《遗传算法原理及其应用》

1207

Chap1 序论

一. 遗传算法的生物学基础

1.1 遗传与变异

基本概念

Cell:细胞
Chromosome:染色体
Gene:基因
Locus:基因座
Allele:等位基因
Genotype:基因型
Phenotype:表现型
Genome:基因组
Reproduction:复制
Crossover:交叉
Mutation:变异

1.2 进化

基本术语

Evolution:进化
Population:群体
Individual:个体
Fitness:适应度

1.3 遗传与进化的系统观

特点:

1) 生物的所有遗传信息都包含在其染色体中,染色体决定了生物的性状;
2) 染色体是基因及其有规律的排列所构成,遗传和进化过程发生在染色体上;
3) 生物的繁殖过程是由其基因的复制过程完成的;
4) 通过同源染色体之间的交叉或染色体的变异会产生新的物种,使生物呈现新的性状;
5) 对环境适应性好的基因或者染色体会经常比适应性差的基因或染色体有更多的机会遗传到下一代。

二. 遗传算法简介

遗传算法是模拟生物在自然环境中的遗传和进化过程而形成的一种自适应全局优化概率

搜索算法。

2.1 遗传算法概要

对于一个求函数最大值的优化问题（求函数最小值也类同），一般可描述为下述数学规划模型：

$$\begin{cases} \max & f(X) \\ s.t. & X \in R \\ & R \subseteq U \end{cases}$$

式中，$X=[x_1,x_2,...,x_n]^T$ 为决策变量，f（X）为**目标函数**，第 2，3 式为**约束条件**，U 是**基本空间**，R 是 U 的一个子集。满足约束条件的解 X 称为**可行解**，集合 R 表示由所有满足约束条件的解所组成的一个集合，叫做**可行解集合**。

对上述最优化问题，目标函数和约束条件种类繁多，由的是线性的，有的是非线性的；有的是连续的，有的是离散的；有点是单峰的，有的是多峰的。

求最优解或近似最优解的方法主要有三种：枚举法，启发式算法和搜索算法：

1） **枚举法：**枚举出可行解集合内的所有的可行解，以求出精确最优解。对于连续函数，首先要求对其进行离散化处理。该法效率较低。

2） **启发式算法：**寻求一种能产生可行解的启发式规则，以找到一个最优解或近似最优解。此法对每个问题都必须找出其特有的启发式规则，不具有通用性。

3） **搜索算法：**寻求一种搜索算法，该算法在可行解集合的一个子集内进行搜索操作，以找到问题的最优或者近似最优解。若适当利用一些启发式知识，可以在近似解的质量和求解效率上达到一种较好的平衡。

遗传算法中，将 n 维决策向量 $X=[x_1,x_2,...,x_n]^T$ 用 n 个记号 X_i (i=1,2,…,n) 所组成的符号串 X 表示：

$$X=X_1X_2...X_n \Rightarrow X=[x_1,x_2,...,x_n]^T \quad X=X_1X_2...X_n \Rightarrow X=[x_1,x_2,...,x_n]^T ,$$

把每一个 X_i 看作一个**遗传基因**，它的所有的可能取值称为**等位基因**。这样,X 就可看做是由 n 个遗传基因组成的一个**染色体**。一般情况下，染色体的长度 n 是固定的，但对一些问题你也可以是变化的。根据不同的情况，这里的等位基因可以是一组整数，也可以是某一范围内的实数值，或者纯粹的一个记号。最简单的等位基因是由 0 和 1 这两个整数组成的，相应的染色体就可以表示为一个二进制符号串。这种编码形成的排列形式 X 就是个体的**基因型**，与之对应的 X 值就是个体的**表现型**。个体的适应度与其对应的个体表现型 X 的目标函数值相关联，X 越接近于目标函数的最优点，其适应度越大；反之，其适应度越小。

遗传算法中，决策变量 X 组成了问题的解空间。对问题最优解的搜索是通过对染色体 X 的搜索过程来进行的，从而由所有的染色体 X 就组成了问题的搜索空间。遗传算法的运算过程也是一个反复迭代过程，第 t 代群体记做 P(t), 经过一代遗传和进化后，得到第 t＋1 代群体，它们也是由多个个体组成的集合，记做 P(t＋1)。

12．08

遗传算法中最优解的搜索过程也是模仿生物的进化过程,通过染色体之间的交叉和染色体的变异来完成。通过所谓的遗传算子（genetic operators）作用于群体 P（t）中，进行遗传

操作，从而得到新一代群体 P(t＋1)。

A. **选择（selection）：** 根据各个个体的适应度，按照一定的规则或方法，从第 t 代群体 P(t)中选择出一些优良的个体遗传到下一代群体 P(t＋1)中。

B. **交叉（crossover）：**将群体 P（t）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，crossover rate）交换他们之间的染色体。

C. **变异（mutation）：**对群体 P(t)中的每一个个体，以某一概率（称为变异概率，mutation rate）改变某一个或某一些基因座上的基因值为其他的等位基因。

图 1－2 所示为遗传算法的运算过程示意图：

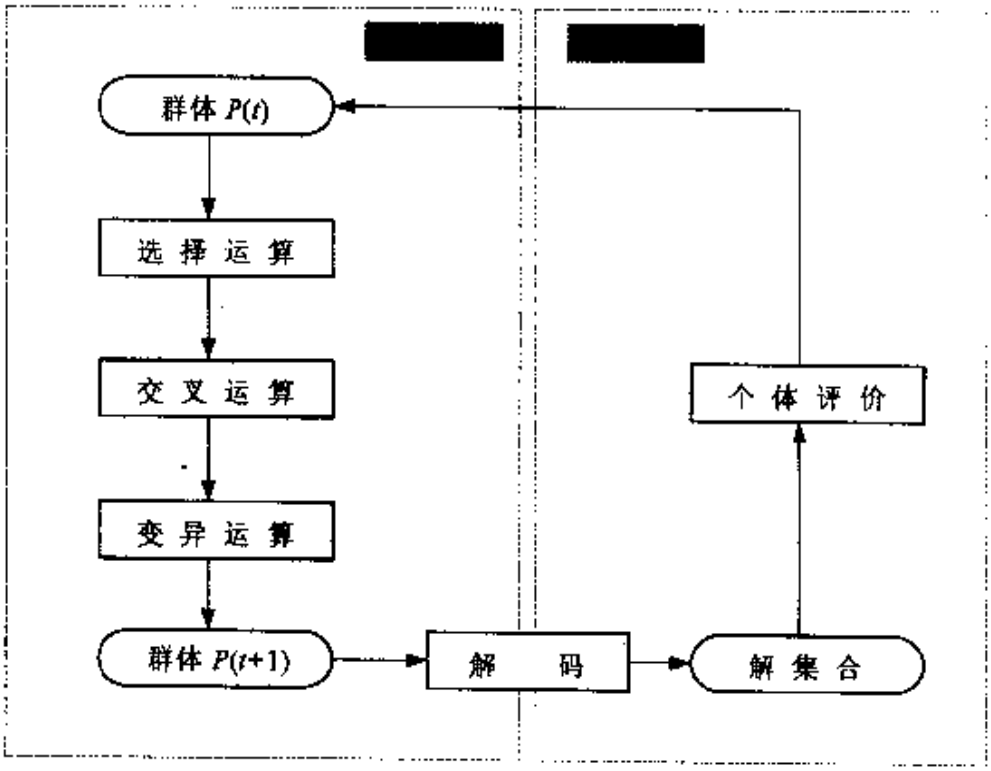


图 1-2 遗传算法的运算过程示意

由该图可以看出，使用上述三种遗传算子（选择算子，交叉算子，变异算子）的遗传算法的主要运算过程如下所述：

步骤一：初始化。 设置进化代数计数器 t<－0；设置最大进化代数 T；随机生成 M 个个体作为初始群体 P(0)。

步骤二：个体评价。计算群体 P(t)中各个个体的适应度。

步骤三：选择运算。将选择算子作用于群体。

步骤四：交叉运算。将交叉算子作用于群体。

步骤五：变异运算。将变异算子作用于群体。群体 P(t)经过选择，交叉，变异运算之后得到下一代群体 P(t＋1)。

步骤六：终止条件判断。若 t<=T，则：t<－t＋1，转到步骤二；若 t>T，则以进化过程中得到的具有最大适应度的个体作为最优解输出，终止计算。

2．2 遗传算法的手工模拟计算示例

求下述二元函数的最大值：

$$\begin{cases} \max \quad f(x_1, x_2) = x_1^2 + x_2^2 \\ \text{s. t.} \quad x_1 \in \{0, 1, 2, \cdots, 7\} \\ \qquad\quad x_2 \in \{0, 1, 2, \cdots, 7\} \end{cases}$$

现对其主要运算过程作如下解释：

1） 个体编码。 遗传算法的运算对象是表示个体的符号串，所以必须把变量 x_1，x_2 编码为一种符号串。该例中，x_1，x_2 取 0～7 之间的整数，可分别用 3 位无符号二进制整数来表示，将它们连接在一起所组成的 6 位无符号二进制整数就形成了个体的基因型，表示一个可行解。例如，基因型 X=101110 所对应的表现型是：$X=[5,6]^T$。个体的表现型 x 和基因型 X 之间可通过编码和解码程序相互转换。

2）初始群体的产生。 遗传算法是对群体进行的进化操作，需要给其准备一些表示其实搜索点的初始群体数据。本例中，群体规模的大小取为 4，即群体由 4 个个体组成，每个个体可通过随机方法产生。一个随机产生的初始群体如表 1－1 中第②栏所示。

3）适应度计算。 遗传算法中以个体适应度的大小来评定各个个体的优劣程度，从而决定其遗传机会的大小。结合本例情况，可直接用目标函数值作为个体的适应度。为计算函数的目标值，需对个体基因型 X 进行解码。表 1－2 中第③，④栏所示为初始群体中各个个体的解码结果，第⑤栏所示为各个个体所对应的目标函数值，它也示个体的适应度，第⑤栏还给出了群体中适应度的最大值和平均值。

4）选择运算。 选择运算（或称之为复制运算）把当前群体中适应度较高的个体按某种规则或模型遗传到下一代群体中。一般要求适应度较高的个体将有更多的机会遗传到下一代群体中。本例中，采用与适应度成正比的概率来确定各个个体复制到下一代群体中的数量。其具体操作过程是：先计算出群体中所有个体的适应度的总和 $\sum f_i$；其次，计算出各个个体的相对适应度的大小 $f_i / \sum f_i$，如表中第⑥栏所示，它即为每个个体被遗传到下一代群体中的概率，每个概率值组成一个区域，全部概率值之和为 1；最后再产生一个 0 到 1 的随机数，依据该随机数落在哪个区域内就选择哪个个体。如表中第 ⑦，⑧栏所示为一个随机产生的选择结果。

5） 交叉运算。 交叉运算是遗传算法中产生新个体的主要操作过程，它以某一概率相互交换某两个个体之间的部分染色体。本例采用单点交叉的方法，其具体操作过程是：先对群体进行随机配对，如表中第⑨栏所示为一种随机配对情况；其次随机设置的交叉位置，如表中第⑩栏所示为一随机产生的交叉点位置，其中的数字表示交叉点设置在该基因座之后；最后在相互交换配对染色体之间的部分基因。表中第 ⑾栏所示为交叉运算的结果。

6） 变异运算。 变异运算是对个体的某一个或某一些基因座上的基因值按某一较小的概率进行改变，它也是产生新个体的一种操作方法。本例中，采用基本位变异的方法来进行变异运算，具体操作过程是：首先确定出各个个体的基因变异位置，表中第（12）栏所示为随机产生的变异点位置，其中的数字表示变异点在该基因座处；然后依照某一概率将变异点的原有基因值取反。表中第（13）栏所示为变异结果。

表 1-1 遗传算法的手工模拟计算

① 个体编号 i	② 初始群体 $P(0)$	③ x_1	④ x_2	⑤ $f_i(x_1, x_2)$	⑥ $f_i/\sum f_i$
1	011101	3	5	34	0.24
2	101011	5	3	34	0.24
3	011100	3	4	25	0.17
4	111001	7	1	50	0.35

$\sum f_i = 143$, $f_{max} = 50$, $\bar{f} = 35.75$

⑦ 选择次数	⑧ 选择结果	⑨ 配对情况	⑩ 交叉点位置	⑪ 交叉结果	⑫ 变异点	⑬ 变异结果
1	011101			011001	4	011101
1	111001	1-2	1-2：2	111101	5	111111
0	101011	3-4	3-4：4	101001	2	111001
2	111001			111011	6	111010

⑭ 子代群体 $P(1)$	⑮ x_1	⑯ x_2	⑰ $f_i(x_1, x_2)$	⑱ $f_i/\sum f_i$
011101	3	5	34	0.14
111111	7	7	98	0.42
111001	7	1	50	0.21
111010	7	2	53	0.23

$\sum f_i = 235$, $f_{max} = 98$, $\bar{f} = 58.75$

对群体 P(t)进行一轮选择，交叉，变异运算后可得到新一代的群体 P(t＋1)。

三．遗传算法的特点

为了解决各种优化计算问题，人们提出了各种优化算法，如单纯形法，梯度法，动态规划法，分枝定界法等。而遗传算法是一类可用于复杂系统优化计算的鲁棒搜索算法，其特点如下：

1. 遗传算法以决策变量的编码作为运算对象。
2. 遗传算法直接以目标函数值作为搜索信息。
3. 遗传算法同时使用多个搜索点的搜索信息。
4. 遗传算法使用概率搜索技术。

四．遗传算法的发展

1. J.H.Holland 遗传算法的基本定理－－模式定理（Schema Theorem）
2. J.D.Bagley 在遗传算法的不同阶段采用不同的选择率，有利于防止早熟。
3. K.A.De Jong 定义了评价遗传算法性能的在线指标和离线指标。
4. D.J.Goldberg 《搜索，优化和机器学习中的遗传算法》
5. J. R. Koza 遗传编程（Genetic Programming , 1992）

五．遗传算法的应用

1. 函数优化。
2. 组合优化。
3. 生产调度问题。
4. 自动控制。
5. 机器人学习。
6. 图像处理。
7. 人工生命。
8. 遗传编程。
9. 机器学习。

Chap2 基本遗传算法

一．基本遗传算法描述

Goldberg 总结出了一种统一的最基本的遗传算法－－基本遗传算法（Simple Genetic Algorithms，简称 **SGA**），只使用选择算子，交叉算子和变异算子这三种基本遗传算子。

1．基本遗传算法的构成要素

1） 染色体编码方法。 基本遗传算法使用固定长度的二进制符号串儿来表示群体中的个体，其等位基因是由二值符号集{0,1}所组成的。初始群体中各个个体的基因值可用均匀分布的随机数来生成。

2） 个体适应度评价。 基本遗传算法按与个体适应度成正比的概率来决定当前群体中每个个体遗传到下一代群体中的机会多少。为正确计算这个概率，这里要求所有个体的适应度必须为正或零。至于，必须预先确定好由目标函数值到个体适应度之间的转换规则，特别是要预先确定好当目标函数值为负数时的处理。

3） 遗传算子。 使用下述三种遗传算子：

A.. 选择运算使用比例选择算子；

B. 交叉运算使用单点交叉算子；

C．变异运算使用基本位变异算子或均匀变异算子。

4） 基本遗传算法的运行参数。 基本遗传算法有下述 4 个运行参数需要提前设定：

A. M：群体大小。一般取为 20～100；

B. T：遗传运算终止进化代数。一般取为 100～500；

C. P_c：交叉概率。一般取为 0.4～0.99;

D. P_m：变异概率。一般取为 0.0001~0.1。

2．基本遗传算法描述

基本遗传算法可定义为一个 8 元组：

$$SGA=(C，E，P_0，M，\Phi，\Gamma，\Psi，T)$$

式中 C－－个体的编码方式；　　Φ－－选择算子 ；

E－－个体适应度评价函数；　　Γ－－交叉算子；

P_0－－初始群体；　　Ψ－－变异算子；

M－－群体大小；　　T－－遗传运算终止条件。

用伪码表示算法如下：

```
Procedure SGA
begin
    initialize P (0);
    t = 0;
    while (t ≤ T) do
        for i = 1 to M do
            Evaluate fitness of P (t);
        end for
        for i = 1 to M do
            Select operation to P (t);
        end for
        for i = 1 to M/2 do
            Crossover operation to P (t);
        end for
        for i = 1 to M do
            Mutation operation to P (t);
        end for
        for i = 1 to M do
            P (t + 1) = P (t)
        end for
        t = t + 1;
    end while
end
```

12. 09

二．基本遗传算法的实现

1．个体适应度评价

对于求目标函数最小值的优化问题，理论上只需要简单地对其增加一个负号转化为求目标函数最大值的优化问题，即：

$$\min f(X) = \max(-f(X)) \qquad (2\text{-}2)$$

当优化目标是求函数最大值，并且目标函数总取正值时，可以直接设定个体的适应度F(X)就等于相应的目标函数值 f(X)，即：

$$F(X)=f(X) \qquad (2\text{-}3)$$

但实际问题中的目标函数值有正也有负的时候，优化目标有函数最大值，也有函数最小值，显然上面两式保证不了所有情况下个体的适应度都是非负的这个要求。为此，基本遗传算法一般采用下面的两种方法之一将目标函数值 f（X）变换为个体的适应度 F(X)。

A. 对于**求目标函数最大值**的优化问题，变换方法为：

$$F(X) = \begin{cases} f(X) + C_{min}, & \text{if } f(X) + C_{min} > 0 \\ 0, & \text{if } f(X) + C_{min} \leqslant 0 \end{cases} \qquad (2\text{-}4)$$

其中，C_{min} 为一个适当地相对较小的数，可用下面几种方法来选取：

a. 预先指定的一个较小的数；
b. 进化到当前代为止的最小的目标函数值；
c. 当前代或最近几代群体的最小的目标函数值。

B. 对于**求目标函数最小值**的优化问题，变换方法为：

$$F(X) = \begin{cases} C_{max} - f(X), & \text{if } f(X) < C_{max} \\ 0, & \text{if } f(X) \geqslant C_{max} \end{cases} \qquad (2\text{-}5)$$

式中，C_{max} 为一个适当地相对较大的数，可用下面几种方法之一来选取：

a. 预先指定的一个较大的数。
b. 进化到当前代为止的最大目标函数值。
c. 当前代或最近几代群体中的最大的目标函数值。

2．比例选择算子

比例选择算子是指个体被选中到下一代群体中的概率与该个体的适应度大小成正比，也叫赌轮（Roulette Wheel）选择。其具体执行过程是：

A. 先计算出群体中所有个体的适应度的总和；
B. 其次计算出每个个体相对适应度，它即为个体被遗传到下一代群体的概率；
C. 最后再使用模拟赌轮操作（即 0 到 1 之间的随机数）来确定各个个体被选中的次数。

3．单点交叉算子

具体执行过程：

A. 对群体中的个体进行两两随机配对。若群体大小为 M，则共有【M/2】对相互配对的个体组。

B. 对每一对相互配对的个体，随机设置某一基因座之后的位置为交叉点。若染色体的长度为 n，则共有（n－1）个可能的交叉位置。

C. 对每一对相互配对的个体，依设定的交叉概率 p_c 在交叉点处相互交换两个个体的部分染色体，从而产生出两个新的个体。

4．基本位变异算子

基本位变异算子对于基本遗传算法来说，就是对需要变异的某一基因座上的原有的基因值进行取非操作。其执行过程是：

A．对个体的每一个基因座，依变异概率 p_m 指定为变异点。

B．对每一个指定的变异点，对其基因值取非运算或者用其他等位基因值来代替，从而产生出一个新的个体。

三．基本遗传算法应用举例

1．遗传算法的应用步骤

A．确定决策变量及其各种约束条件，即确定出个体的表现型 X 和问题的解空间。

B．建立优化模型，即确定出目标函数的类型（是求目标函数的最大值还是最小值？）及其数学描述形式或量化方法。

C．确定表示可行解的染色体编码方法，也即确定出个体的基因型 X 及遗传算法的搜索空间。

D．确定解码方法，即确定出由个体基因型 X 到个体表现型 X 的对应关系或转换方法。

E．确定个体适应度的量化评价方法，即确定出由目标函数值 f（X）到个体适应度 F(X)的转换规则。

F．设计遗传算子，即确定出选择运算，交叉运算，变异运算等遗传算子的具体操作方法。

G．确定遗传算法的有关运行参数，即确定出遗传算法的 M,T,p_c, p_m 等参数。

由上述构造步骤可以看出，可行解的编码方法，遗传算子的设计是构造遗传算法时需要考虑的两个主要问题，也是设计遗传算法时的两个关键步骤。对不同的优化问题需要使用不同的编码方法和不同操作的遗传算子，它们与所求解的具体问题密切相关，因而对所求解的理解程度是遗传算法应用成功与否的关键。

下图中所示为遗传算法的主要构造过程示意图。

2．基本遗传算法在函数优化中的应用举例

例：Rosenbrock 函数的全局最大值计算。

$$\max \quad f(x_1,x_2)=100(x_1^2-x_2)^2+(1-x_1)^2 \qquad (2-6)$$

$$s.t. \quad -2.048\le x_i\le 2.048 \quad (i=1,2) \qquad (2-7)$$

该函数有两个局部极大值，分别是 f（2.048, -2.048）=3897.7342 和 f（-2.048, -2.048）=3905.9262，其中后者为全局最大点。

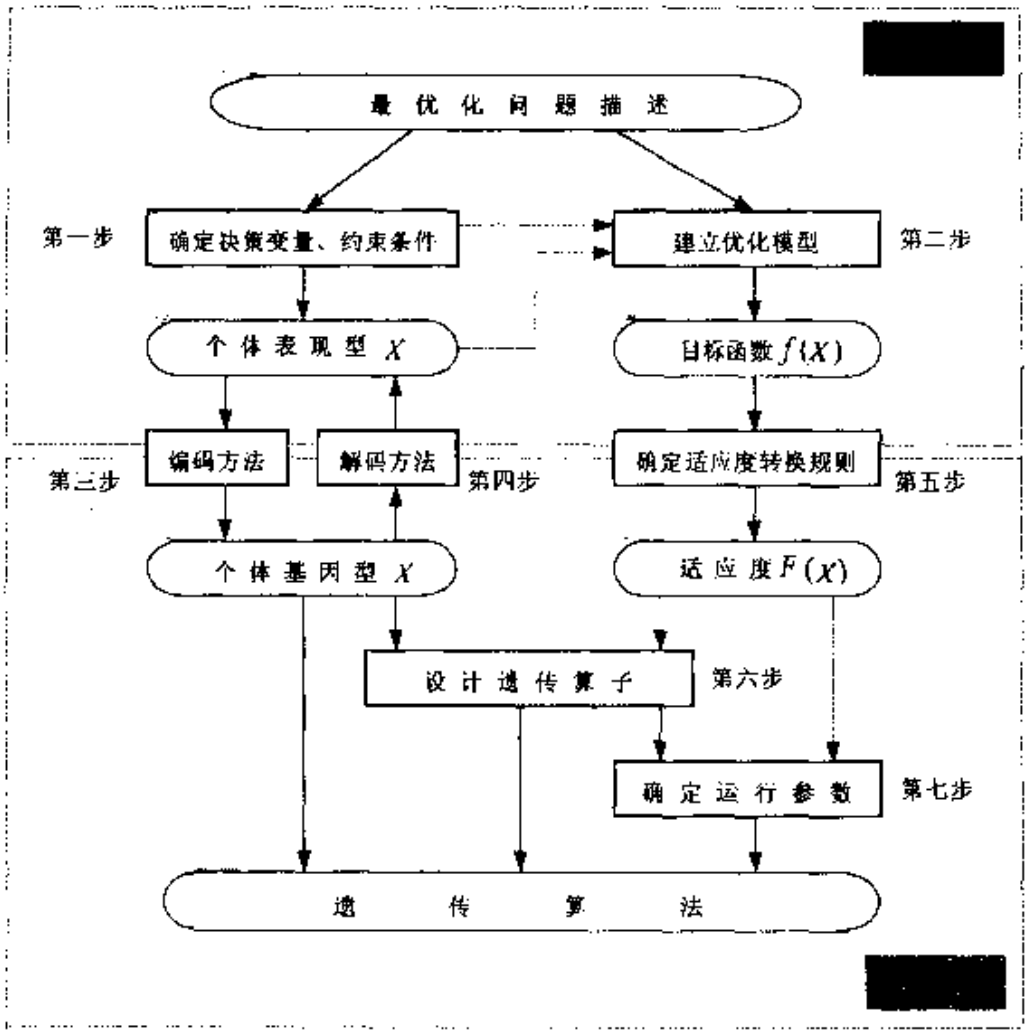


图 2-2 遗传算法的主要构造过程示意图

下面介绍求解该问题的遗传算法的构造过程。

A. 确定决策变量和约束条件。式 2－7 已经给出了该问题的决策变量和约束条件。

B. 建立优化模型。式 2－6 已经给出了该问题的数学模型。

C. 确定编码方法。用长度为 10 位的二进制编码串来分别表示两个决策变量 x_1，x_2。10 位二进制编码串可以表示从 0 到 1023 个均等的区域，包括端点在内有 1024 个不同的离散点。从离散点-2.048 到 2.048，依次让它们分别对应于 0000000000（0）到 1111111111（1023）之间的二进制编码。再将分别表示 x_1，x_2 的二个二进制编码串连接在一起，组成一个 20 位长的二进制编码串，它就构成了这个函数优化问题的染色体编码方法。使用这种编码方法，解空间和遗传算法的搜索空间具有一一对应关系。

D. 确定解码方法。 编码的逆过程。

E. 确定个体评价方法。直接用目标函数值作为个体的适应度。

F. 设计遗传算子。选择运算采用比例选择算子；交叉运算采用单点交叉算子；变异运

算采用基本位变异算子。

G. 确定遗传算法的运行参数。对于本例，设定遗传算法的运行参数如下：

群体大小：M=80

终止条件：T=200

交叉概率：p_c＝0.6

变异概率：p_m＝0.001

通过运行后，通常将其进化过程用图表示出来。其中的横坐标表示进化代数，纵坐标表示适应度，适应度通常分别作出适应度的最大值和平均值的图形。

12.10

Chap 3. 遗传算法的基本实现技术

一. 编码方法

编码是应用遗传算法时要解决的首要问题，它除了决定了个体的染色体排列形式之外，它还决定了个体从搜索空间的基因型变换到解空间的表现型时的解码方法，还影响到交叉算子，变异算子等遗传算子的运算方法。

De Jong 曾经提出了两条操作性较强的实用编码原则：

A. （有意义积木编码原则）：应使用能易于产生说求问题相关的且具有低价，短定义长度模式的编码方案。

B. （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。

上述编码原则仅仅给出了设计编码方案时的一个指导性大纲，它并不适合于所有的问题。所以对于实际应用问题，仍必须对编码方法，交叉运算方法，变异运算方法，解码方法等统一考虑，以寻求到一种对问题的描述最为方便，遗传运算效率最高的编码方案。

1．二进制编码方法

二进制编码符号串的长度与问题所要求的求解精度有关。假设某一参数的取值范围是[U_{min}，U_{max}],我们用长度为 l 的二进制编码符号串来表示该参数，则它总共能够产生 2^l 种不同不编码，若使参数编码时对应关系如下：

$$\begin{aligned}
&00000000\cdots00000000 = 0 && \longrightarrow && U_{min}\\
&00000000\cdots00000001 = 1 && \longrightarrow && U_{min}+\delta\\
&\quad\vdots\quad\vdots\quad\vdots && && \vdots\quad\vdots\\
&11111111\cdots11111111 = 2^l-1 && \longrightarrow && U_{max}
\end{aligned}$$

则二进制编码的编码精度为：

$$\delta = \frac{U_{max}-U_{min}}{2^l-1} \qquad (3\text{-}1)$$

假设某一个体的编码是：

$$X:\ b_l b_{l-1} b_{l-2}\cdots b_2 b_1$$

则其解码公式是：

$$x = U_{\min} + \left(\sum_{i=l}^{l} b_i \cdot 2^{i-1} \right) \cdot \frac{U_{\max} - U_{\min}}{2^l - 1} \qquad (3\text{-}2)$$

二进制编码方法有下述一些优点：

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）符合最小字符集编码原则；

4）便于利用模式定理对算法进行理论分析。

2．格雷码编码方法

二进制编码不便于反映出所求问题的结构特征，对于一些连续函数的优化问题的，也由于遗传运算的随机特性而使得其局部搜索能力较差。为此，人们提出了格雷码（Gray code）来对个体进行编码。

格雷码是这样的一种编码方法，其连续的两个整数所对应的编码值之间仅仅只有一个码位是不同的，其余码位都完全相同。例如下表所示：

表 3-1 二进制码与格雷码

十进制数	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

假设有一个二进制编码为 $B = b_m b_{m-1} \cdots b_2 b_1$，其对应的格雷码为 $G = g_m g_{m-1} \cdots g_2 g_1$。由二进制编码到格雷码的转换公式是：

$$\begin{cases} g_m = b_m \\ g_i = b_{i+1} \oplus b_i, \quad i = m-1, \ m-2, \ \cdots, \ 1 \end{cases} \tag{3-3}$$

由格雷码到二进制码的转换公式是：

$$\begin{cases} b_m = g_m \\ b_i = b_{i+1} \oplus g_i, \quad i = m-1, \ m-2, \ \cdots, \ 1 \end{cases} \tag{3-4}$$

其中的 $\oplus$ 表示异或运算符。

格雷码有这样一个特点：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）。这个特点是遗传算法中使用格雷码来进行编码的主要原因。

格雷码是二进制编码方法的一种变形，其编码精度与相同长度的二进制编码的精度相同。

格雷码编码方法的主要优点是：

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；

4） 便于利用模式定理对算法进行理论分析。

3．浮点数编码方法

二进制编码方法存在的缺点：

1） 二进制编码存在着连续函数离散化时的映射误差。

2） 二进制编码不便于反映所求问题的特定知识，不便于开发针对问题专门知识的遗传算法算子。

浮点数编码方法：指个体的每个基因值用某一范围内的一个　点数来表示，个体的编码长度等于其决策变量的个数。因为这种编码方法使用的时决策变量的　实值，所以也叫做　值编码方法。

例如，若某个优化问题有 5 个变量 x_i （i＝1，2，…，5），每个变量都有对应的上下　$\left[U^i_{\min}, U^i_{\max}\right]$,则：

X：	5.80	6.90	3.50	3.80	5.00

就表示一个体的基因型，其对应的表现型是：$x=[5.80, 6.90, 3.50, 3.80, 5.00]^T$ 。

在　点数编码方法中，必须保　基因值在给定的区间　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　制范围内。再者，当用多个字　来表示一个基因值时，交叉运算必须在两个基因的分界字　处进行，而不能够在某个基因的中间分　处进行。

点数编码方法有下面几个优点：

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） 便于较大空间的遗传搜索；
4） 改 了遗传算法的计算复杂性，提高了运算效率；
5） 5 便于遗传算法与经 优化方法的 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．符号编码方法

符号编码方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。这个符号集可以是一个子 表，如{A,B,C,…};也可以是一个数字序列号表，如{1,2,3,…};还可以是一个代码表，如{A1,A2,A3,…}等等。

符号编码的主要优点是：

1） 符合有意义积木 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） 便于遗传算法与相近算法之间的 合使用。

5．多参数级联编码方法

将各个参数分别一某种编码方法进行编码，然后将它们的编码按一定 序连接在一起就组成了表示全部参数的个体编码。这种编码方法称为多参数 联编码方法。

这种方法中，各个参数的编码可以是二进制，格雷码， 点数或者符号编码等，每个参数可以具有不同的上下界，不同的编码长度或者编码精度。

6．多参数交叉编码方法

基本 ：将各个参数中起主要作用的码位集中在一起，这样他们就不易于被遗传算子 。

在进行多参数交叉编码时，可先对各个参数进行分组编码（假设共有 n 个参数，每个参数都用长度为 m 的二进制编码串来表示）；然后取各个参数编码串中的最高位联接到一起，以他们作为个体编码串的前 n 位编码；再…..

多参数交叉编码方法特别适合于各个参数之间的相互关系较强，各个参数对最优解的 相当时的优化问题。

12．11

二． 适应度函数

遗传算法中，使用适应度来度量群体中各个个体在优化计算中有可能达到或接近于或有 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就较大；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）**。

1．目标函数与适应度函数

评价个体适应度的一个过程是：

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

2．适应度尺度变换

1）早熟现象：在遗传算法运行的初浮阶段，群体中可能会有少数几个个体的适应度非常高，若按照常用的比例选择算子来确定个体的遗传数量时，则可能它们 的比例非常高， 至全部是它们组成。这样，产生新个体作用较大的交叉算子就起不了 作用，使群体的多样性 低，所求的解 在某一局部最优点上。

2) 在遗传算法运行的后浮，群体中所有个体的平均适应度可能会接近于群体中最 个体的适应度，它们之间的无 力，进化过程 化成了一种随机选择的过程，导无法对某些 点区域进行 点搜索，从而影响遗传算法的运行效率。

为了 第一种情况，我们 在遗传算法的初浮阶段，算法能够对一些适应度较高的个体进行 制， 低其适应度与其他个体适应度之间的差异程度， 制其遗传到下一代的数量，保证群体的多样性；为了 第二种情况，我们 在遗传算法的后浮阶段，算法能够对个体的适应度进行适当的 大， 大最 个体适应度与其他个体适应度之间的差异程度，以提高个体之间的 性。

这种对个体适应度所做的 大或 小变换就称为**适应度尺度变换**（Fitness Scaling）。目前常用的个体适应度 度变换方法主要有三种：线性 度变换， 度变换和指数 度变换。

A． **线性尺度变换**。 变换公式如下：

$$F' = aF + b \qquad (3-8)$$

系数 a，b 直接影响到线性 度变换的大小，对其选取有一定的要求：

a．制度变换后全部个体的新适应度的平均值 F'_{avg} 要等于其原适应度平均值 F_{avg}。即： $F'_{avg} = F_{avg}$ （3－9）

这一条是为了保证群体中适应度接近于平均适应度的个体能够有浮 的数量被遗传到下一代群体中。

b． 度变换后群体中新的最大适应度 F'_{max} 要等于其平均适应度 F_{avg} 的指定数。即： $F'_{max} = C.F_{avg}$ （3－10）

式中，C 为最 个体的浮 复制数量，对于群体规模大小为 50～100 个个体的情况，一般选取 1.2~2。这条是为了保 群体中最好的个体能够浮复制 C 到新一代群体中。

B．**乘幂尺度变换**。 度变换公式为

$$F' = F^k \qquad (3-11)$$

k 于所求解的问题有关，并且在算法的执行过程中需要不断对其进行 正能够使得其 度变换满足一定的 要求。

C．**指数尺度变换**。 变换公式如下：

$$F' = \exp(- F) \qquad (3-12)$$

式中 决定了选择的强制性， 越小，原有适应度较高的个体的新适应度就越与其他个体的新适应度相差较大，即越增加了选择该个体的强制性。

三．选择算子

选择操作建立在对个体的适应度进行评价的基 之上。选择操作的主要目的是为了基因缺 ，提高全局 性和计算效率。

1． 比例选择（Proportional Model）

2． 最优保存策略

选择最好适应度的个体作为种子选手，直接保 到下一代。其具体操作过程是：

1） 找出当前群体中适应度最高的个体和适应度最低的个体；

2） 若当前群体中最 个体的适应度比总的 为止的最好个体的适应度还要高，则以当前群体的最 个体作为新的 为止的最好个体；

3） 用 为止的最好个体替换 当前群体中的最差个体。

最优保存策 可 为选择操作的一部分，它可以保证 为止所得到的最优个体不会被交叉，变异等遗传运算所 ，它是遗传算法 的一个 要保证。但是 一方面，它也易使得某个局部最优个体不易被 反而 散，从而使得算法的全局搜索能力不强。所以该方法一般要与其他一些选择操作方法配合起来使用，方可有良好的效果。

3．确定式采样选择（Deterministic Sampling）

其具体操作过程是：

1） 计算群体中各个个体在下一代群体中的浮 生存数模 N_i ：

$$N_i = M \cdot F_i / \sum_{i=1}^{M} F_i \qquad (i = 1,\ 2,\ \cdots,\ M)$$

2） 用 N_i 的整数部分确定各个对应个体在下一代群体中是生存数目。由该步共可确定处下一代群体中的个体总数 M’（对其整数部分求和）；

3） 按照 N_i 的小数部分对个体进行 序排序， 序取前 M-M’个个体加 到下一代群体中。至此可完全确定处下一代群体中的 M 个个体。

这种选择操作方法可保证适应度较大的一些个体一定能够被保 在下一代群体中。

4．无回放随机选择

这种选择操作也叫做**期望值选择方法（Expected Value Model）**，其基本 是根据每个个体在下一代群体中的生存浮 值进行随机选择运算。其具体操作过程是：

1） 计算群体中每个个体在下一代群体中的生存浮 数目 N_i：

$$N_i = M \cdot F_i / \sum_{i=1}^{M} F_i \qquad (i = 1, 2, \cdots, M)$$

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浮 数模 0.5，若某一个个体 被选中参与交叉运算，则它在下一代中的生存浮 数目 1.0；

3） 随着选择过程的进行，若某个个体的生存浮 数目小于 0 时，则这个个体再也不会被选中。

5．无回放余数随机选择（Remainder Stochastic Sampling with Replacement）

其具体操作过程是：

1） 计算群体中每个个体在下一代群体中的生存浮 数目 N_i：

$$N_i = M \cdot F_i / \sum_{i=1}^{M} F_i \qquad (i = 1, 2, \cdots, M)$$

2） 用 N_i 的整数部分确定各个对应个体在下一代群体中是生存数目。由该步共可确定处下一代群体中的个体总数 M’（对其整数部分求和）；

3） 以 $F_i - \lfloor N_i \rfloor \cdot \sum_{i=1}^{M} F_i / M$ 为各个个体的新的适应度，用比例选择方法来确定下一代中还 确定的个体。

12．13

6．排序选择（Rank-Based Model）

其主要着 点是个体适应度之间的大小关系，对个体适应度是否取正值或负值以及个体适应度之间的数值差异程度并无特 要求。

其主要 是：对群体中的所有个体按其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。其具体操作过程是：

1） 对群体中的所有个体按照其适应度大小进行 序排序；

2） 根据具体求解问题，设计一个概率分配表，将各个概率值按上述排序次序分配给各个个体；

3） 以各个个体所分配到的概率值作为其能够被遗传到下一代的概率，基于这些概率值所用比例选择的方法来产生下一代群体。

7．随机联赛选择（Stcochastic Tournament Model）

它也是一种基于个体适应度之间大小关系的选择方法。其基本 是：每次选择几个个体之间适应度最高的一个个体遗传到下一代全体中。

此方法中，每次进行适应度大小比较的个体数目称为联 规模。一般情况下，联 规模 N 的取值为 2。其具体操作过程是：

1）从群体中随机选择 N 个个体进行适应度大小的比较，将其中适应度最高的个体遗传到下一代群体中；

2）将上述过程 复 M 次，就可以得到下一代群体的 M 个个体。

四．交叉算子

遗传算法中的交叉运算，是指对两个互相配对的染色体 某种方式互相交换其部分基因，从而形成两个新的个体。遗传算法中，在交叉运算之前先对群体中的个体进行配对，最常用的配对策 是随机配对。交叉运算一般要求它即不要 多地 个体编码串中表示优良性状的优良模式， 要能够有效地产生出一些较好的新个体模式。 外，交叉算子的设计要和个体编码设计统一考虑。

交叉算子的设计包括一下两方面的内 ：

1） 如 确定交叉点的位置？

2） 如 进行部分基因交换？

最常用的交叉算子是单点交叉算子。

1．单点交叉（One－point Crossover）

它是指在个体编码串中只随机设置一个交叉点，然后在该点相互交换两个配对个体的部分染色体。单点交叉的 要特点是：若 接基因座之间的关系能提 较好的个体性状和较高的个体适应度的 ，则这种单点交叉操作 这种个体性状和 低个体适应度的可能性最小。

2．双点交叉与多点交叉（Two-point Crossover）

它是指在个体编码串中随机设置了二个交叉点，然后再进行部分基因交换。其具体操作过程是：

1） 在相互配对的两个个体编码串中随机设置两个交叉点；

2） 交换两个个体在所设定的两个交叉点之间的部分染色体。

将单点交叉和 点交叉的概率加以 ，可以得到多点交叉（multi－point crossover）的概念。

一般来说，不要使用多点交叉算子，它有可能 一些号的模式。交叉点越多，优良模式被 的可能性越大。

3．均匀交叉（Uniform Crossover）

指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，从而形成两个新的个体。均匀交叉实际上可以 于多点交叉的范围，其具体运算可通过设置一 字来确定个体的各个基因如 由哪一个 代个体来提 。其具体操作过程如下：

1）随机产生一个与个体编码串长度等长的 字 $W=w_1w_2\ldots w_i\ldots w_l$,其中 l 为编码串长度；

2）由下述规则从 A,B 两个 代个体中产生出两个新的子代 A',B'：

若 $w_i=0$，则 A'在第 i 个基因座上的基因值 A 对应基因值，B'在第 i 个基因座上的基因值 B 的对应的基因值。

4．算术交叉（Arithmetic Crossover）

指由两个个体的线性组合而产生出两个新的个体，通常这类交叉操作的对象是点数编码所表示的个体。 X_B^t

假设在两个个体 X_A^t， X_B^t 之间进行算术交叉，则交叉运算后所产生出的连 新个体是：

$$\begin{cases} X_A^{t+1} = \alpha X_B^t + (1-\alpha) X_A^t \\ X_B^{t+1} = \alpha X_A^t + (1-\alpha) X_B^t \end{cases} \qquad (3\text{-}14)$$

式中，a 为一参数，它可以是一个常数，此时所进行的交叉运算称为均匀算术交叉；它也可以是一个由进化代数所确定的变量，此时称为非均匀算术交叉。其主要操作过程是：

1） 确定两个个体进行线性组合时的系数 a；

2） 依据（3－14）生成两个新的个体。

五．变异算子

所谓变异运算，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的其他等位来替换，从而形成一个新的个体。

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；而变异运算只是产生新个体的 方法，但是它是必不可少的一个运算步骤，决定了遗传算法的局部搜索能力。

使用变异算子主要有以下两个目的：

1） 改 遗传算法的局部搜索能力；

2） 维 群体的多样性，防止出现早熟现象。

变异算子的设计内 包括如下两个方面的内 ：

1） 如 确定变异的位置？

2） 如 进行基因值替换？

最简单的变异算子是基本位变异算子。

1．基本位变异（Simple Mutation）

它是指对个体编码串中以变异概率 p_m 随机指定的某一位或某几位基因座上的基因值作变异运算。

2．均匀变异（Uniform Mutation）

它是指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。其具体操作过程是：

1） 依次指定个体编码串中的每个基因座为变异点；

2） 对每个变异点，以变异概率 p 从对应基因的取值范围内取一随机数来替代原有基因值。

均匀变异操作特别适合应用于遗传算法的初浮运行阶段，它使得搜索点可以在整个搜索空间内自由地 动，从而可以增加群体的多样性，使算法处理更多的模式。

3．边界变异（Boundary Mutation）

它是均匀变异操作的一个变形遗传算法，在进行操作时，随机地取基因座的二个对应界基因值之一 替代原有基因值。

12.14.2003

4．非均匀变异（Non－uniform Mutation）

均匀变异操作取某一范围内均匀分别的随机数来替换原有基因值，可使得个体在搜索空间自由 动。但 一个方面，它却不能够对某一 点区域进行局部搜索。为此，我们对原有基因值作一随机扰动，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后，相当于整个解向量在解空间中作了一个轻微的变得。这种变异操作方法就称为非均匀变异

非均匀变异的具体操作过程与均匀变异类似，但它 点搜索原个体附近的微小区域。

在进行由 $X=x_1x_2\ldots x_k\ldots x_l$ 向 $X'=x_1x_2\ldots x'_k\ldots x_l$ 的非均匀变异操作时，若变异点 x_k 处的基因值取值范围为$[U^k_{min}, U^k_{max}]$，则新的基因值 x'_k 由下式确定：

$$x'_k = \begin{cases} x_k + \Delta(t, U^k_{max} - v_k), & \text{if} \quad \text{random}(0,1) = 0 \\ x_k - \Delta(t, v_k - U^k_{min}), & \text{if} \quad \text{random}(0,1) = 1 \end{cases} \qquad (3\text{-}17)$$

式中， $\Delta(t,y)$（y 代表$U^k_{max} - v_k$ 和 $v_k - U^k_{min}$ ）表示[0,Y]范围内符合非均匀分布的一个随机数，要求随着进化代数 t 的增加，$\Delta(t,y)$ 接近于 0 的概率也逐渐增加。例如，$\Delta(t,y)$ 可照下式定义：

$$\Delta(t,y) = y \cdot (1 - r^{(1-t/T)^b}) \qquad (3\text{-}18)$$

式中，r 为[0，1]范围内符合均匀概率分布的一个随机数，T 是最大进化代数，b 是一个系统参数，它决定了随机扰动对于进化代数 t 的依赖程度。

5．高斯变异（Gaussian Mutation）

是改进遗传算法对 点搜索区域的局部搜索性能的 外一种变异操作方法。所谓告诉变异操作是指进行变异操作时，用符合均值μ，方差为 s^2 正态分布的一个随机数来替换原有基因值。

12．15．2003

六．遗传算法的运行参数

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，代沟 G 等。其选取的一般规则是：

1．编码串长度 l。 使用二进制编码来表示个体时，编码串长度 l 的选取与问题所要求的求解精度有关；使用 点数编码来表示个体时，编码串长度 l 与决策变量的个数 n 相等；使用符号编码来表示个体时，编码串长度 l 由问题的编码方式来确定； 外，也可以使用变长度的编码来表示个体。

2 群体大小 M。群体大小 M 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 度，但却 低了遗传算法的多样性，有可能会引起早熟现象；而当 M 取值交大时， 会使得遗传算法的运行效率 低。一般建议的取值范围是 20～100。

3 交叉概率 p_c 。交叉概率一般取值较大。但过大 易 群体中的优良模式，若过小产生新个体的 度较慢。一般建议取值范围是 0.4~0.99。 外，也可以使用自适应的 来确定交叉概率。

4 变异概率 p_m。变异概率较大时，可能 较好的模式； 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。 外也可以使用自适应的 来确定变异概率。

5 终止代数 T。一般建议取值范围是 100～1000，它还可以利用某种判定准则，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。常用的判定准则有下面两种：

1) 连续几代个体平均适应度的差异小于某一个极小的阈值；

2）群体中所有个体的适应度的方差小于某一个极小的阈值。

6．代沟 G。代沟 G 是表示各代群体之间个体 叠程度的一个参数，它表示每一个群体中被替换 的个体在全部个体中所 的百分率。

七．约束条件的处理方法

实际应用中的优化问题一般都含有一定的约束条件，它们的描述形式各种各样。目前尚无一般化方法，只能是针对具体应用问题及约束条件的特征，再考虑遗传算法中遗传算子的运行能力，选用不同的处理方法。在构造遗传算法时，处理约束条件的常用方法主要有如下三种：搜索空间 定法，可行解变换法，惩罚函数法。

1．搜索空间限定法

其基本 是：对遗传算法的搜索空间的大小加以 制，使得搜索空间中表示一个个体的点与解空间中表示一个可行解的点有一一对应的关系。

1）用编码方法来保证总是能够产生出在解空间中有对应可行解的染色体。这个实现要求我们设计出一种比较好的个体编码方案。

2）用程序来保证直到产生出解空间中有对应可行解的染色体之前，一直进行交叉运算和变异运算。

2．可行解变换法

其基本 是：在由个体基因型到个体表现型的变换中，增加使其满足约束条件的处理过程。即寻找出一种个体基因型和个体表现型之间的多对一的变换关系，使进化过程中所产生的个体总能够通过这个变换而转化成解空间中满足约束条件的一个可行解。

3．惩罚函数法

其基本 是：对在解空间中无对应可行解的个体，计算其适应度时，处以一个惩罚函数，从而 低该个体的适应度，使该个体被遗传到下一代群体中的机会 少。即用下式对个体的适应度进行调整：

$$F(x,y)=f(x,y)-a \bullet \max\{0,x^2+y^2-1\}$$

式子中，F(X)为原适应度，F’（X）为考虑了惩罚函数之后的新适应度，P（X）是惩罚函数。

例如：在处理 $x^2+y^2<=1$，并且 $x\in[-1,1]$， $y\in[-1,1]$ 这个约束条件时，融合惩罚函数的 ，可由下面的计算公式来计算个体的适应度：

$$F(x,y)=f(x,y)-a\bullet\max\{0,x^2+y^2-1\}$$

式中，a>0 就是确定惩罚函数作用强度的一个系数。

八．遗传算法的工具箱

第四章 遗传算法的高级实现技术

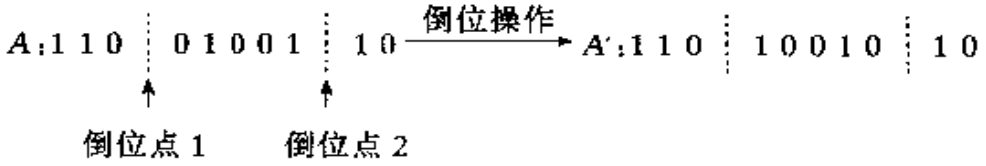
一． 倒位算子

1．倒位操作（Inverse Operation）

它是指颠倒个体编码串中随机指定的二个基因座之间的基因排列 序，从而形成一个新的染色体。其具体操作过程是：

1） 在个体编码串中随机指定二个基因值之后的位置为倒位点；

2） 以倒位概率 p_i 颠倒这二个倒位点之间的基因排列 序。如下例所示：



要说明的是，倒位算子只是个体编码串 新排序算子的一种，还有其他的一些 新排序算子，将在后面介绍。

2003-12-16

2．倒位算子应用示例

下面我们以栅格空间内的 动机器人路径规划为例，说明倒位算子的作用。

如图 4－1 所示为一 动机器人的 2 维规划空间，整个空间依据机器人的 寸被划分为一 一 的栅格，每个栅格有一标号，图中带有阴影的栅格表示障碍物，S 为机器人行走路线的起点，G 为终点。

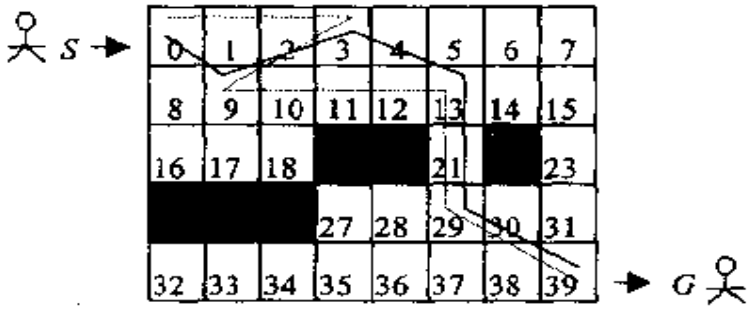


图 4-1 移动机器人的规划空间

用遗传算法进行及其人路径规划时，可取机器人 动过程中所经过的栅格之标号的序排列来作为一个个体（一条行走线路）的表现形式，如下所示即表示一条行走线路：

PATH：0－3－9－13－29－39 （虚线所示路线）

若在上述行走线路的第二个路径点和第三个路径点之间进行倒位操作，则可得到一条新的行走路线：

PATH：0－9－3－13－29－39 （实线所示路线）

如果以路径长短作为路径优劣的评价标准的 ，则新的行走路线比原有行走路线好。

二． 二倍体与显性操作算子

1．二倍体结构的生物基础

二 体结构中各个基因有显性基因（Dominance Gene）和隐形基因（Recessive Gene）之分。二类基因使个体所呈现出的表现型由下述规则来决定：在每个基因座上，当两个同源染色体其中之一为显性时，则该基因所对应的性状表现为显性；而仅当两个同源染色体中对应基因皆为隐性时，该基因所对应的性状 表现为隐性。

上述性状的特点是：显性基因在纯合子（AA）或杂合子（A 或 a）情况下均能够被表现出，而隐性基因只能在纯合子（aa）情况下 能被表现出。

二 体的意 ：

1） 二 体的记忆能力，使得生物能够记忆以前所经历过的环境变化，使得生物的遗传进化过程能够 地适应环境的变化。

2）显性操作的鲁棒性，它使得即使随机选择了适应度不高的个体，而在显性操作的作用下，能够用其 一同源染色体对其进行校正，从而 这个有害选择所带来的不利之处。这样能够提高遗传算法的运行效率，维护好的搜索群体。

2． 二倍体结构在遗传算法中的实现方案

Hollstien 提出了二 体与显性操作的 基因座显性映射方法。该方法中，每个二进制基因用两个基因来描述，一个称为函数基因，取通常含义的 0 或 1 直； 外一个称为饰基因，取值为 M 或 m，其中 M 表示显性基因，m 表示隐性基因。对函数基因取值为 0 的基因，当两个同源染色体中至少有一个 饰基因是 M 时，则该基因呈显性，否则该基因呈隐性。这种映射关系如图 4－2 所示。

	0M	0m	1M	1m
0M	0	0	0	0
0m	0	0	0	1
1M	0	0	1	1
1m	0	1	1	1

图 4-2 双基因座显性映射方法

3．使用双倍体的遗传算法描述

算法 DiploidyGA

1） 初始化，并设置进化代数计数器初值：t=1；

2） 随机产生具有二 体结构的初始群体 P（t）；

3） 对初始群体 P（t）进行显性操作；

4） 评价初始群体 P（t）中各个个体的适应度；

5） 交叉操作。有每两个随机配对的二 体个体进行交叉操作时，共可产生四个单体个体；

6） 变异操作。对群体中的各个个体进行变异操作时，需要考虑隐性基因的作用；

7） 对群体进行显性操作；

8） 评价现在群体中的各个个体的适应度；

9） 个体选择，复制操作；

10） 终止条件判断。若不满足终止条件，则跳转到（5）步， 续进化；否则，输出当前最优个体，算法结束。

2003-12-18

三．变长度染色体遗传算法

1. 变长度染色体遗传算法的编码与解码

将常规遗传算法的染色体编码串中各基因座位置及相应基因组成一个二元组，把这个二元组按一定 序排列起来，就组成了变长度染色体的一种通用染色体编码方式。一般它可以表示为：

$$X^m:(i_1, v_1)(i_2, v_2)\ \cdots\ (i_k, v_k)\ \cdots\ (i_n, v_n)$$

其中 i_k 是所描述的基因在原常规染色体中的基因座编号，v_k 为对应的基因值。

对于所求解的问题，若使用常规遗传算法时的染色体长度固定为 l，各基因值取自集合 V，则有：

$$1 \leqslant i_k \leqslant l \quad (k=1, 2, \cdots, n) \qquad (4\text{-}1)$$

$$v_k \in V \quad (k=1,2,\cdots,n) \qquad (4\text{-}2)$$

例如，常规染色体 X：100101，表示为 X^m：(1,1)(2,0)(3,0)(4,1)(5,0)(6,1)

2. 切断算子与拼接算子

变长度染色体遗传算法除了使用常规遗传算法中的选择算子合变异算子以外，不再使用交叉算子，而代之为使用下述的切断算子和拼接算子，以它们作为产生新个体的主要遗传算子。

1） 切断算子（Cut Operator）

切断算子以某一预先指定的概率，在变长度染色体中随机选择一个基因座，在该处将个体的基因型切断，使之称为二个个体的基因型。

2） 拼接算子（Splice Operator）

拼接算子以某一预先指定的概率，将二个个体的基因型连接在一起，使它们合并为一个个体的基因型。

3．变长度染色体遗传算法的算法结构

与基本遗传算法类似

四．小生境遗传算法

4．4．1 小生境与遗传算法

让遗传算法的个体在一个特定的生存环境中进化，以找出更多的最优解，从而利于求解出多峰值函数的优化计算问题。

4．4．2 遗传算法中小生境的实现方法

1． 基于预选择（Preselection）的小生境实现方法

其基本 是：仅当新产生的子代个体的适应度超过其 代个体的适应度时，所产生出的子代个体 能替换其 代个体而遗传到下一代群体中，否则 代个体仍保 在下一代群体中。

2．基于排挤（Crowding）的小生境实现方法

其基本 是：设置一排挤因子 CF，由群体中随机选择的 1/CF 个个体组成排挤成员，然后依据新产生的个体与排挤成员的相似性来排挤 一些与排挤成员相类似的个体。这里的个体的相似性可用个体编码串之间的海明距离来度量。

3．基于共享函数（Sharing）的小生境实现方法

其基本 是：通过反映个体之间相似程度的共享函数来调整群体中各个个体的适应度，从而在这以后的群体进化过程中，算法能够依据这个调整后的新适应度来进行选择运算，以维护群体的多样性，创造出小生境的进化环境。

共享函数（Sharing Function）是表示群体中两个个体之间密切关系程度的一个函数，可记为 S（d_{ij}），其中 d_{ij} 表示个体 i 和个体 j 之间的某种关系。

2003-12-22

4．4．3 小生境遗传算法在多峰值函数全局最优中的应用

基本思想：首先两两比较群体中各个个体之间的距离，若这个距离在预先指定的距离 L 之内的 ，再比较两者之间的适应度的大小，并对其中适应度较低的个体施加一个较强的惩罚函数，极大地 低起 能力，这样，对于在预先指定的某一距离 L 之内的两个个体，其中较差的个体经处理后，其适应度变得更差，它在后面的进化过程中被 的概率就极大。也就是说，在距离 L 之内将只存在一个优良的个体，从而既维护了群体的多样性， 使得各个个体之间保 一定的距离，并使得个体能够在整个约束空间中分散开来，这样就实现了一种小生境遗传算法。

小生境算法描述如下：

a．设置进化代数计数器 t＝1，随机生成 M 个初始个体组成初始群体 P（t），并求出各个个体的适应度 F_i（i＝1，2，…，M）；

b．根据各个个体的适应度对其进行 序排序，记忆前 N 个个体（N<M）；

c．选择运算。对群体 P（t）进行比例选择运算，得到 P’（t）；

d．交叉算子。对选择出的个体集合 P’（t）做单点交叉运算，得到 P’’（t）；

e．变异运算。对 P’’（t）做均匀变异运算，得到 P’’’(t)；

f．小生境 运算。将 e 中得到的 M 个个体和第 b 步所记忆的 N 个个体合并在一起，得到一个含有 M＋N 个个体的新群体；对这 M+N 个个体，按照下式求出每两个个体 X_i 和 X_j 之间的海明距离：

$$\| X_i - X_j \| = \sqrt{\sum_{k=1}^{M} (x_{ik} - x_{jk})^2} \quad \begin{pmatrix} i=1,2,\ldots,M+N-1 \\ j=i+1,\ldots,M+N \end{pmatrix}$$

当 $\|X_i-X_j\|<L$ 时，比较个体 X_i 和个体 X_j 的适应度的大小，并对其中适应度较低者的个体 以罚函数： F_{min}（ X_j）＝Penalty

g．更加这 M+N 个个体的新适应度对各个个体进行 序排序，记忆前 N 个个体；

h．终止条件判断。若不满足终止条件，则：更新进化代数计数器 t＝t＋1，并将第 g 步排序中的前 M 个个体作为新的下一代群体 P（t），然后转到第 c 步；若满足终止条件，则：输出计算结果，算法结束。

例：Shubert 函数的全局最优化计算。

$$\min f(x_1,x_2)=\{\sum_{i=1}^{5} i\bullet\cos[(i+1)x_1+i]\}*\{\sum_{i=1}^{5} i\bullet\cos[(i+1)x_2+i]\}$$

$$S.T. \quad -10\le x_i\le 10 \qquad (i=1,2)$$

上述函数共有 760 个局部最优点，其中 18 个是全局最优点，全局最优点处的目标函数值是 f_{opt}（x_1，x_2）＝－186. 731。

用上述小生境遗传算法求解该例题时，可用下式进行目标函数值到个体适应度的变换处理：

$$F(x_1,x_2)=\begin{cases}1-0.05f(x_1,x_2) & (if \quad f(x_1,x_2)<0)\\ 1 & (if \quad f(x_1,x_2)\ge 0)\end{cases}$$

运用该算法时所使用的运行参数是：

l＝20×2（二进制编码串长度，其中每个变量用 10 位二进制编码来表示）

M=50，T=500，p_c＝0.8，p_m＝0.1，L=0.5 (小生境的距离参数) Penlty＝10^{-30}（罚函数）

4．5 混合遗传算法

4．5．1 混合遗传算法的思想

遗传算法的主要问题： 易产生早熟现象，局部寻最优能力较差，并且通常其结果比针对问题的启发式算法的求解效率要差， 外，遗传算法也无法 多次搜索同一个可行解。

合遗传算法的 ：

a．引 了局部搜索过程。

b. 增加了编码变换操作过程。

合遗传算法的基本构成原则

a．尽量采用原有算法的编码。

b．利用原有算法的优点。

c. 改进遗传算子。

2003-12-23

4．5．2 模拟退火算法（Simulated Annealing）

模拟 火算法是基于金 火的机理而建立起的一种全局最优化方法，它能够以随机搜索技术从概率的意义上找出目标函数的全局最小点。模拟 火算法的构成要素如下：

1．搜索空间 Ω

搜索空间也称为状态空间，它由可行解的集合所组成，其中的一个状态 x 就代表一个可行解。

2．能量函数 E(x)

能量函数也就是需要进行优化计算的目标函数，其最小点为所求的最优解。

3．状态转移规则 P

状态转 是指从一个状态 x_{old}（一个可行解）向 一个状态 x_{new}（ 一个可行解）的转 概率，它与当前的温度参数 T 有关。

4．冷却进度表 T(t)

冷却进度表是指从某一高温状态 T_0 向低温状态冷却时的 温管理表。

假设时刻 t 的温度 T(t)，则经 的模拟 火算法的 温方式为：

$$\begin{cases} \max \quad f(x_1,x_2,...,x_n)=\sum_{i=1}^{n} c_i x_i \\ s.t. \quad \sum_{i=1}^{n} w_i x_i \le v \\ x_i \in \{0,1\} \quad (i=1,2,...,n) \end{cases}$$

而 模拟 火算法的 温方式为：

$$T(t)=\frac{T_0}{1+t}$$

这两种方式都能够使得模拟 火算法 于全局最小点。

外，在实际应用中，为计算简便起见，也可用下式进行温度管理：

T(t)＝k·T（t－1）

式子中，k 为 小于 1.0 的系数。

假设在状态 x_{old} 时，系统受到某种扰动而可能会使其状态变为 x_{new}。与此相对应，系统的能量也可能会从 $E(x_{old})$变成 $E(x_{new})$。系统由状态 x_{old} 变成 x_{new} 的接受概率可由下面的 Meteopolis 规则来确定：

$$p=\begin{cases} 1 & if \quad E(x_{new})<E(x_{old}) \\ \exp(-\frac{E(x_{new})-E(x_{old})}{T}) & if \quad E(x_{new})\ge E(x_{old}) \end{cases}$$

上式的含义是：当新状态使系统的能量函数值 小时，系统一定接受这个新的状态；而当新状态使系统的能量函数值增加时，系统以某一概率接受这个新的状态。

固定温度参数 T，反复进行状态转 过程，接受概率 p（x）将 从 Gibbs 分布：

$$p(x)=\frac{1}{Z}\exp(-\frac{E(x)}{T})$$

式中，Z 是使得概率值正规划的系数。

模拟 火算法可以描述如下：

算法 SimulatedAnnealing

① 随机产生一个初始最优点，以它作为当前最优点，并计算目标函数值。

② 设置初始温度：$\theta \leftarrow T_0$。

③ 设置循环计数器初值：$t \leftarrow 1$。

④ 对当前最优点作一随机变动，产生一新的最优点，计算新的目标函数值，并计算目标函数值的增量 Δ。

⑤ 如果 $\Delta < 0$，则接受该新产生的最优点为当前最优点；
如果 $\Delta \geqslant 0$，则以概率 $p = \exp(-\Delta/\theta)$ 接受该新产生的最优点为当前最优点。

⑥ 如果 $t <$ 终止步数，则：$t \leftarrow t + 1$，转向第 ④ 步。

⑦ 如果未到达冷却状态，则：$\theta \leftarrow T(t)$，转向第 ③ 步；
如果已到达冷却状态，则：输出当前最优点，计算结束。

4．5．4 遗传模拟退火算法

基本 ：遗传模拟 火算法是将遗传算法与模拟 火算法相结合而构成的一种优化算法。遗传算法的局部搜索能力较差，但把握搜索过程总体的能力较强；而模拟 火算法具有较强的局部搜索能力，并能使搜索过程 陷 局部最优解，但模拟 火算法却对整个搜索空间的状况了解不多，不便于使搜索过程进 最有 的搜索区域，从而使得模拟 火算法的运行效率不高。因而将两者结合，取长补短，可能开发出性能优良的全局搜索算法。

遗传模拟 火算法描述：

算法 GeneticSimulatedAnnealing

① 进化代数计数器初始化：$t \leftarrow 0$。

② 随机产生初始群体 $P(t)$。

③ 评价群体 $P(t)$ 的适应度。

④ 个体交叉操作：$P'(t) \leftarrow \text{Crossover}[P(t)]$。

⑤ 个体变异操作：$P''(t) \leftarrow \text{Mutation}[P'(t)]$。

⑥ 个体模拟退火操作：$P'''(t) \leftarrow \text{SimulatedAnnealing}[P''(t)]$。

⑦ 评价群体 $P'''(t)$ 的适应度。

⑧ 个体选择、复制操作：$P(t+1) \leftarrow \text{Reproduction}[P(t) \cup P'''(t)]$。

⑨ 终止条件判断。若不满足终止条件，则：$t \leftarrow t + 1$，转到第 ④ 步，继续进化过程；若满足终止条件，则：输出当前最优个体，算法结束。

一种构造 合遗传算法的方法是在模拟 火算法中溶 遗传算法的 。例如，Mathefound 所开发的并行组合模拟 火算法 PRSA(Parellel Recombination Simulated

Annealing)，其算法描述为：

① 随机生成含有 M 个个体的初始群体 $P(0)$。

② 设置初始温度参数：$T \leftarrow T_{max}$。

③ 对 $P(t)$ 中的各个个体进行随机配对，对其中的每一对个体组作下述处理：

· 进行交叉和变异运算，由两个父代个体 p_1、p_2 生成两个子代个体 c_1 和 c_2。

· 对由父代个体和子代个体所组成的两个个体组 p_1 和 c_1、p_2 和 c_2，以概率 p 接受父代个体为下一代群体中的个体，以概率 $(1-p)$ 接受子代个体为下一代群体中的个体。

其中，$$p = \frac{1}{1 + \exp\left(\frac{f_p - f_c}{T}\right)} \qquad (4\text{-}12)$$

式中，f_p 和 f_c 分别为父代个体和子代个体所对应的目标函数值。

④ 终止条件判断。若不满足终止条件，则：按降温表更新温度参数 T，$t \leftarrow t+1$，转向第 ③ 步；若满足终止条件，则：输出当前最优点，算法结束。

4．5．5 混合遗传算法在求解背包问题中的应用

背包问题的一般提法是：已知 n 个物品的 量及其价值分别为 $w_i>0$ 和 $c_i>0$（i＝1,2,…n），背包的 量假设为 v>0，如 选择哪些物品 该背包可使得在背包的 量约束 制之内所物品的总价值最大？

该问题的模型可表示为下述整数规划模型：

$$\begin{cases} \max \quad f(x_1, x_2, ..., x_n) = \sum_{i=1}^{n} c_i x_i \\ s.t. \quad \sum_{i=1}^{n} w_i x_i \le v \\ x_i \in \{0,1\} \quad (i = 1, 2, ..., n) \end{cases}$$

式中 x_i 为 0－1 决策变量，x_i＝1 表示将物品 i 背包中，反之则表示不 。

背包问题是一个 型的 NP 完全（Nondeterministic Polynomial Completeness）难题，对该问题的求解可以解决管理中的资源分配，投资决策， 载问题等实际问题，采用的方法主要是一些启发式算法（如贪婪算法等），也可以用遗传算法来求解该问题。

采用基于贪婪算法的染色体编码串转换算法的描述如下：

算法 DecodeByGreedyAlgorithm

① 对所有 $x_i = 1$ 的物品，按它们的价值密度 $\rho_i = c_i / w_i (i = 1, 2, \cdots, n)$ 从大到小排序，形成一个队列 $q(i)$，其中 $q(1)$ 为价值密度最大的物品序号，$q(2)$ 为价值密度次大的物品序号，等等。

② 置 $k = 1$。

③ 利用贪婪算法进行解码变换。

如果 $\sum_{j=1}^{k} w_{q(j)} \leqslant v$，则将 $q(k)$ 所对应的物品装入背包中，即置：$x_{q(k} = 1$、$k = k + 1$，转向第 ③ 步；

如果 $\sum_{j=1}^{k} w_{q(j)} > v$，则转向第 ④ 步。

④ 对于队列序号 j 从 k 到 n，它们所对应的物品不放入背包中，即置 $x_{q(j} = 0$。

第五章 并行遗传算法

5．1 遗传算法的并行化

5．1．1 遗传算法并行化的目的

主要目的的提高遗传算法的运行 度。

5．1．2 遗传算法的并行性分析

Goldberg 对基本遗传算法的 纳

遗传算法可以从下面四种并行性方面着手对其进行改进和发展。

1．并行性 I: 个体适应度评价的并行性
2．并行性 II: 整个群体中各个个体适应度评价的并行性
3．并行性 III：子代群体产生过程的并行性
4．并行性 IV：基于群体分组的并行性

5．1．3． 并行遗传算法的实现方法分类

为了提高遗传算法的运算 度，改 其求解性能，对基于上述各种并发机制的发掘和利用，人们在并行计算机或局域网环境下开发出了多种不同的并行遗传算法。概况起来，实现这些并行遗传算法的方法大体上可以分为如下两类：标准型并行方法（Standard Parallel Approach）和分解型并行方法（Decomposition Parallel Approach）。

标准型并行方法并 改变串行遗传算法的基本特点，它在一个总体的环境中实现进化计算，所以它需要实用一个统一的群体，实现时也需要有一个全局存储器，还需要有一个同意的控制机构来协调群体的遗传进化过程及群体之间的通讯过程。

分解型并行方法将整个群体分解为几个子群体，各个子群体被分配在不同的处理机上，它们各自串行运行所在处理机上的基本遗传算法，然后在适当的时候，各处理机之间相互交换一些信息。

遗传算法的并行机制

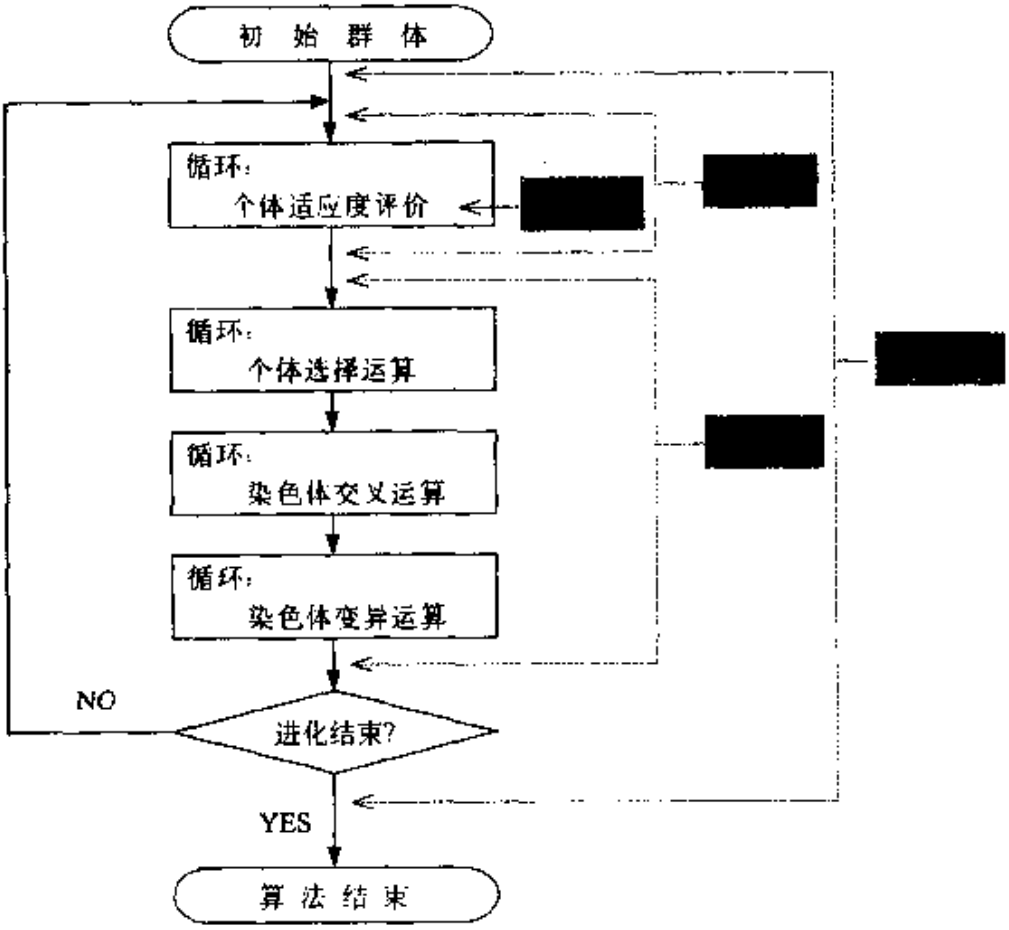


图 5-1 遗传算法的并行机制

2003-12-28

5．1．4 并行遗传算法的硬件支持环境及性能评价

加速比

设 T_1 为某算法在串行计算机上的运行时间，T_p 是该算法在由 p 各处理机说构成的并行机上的运行时间，则此算法在该并行计算机上的加 比 S_p 定义为：

$$S_p = T_1/T_p$$

5．2 实现并行遗传算法的标准型并行方法

5．2．1 标准型并行方法的基本思想

这类并行方法不改变简单遗传算法的基本结构特点，即群体的全部个体都在统一的环境中进化。其基本出发点是从局部的角度开发个体进化的并行性。在应用遗传算法方法进行优化计算时，各个个体的适应度评价以及染色体的选择、交叉和变异等操作是可以相互单独进行的。这样，利用具有共享存储器结构的并行机，就可对群体的进化过程进行并行计算以达到提高遗传算法运行 度的目的。这类方法在个体适应度计算量较大的场合比较有效。

5．2．2 算法实例

略

5．3 实现并行遗传算法的分解型并行方法

5．3．1 分解型并行方法的基本思想

这种方法将整个群体划分为几个子群体，各个子群体分配在各自的处理机或局域网工作站上独立地进行简单遗传算法的进化操作，在适当的时候各个子群体之间相互交换一些信息。其基本出发点是从全局的角度开发群体进化的并行性。这种方法改变了简单遗传算法的基本特点，即各子群体独立地进行进化，而不是全部群体采用同一机制进化。

在构造这种并行遗传算法时，需要考虑下述几个主要问题：

1．子群体划分方式。

2．信息交换方式。

1） 参加信息交换的对象（哪几个处理机之间可以相互交换信息？）；

2） 交换信息的内 （是随机交换，还是择优交换？）；

3） 交换时间或频率（ 时交换？）；

4） 交换信息量（交换几个个体？）

5．3．2 分解型并行遗传算法的形式化定义与描述

基于群体分组的并行遗传算法的运算过程是：各个处理机运行各自的基本遗传算法，对它自身所带的存储器中的 M/p 个个体进行遗传算法操作；当达到一个预先指定的信息交换时间 T 时，每个处理机 p_i 发送需要交换的信息 Z 到其信息交换对象 X^i；同时，它也接受来自于 X^i 的信息，用该信息来替换本处理机上的某一个或者某一些个体。上述过程不断地迭代进行，直到满足某个终止条件为止。

5．3．3 子群体信息交换模型

实现分解并行遗传算法所采用的群体模型主要有三种：

1．踏脚石群体模型（Stepping－stone Model）

2．岛屿群体模型（Island Model）

3. 居群体模型（Neighborhood Model）

5．4 伪并行遗传算法

5。4．1 伪并行遗传算法的基本思想

在简单遗传算法中，利用并行遗传算法的 ，将群体划分为一些子群体，各子群体按一定的模式分别进行独立进化，在适当的时候，某一些子群体之间交换一些信息，这样可以维 群体的多样性，从而达到抑制早熟现象的效果。

5．4．2 伪遗传算法的应用举例

例 1． 六峰值驼背函数（Six－hump Camel Back Function）的全局最小值计算问题：

$$\min \quad f(x,y)=(4-2.1x^2+\frac{x^4}{3})x^2+xy+(-4+4y^2)y^2$$

$$\text{s.t.} \quad -3\leqslant x\leqslant 3$$

$$-2\leqslant y\leqslant 2$$

该函数有六个局部最小点，其中（－0.0898,0.7126）和（0.0898 –0.71216）为全局最小点,最小值为 –1.031628。

例 2． Rosenbrock 函数的全局最大值计算问题：

$$\max \quad f(x_1, x_2)=100(x_1^2-x_2)^2+(1-x_1)^2$$

$$\text{s.t.} -2.048\leqslant x_i\leqslant 2.048 \qquad (i=1, 2)$$

该函数有两个局部最大点 f(2.048, -2.048)=3897.7342 和 f(-2.048, -2.048)=3905.9262，其中，后者为全局最大点。

第六章 遗传算法的数学理论

6．1 模式定理

6．1．1 模式

【定义 6.1 】 模式（Schema）表示一些相似的模 ，它描述了在某些位置上具有相似结构特征的个体编码串的一个子集。

以二进制编码方式为例，个体是由二值字符集 V＝{0, 1}中的元素组成的一个编码串，而 模式却是由三值字符集 V_+={0, 1, * } 中的元素说组成的一个编码串，其中的“*”表示通配符，它既可被当作“1”，也可被当作“0”。

例如， 模式 H＝11**1 描述了长度为 5，且在位置 1、2、5 取值为“1”的所有字符串的集合{ 11001，11011，11101，11111}。

遗传算法的本质是对模式所进行的一系列运算，即通过选择算子将当前群体中的优良模式遗传到下一代群体中，通过交叉算子对模式的 组，通过变异算子对模式的突变。通过这些遗传运算，一些较差的模式逐步被 ，而一些较好的模式逐步被遗传和进化，最终就可得到问题的最优解。

【定义 6.2】 模式 H 中具有确定基因值的位置数目称为该模式的模式阶（Schema Order），记为 o（H）。

当字符串的长度固定时，模式阶越高，能与该模式匹配的字符串数就越少，因而该模式的确定性就越高。

【定义 6.3】 模式 H 中第一个确定基因值的位置和最后一个确定基因值的位置之间的距离称为该模式的模式定义长度（Schema Defining Length），记为 δ(H)。

例如，δ（11*0**）=3，δ(0 * * * 1)=4。对于只有一个确定基因值的模式，规定它们的模式长度为 1。

6．1．2 模式定理

【模式定理】 遗传算法中，在选择、交叉和变异算子的作用下，具有低价、短的定义长度，并且平均适应度高于群体平均适应度的模式将按指数 增长。

6．2 积木块假设与遗传算法欺骗问题

6．2．1 积木块假设

模式定理说明了具有结构特征的模式在遗传进化过程中其样本将按指数 增长，这种模式就是具有低价、短的定义长度，且平均适应度高于群体平均适应度的模式。这种类型的模式被称为基因 或积木 （Building Block）。

【积木块假设】 个体的基因 通过选择、交叉、变异等遗传算子的作用，能够相互拼接在一起，形成适应度更高的个体编码串。

6．2．2 遗传算法的欺骗问题（GA Deceptive Problem）

优良模式周围存在不良模式包围所导

6．3 隐含并行性

6．4 遗传算法的收敛性分析

6．5 适应度函数的自相关分析

第七章 遗传算法的应用

7．1 数值函数优化计算

7．1．1 遗传算法与纯数值函数优化计算

7．1．2 评价遗传算法性能的常用测试函数

1. De Jong 函数 F1：

$$\begin{cases} f_1(x_1,x_2,x_3)=\sum_{i=1}^{3}x_i^2 \\ -5.12\leqslant x_i\leqslant 5.12 \qquad (i=1,2,3) \end{cases}$$

这是一个简单的平方和函数，只有一个极小点 f_1（0，0，0）=0。

2. De Jong 函数 F2：

$$\begin{cases} f_2(x_1,x_2)=100(x_1^2-x_2)^2+(1-x_1)^2 \\ -2.048\leqslant x_i\leqslant 2.048 \qquad (i=1,2) \end{cases}$$

3. De Jong 函数 F3：

$$\begin{cases} f_3(x_1,x_2,\cdots,x_5)=\sum_{i=1}^{5}\text{integer}(x_i) \\ -5.12\leqslant x_i\leqslant 5.12 \qquad (i=1,2,\cdots,5 \end{cases}$$

一个全局最小值为 30。

4. De Jong 函数 F4：

$$\begin{cases} f_4(x_1,x_2,\cdots,x_{30})=\sum_{i=1}^{30}ix_i^4+\text{Gauss}(0,1) \\ -1.28\leqslant x_i\leqslant 1.28 \qquad (i=1,2,\cdots,30) \end{cases}$$

一个全局最小值 f4（0，0，……，0）=0。

5. De Jong 函数 F5：

$$\begin{cases} f_5(x_1,x_2)=0.002+\sum_{j=1}^{25}\dfrac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6} \\ -65.536\leqslant x_i\leqslant 65.536 \qquad (i=1,2) \\ [a_{ij}]= \\ \begin{bmatrix} -32 & -16 & 0 & 16 & 32 & -32 & -16 & \cdots & 0 & 16 & 32 \\ -32 & -32 & -32 & -32 & -32 & -16 & -16 & \cdots & 32 & 32 & 32 \end{bmatrix} \end{cases}$$

一个全局最小点 f5（－32，－32）=0.998。

6．Schaffer 函数 F6：

$$\begin{cases} f_6(x_1, x_2) = 0.5 + \dfrac{\sin^2\sqrt{x_1^2 + x_2^2} - 0.5}{[1.0 + 0.001(x_1^2 + x_2^2)]^2} \\ -100 \leqslant x_i \leqslant 100 \qquad (i = 1, 2) \end{cases}$$

该函数有一个全局最小点 f6（0，0）＝0。

7．Schaffer 函数 F7：

$$\begin{cases} f_7(x_1, x_2) = (x_1^2 + x_2^2)^{0.25}[\sin^2(50(x_1^2 + x_2^2)^{0.1}) + 1.0] \\ -100 \leqslant x_i \leqslant 100 \qquad (i = 1, 2) \end{cases}$$

该函数有一个全局极小点 f7（0，0）＝0。

8．Goldstein－Price 函数：

$$\begin{cases} f(x_1, x_2) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 \\ \qquad - 14x_2 + 6x_1x_2 + 3x_2^2)] \cdot \\ \qquad [30 + (2x_1 - 3x_2)^2(18 - 32x_1 \\ \qquad + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \\ -2 \leqslant x_i \leqslant 2 \qquad (i = 1, 2) \end{cases}$$

该函数有一个全局极小点 f（0，－1）＝3。

9. Shubert 函数：

$$\begin{cases} f(x_1, x_2) = \sum_{i=1}^{5} i\cos[(i+1)x_1 + i] \cdot \sum_{i=1}^{5} i\cos[(i+1)x_2 + i] \\ -10 \leqslant x_i \leqslant 10 \qquad (i = 1, 2) \end{cases}$$

多峰值函数，有 18 个全局最小点，f＝186.731。

10． 六峰值驼背函数

$$\begin{cases} f(x, y) = \left(4 - 2.1x^2 + \dfrac{x^4}{3}\right)x^2 + xy + (-4 + 4y^2)y^2 \\ -3 \leqslant x \leqslant 3 \\ -2 \leqslant y \leqslant 2 \end{cases}$$

该函数有六个局部极小点，其中（－0.0898,0.7126）和（0.0898,-0.7126）为全局最小点，最小值为－1.031628。

11. 带有复杂约束条件的函数（之一）

$$\begin{cases} \min \quad f(x,y) = 5x_1 + 5x_2 + 5x_3 + 5x_4 - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=1}^{9} y_i \\ \text{s.t.} \quad 2x_1 + 2x_2 + y_6 + y_7 \leqslant 10 \quad 2x_1 + 2x_3 + y_6 + y_8 \leqslant 1 \\ \qquad 2x_2 + 2x_3 + y_7 + y_8 \leqslant 10 \quad -8x_1 + y_6 \leqslant 10 \\ \qquad -8x_2 + y_7 \leqslant 0 \qquad -8x_3 + y_8 \leqslant 0 \\ \qquad -2x_4 - y_1 + y_6 \leqslant 0 \qquad -2y_2 - y_3 + y_7 \leqslant 0 \\ \qquad -2y_4 - y_5 + y_8 \leqslant 0 \qquad 0 \leqslant x_i \leqslant 1 (i = 1,2,3,4 \\ 0 \leqslant y_i \leqslant 1 (i = 1,2,3,4,5,9) \qquad 0 \leqslant y_i (i = 6,7,8) \end{cases}$$

全局极小点 f（1，1，1，1，1，1，1，1，1，1，1，1，3，3，3，1）＝－15。

12. 带有复杂约束条件的函数（之二）

$$\begin{cases} \max f(x_1, x_2, x_3) = \dfrac{3x_1 + x_2 - 2x_3 + 0.8}{2x_1 - x_2 + x_3} + \dfrac{4x_1 - 2x_2 + x_3}{7x_1 + 3x_2 - x_3} \\ \text{s.t.} \quad x_1 + x_2 - x_3 \leqslant 1 \\ \qquad -x_1 + x_2 - x_3 \leqslant -1 \\ \qquad 12x_1 + 5x_2 + 12x_3 \leqslant 34.8 \qquad (7\text{-}12) \\ \qquad 12x_1 + 12x_2 + 7x_3 \leqslant 29.1 \\ \qquad -6x_1 + x_2 + x_3 \leqslant -4.1 \\ 0 \leqslant x_i \quad (i = 1, 2, 3) \end{cases}$$

该函数的全局最大点为：f（1，0，0）＝2.471428。