$$sumF = \sum_{i=1}^{M} F(U_i)$$

③确定对于每一个染色体Ui的选择概率

$$P_i = \frac{F(U_i)}{sumF}$$

④计算每个染色体U_i的累计概率

$$Q_i = \sum_{j=1}^{i} P_j, \quad j = 1,2,\cdots,M$$

在实际操作中，选择新种群的一个染色体按如下的步骤进行：

①生成一个[0, 1]间的随机数r；

②如果$r \geq Q_1$，就选择染色体U_1，否则选择第i个染色体U_i使得

$$Q_{i-1} \leq r \leq Q_i$$

对本例，种群总的适应度为

$$sumF = \sum_{i=1}^{M} F(U_i) = 178.135372$$

对应于每个染色体U_i的选择概率P_i如下

P_1=0.111180, P_2=0.097571, P_3=0.053839,

P_4=0.165077, P_5=0.088057, P_6=0.066806,

P_7=0.100815, P_8=0.110945, P_9=0.148211,

P_{10}=0.057554

对应于每个染色体U_i的选择概率Q_i如下

Q_1=0.111180, Q_2=0.208695, Q_3=0.262534

Q_4=0.427611, Q_5=0.515668, Q_6=0.582475

Q_7=0.683290, Q_8=0.794234, Q_9=0.942446

Q_{10}=1.000000

现在，我们轮盘10次。假设这10次中生成的[0,1]间的随机数如下：

0.301431, 0.322062, 0.766503, 0.811893, 0.350871, 0.583392, 0.177618, 0.343242, 0.032685, 0.197577.

由于$Q_3 \le r_1 \le Q_4$，故第一次选择U_4；又由于$Q_3 \le r_2 \le Q_4$，故U_4第二次再次被选中；用同样的方法，得到新种群如下：

U1=[10011011010010110100000010111001](U4)

U2=[10011011010010110100000010111001](U4)

U3=[00101101010000110001011001100](U8)

U4=[11111000101110110001110100111101](U9)

U5=[10011011010010110100000010111001](U4)

U6=[11010001001111100010011001111101](U7)

U7=[00111010111001100000001010100100](U2)

U8=[10011011010010110100000010111001](U4)

U9=[00000101010010100110111101111110](U1)

U10=[00111010111001100000001010100100](U2)

交叉运算使用单点交叉算子：假设交叉概率为P_C=25%，即在平均水平上有25%的染色体进行

交叉，交叉过程如下

开始

k:=0

当k≤10时继续

r_k∈[0, 1]之间的随机数；

如果r_k<0.25，则

选择U_k为交叉的一个父代；

结束

k:=k+1

结束

结束

假设随机数如下

0.625721, 0.266823, 0.288644, 0.295114,
0.163272, 0.567461, 0.085940, 0.392865,
0.770714, 0.548656

那么就意味着染色体U_5和U_7被选中作为交叉的父代。再随机地选择[1, 33]中的一个整数作为交叉点。假设选中的交叉点为1，则交叉运算如下：

U_5=[1001101101001011010000010111001]

U_7=[0011101011100110000001010100 1000]

U^*_5=[1011101011100↓1000000010101001000]

U^*_7=[0001101101001011010000010111001]

变异运算使用基本位变异算子

假设基因变异的概率是P_m=0.01，对本例，共有33×10=330个基因，则每代变量的基因数是330×0.01＝3.3个。假设变异的基因数是4。生成[0, 1]间的随机数330个，则取随机数中最大的前4位随机数所对于的位进行变异。假设r_{105}, r_{164}, r_{199}, r_{329}是330个随机数中较大的前4位，则基因变异发生在105，164，199和321位，于是得到下表

基因位置	染色体位置	基因数
105	4	6
164	5	32
199	7	1
329	10	32

↓

根据上表，进行如下的变异操作

U_4=[11111000101111011000111010001111101]

↓

U^*_4=[11111100101111011000111010001111101]

U^*_5=[10111010111001100000001010100100 0]

↓

U^{**}_5=[10111010111001100000001010100101 0]

U^*_7=[00011011010010110100000010111001]

↓

U^{**}_7=[10011011010010110100000010111001]

$$U_{10}=[00111010111001100000001010100100]$$

$$\downarrow$$

$$U^{*}_{10}=[00111010111001100000001010100101]$$