第 5 章

模拟生物进化的遗传算法

受自然界规律的启迪，人们设计了许多求解问题的算法，包括人工神经网络、模糊逻辑、遗传算法、DNA 计算、模拟退火算法、禁忌搜索算法、免疫算法、膜计算、量子计算、粒子群优化算法、蚁群优化算法、人工蜂群算法、人工鱼群算法以及细菌群体优化算法等，这些算法皆称为智能计算（Intelligent Computing，IC），也称为计算智能（Computational Intelligence，CI）。智能优化方法通常包括进化计算和群智能两大类，目前已经广泛应用于组合优化、机器学习、智能控制、模式识别、规划设计、网络安全等领域，是 21 世纪智能计算的重要技术之一。

本章首先简要介绍进化算法的概念，详细介绍基本遗传算法，它是进化算法的基本框架；然后介绍双倍体、双种群、自适应等比较典型的改进遗传算法及其应用；最后介绍基于遗传算法的生产调度方法。

5.1 进化算法的生物学背景

> 达尔文与进化论。1858 年 7 月 1 日，达尔文（C. R. Darwin）与华莱士（A. R. Wallace）在伦敦林奈学会上宣读了关于进化论的论文。进化论认为，自然选择是生物进化的动力。生物都有繁殖过盛的倾向，而生存空间和食物是有限的，生物必须“为生存而斗争”。在同一种群中的个体存在着变异，那些具有能适应环境的有利变异特性的个体将存活下来，并繁殖后代；而那些不具有有利变异特性的个体就会被淘汰。如果自然条件的变化是有方向的，则经过长期的自然选择，微小的变异就得到积累而成为显著的变异，由此可能导致亚种和新物种的形成。

进化算法（Evolutionary Algorithm，EA）是以达尔文的进化论思想为基础，通过模拟生物进化过程与机制求解问题的自组织、自适应的人工智能技术，是借鉴生物界自然选择和自然遗传机制的随机搜索算法，进化算法中的各种方法本质上从不同的角度对达尔

文的进化原理进行了不同的运用和阐述,非常适用于传统搜索方法难以解决的复杂的和非线性的优化问题。

进化算法是一个算法簇,包括遗传算法(Genetic Algorithm,GA)、遗传规划(Genetic Programming)、进化策略(Evolution Strategy)和进化规划(Evolution Programming)等。尽管它们有不同的遗传基因表达方式、不同的交叉算子、变异算子和其他特殊算子以及不同的再生和选择方法,但它们的灵感都来自大自然的生物进化。进化算法的基础是遗传算法所描述的框架。

与传统的基于微积分的方法和穷举法等优化算法相比,进化算法是一种具有高鲁棒性和广泛适用性的全局优化方法,具有自组织、自适应、自学习的特性,能够不受问题性质的限制,能适应不同的环境和不同的问题,有效地处理传统优化算法难以解决的大规模复杂优化问题。

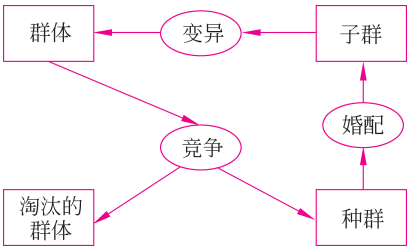

图 5.1 生物进化的基本过程

进化算法类似于生物进化,需要经过长时间的成长和演化,最后收敛到最优化问题的一个或者多个解。因此,了解一些生物进化过程,有助于理解遗传算法的工作过程。

"适者生存"揭示了大自然生物进化过程中的一个规律:适应自然环境的群体往往产生较大的后代群体。生物进化的基本过程如图 5.1 所示。

羚羊群进化过程。有群体的地方就有竞争。羚羊竞争的标准是力气大、速度快。力气大,占有的配偶就多,繁殖的后代就多;速度快,被吃掉的可能性就小。当然,力气小的也可能繁殖后代,只是可能性比较小;跑得快的也可能被吃掉,只是被吃掉的可能性比较小。竞争胜出者进入种群,以一定的随机性选择配偶交配,繁殖后代。后代继承了父母的基因,称为遗传,但其生物特性和父母又不完全一样,会发生一些变化,称为变异。新生的小羚羊加入群体。新的群体再进行竞争,进入新一轮进化。这样不断进化,羚羊群总体上越来越优秀。羚羊群进化过程中的随机性能够使得种群具有多样性。遗传算法模拟生物进化的机理,能够得到最优化问题的最优解。

生物的遗传物质的主要载体是染色体(chromosome),DNA 是其中最主要的遗传物质。染色体中基因的位置称作基因座,而基因所取的值又称为等位基因。基因和基因座决定了染色体的特征,也决定了生物个体(individual)的性状,如头发的颜色是黑色、棕色或者金黄色等。

以一个初始生物群体(population)为起点,经过竞争后,一部分个体被淘汰而无法再

进入这个循环圈，而另一部分则胜出成为种群。竞争过程遵循生物进化中“适者生存，优胜劣汰”的基本规律，所以都有一个竞争标准，或者生物适应环境的评价标准。适应程度高的并不一定进入种群，只是进入种群的可能性比较大；而适应程度低的个体并不一定被淘汰，只是进入种群的可能性比较小。这一重要特性保证了种群的多样性。

在生物进化中，种群经过婚配产生子代群体（简称子群）。在进化的过程中，可能会因为变异而产生具有新的生物特性的个体。基因决定了生物机体的某种特征，如头发的颜色、耳朵的形状等。综合变异的作用，子群成长为新的群体而取代旧的群体。在新的一轮循环过程中，新的群体代替旧的群体而成为循环的开始。

5.2　遗传算法

5.2.1　遗传算法的发展历史

遗传算法的研究兴起在 20 世纪 80 年代末和 90 年代初，但它的起源可追溯到 20 世纪 60 年代初期。早期的研究大多以对自然遗传系统的计算机模拟为主，其特点是侧重于对某些复杂操作的研究。虽然其中像自动博弈、生物模拟、模式识别和函数优化等给人以深刻的印象，但总的来说，这是一个无明确目标的发展时期，缺乏指导性的理论。这种现象直到 20 世纪 70 年代中期，由于美国密歇根大学霍兰（J. Holland）和迪乔恩（G. DeJong）的创造性研究成果的发表才得到改观。

1967 年，霍兰教授的学生巴格利（J. D. Bagley）在他的博士学位论文中首次提出了“遗传算法”这一术语，并讨论了遗传算法在博弈中的应用。他提出的选择、交叉和变异操作与目前遗传算法中的相应操作十分接近，尤其是他对选择做了十分有意义的研究。他认识到，在遗传进化过程的前期和后期，选择概率应适当地变动。为此，他引入了适应度定标的概念。尽管巴格利没有进行计算机模拟实验，但这些思想对于遗传算法后来的发展具有重要的意义。在同一时期，罗森伯格（C. Rosenberger）也对遗传算法进行了研究。他的研究依然是以模拟生物进化为主，但他在遗传操作方面提出了不少独特的设想。

1970 年，卡维基奥（G. Cavicchio）把遗传算法应用于模式识别中。1971 年，霍尔斯汀（R. B. Hollstien）第一个把遗传算法用于函数优化。他在论文《计算机控制系统中人工遗传自适应方法》中阐述了将遗传算法用于数字反馈控制的方法，但实际上，该论文主要讨论了对于二变量函数的优化问题，其中，对于优势基因控制、交叉和变异以及各种编码技术进行了深入的研究。

1975 年是遗传算法研究历史上十分重要的一年。这一年，霍兰出版了他的专著《自然系统和人工系统的适配》，系统地阐述了遗传算法的基本理论和方法，并提出了对遗传算法的理论研究和发展极为重要的模式理论。同年，迪乔恩完成了重要论文《遗传自适

应系统的行为分析》,把霍兰的模式理论与自己的计算实验结合起来。这是遗传算法发展中的又一个里程碑。尽管迪乔恩和霍尔斯汀一样主要侧重于函数优化的应用研究,但他将选择、交叉和变异操作进一步完善和系统化。迪乔恩的研究工作为遗传算法及其应用打下了坚实的基础,他得出的许多结论具有普遍的指导意义。

20 世纪 80 年代以后,遗传算法进入蓬勃发展时期,无论是理论研究还是应用研究都成了十分热门的课题。遗传算法广泛应用于自动控制、生产计划、图像处理、机器人等研究领域。

5.2.2 遗传算法的基本思想

遗传算法主要借鉴了生物进化中"适者生存"的规律。在遗传算法中,染色体对应的是数据或数组,通常是由一维的串结构数据来表示的。串上各个位置对应基因座,而各位置上所取的值对应等位基因。遗传算法处理的是染色体,或者称为个体。一定数量的个体组成了群体,称为种群。种群中个体的数量称为种群的大小,也叫种群的规模。各个个体对环境的适应程度叫适应度。适应度高的个体被选择进行遗传操作,产生新个体,体现了生物遗传中"适者生存"的原理。选择两个染色体进行交叉产生一组新的染色体的过程类似于生物遗传中的交配。编码的某个分量发生变化的过程类似于生物遗传中的变异。

遗传算法对于自然界中生物遗传与进化机理进行模仿,针对不同的问题设计了许多不同的编码方法来表示问题的可行解,产生了多种不同的遗传算子来模仿不同环境下的生物遗传特性。这样,由不同的编码方法和不同的遗传算子就构成了各种不同的遗传算法。但这些遗传算法都具有共同的特点,即通过对生物遗传和进化过程中选择、交叉、变异机理的模仿来完成对问题最优解的自适应搜索过程。基于这个共同的特点,古德伯格(D. E. Goldberg)提出了基本遗传算法(Simple Genetic Algorithms,SGA),只使用选择算子、交叉算子和变异算子 3 种基本遗传算子,其遗传进化操作过程简单,容易理解,给各种遗传算法提供了一个基本框架。

5.2.3 编码

遗传算法中包含 5 个基本要素:参数编码、初始群体的设定、适应度函数的设计、遗传操作设计和控制参数设定。

> 字符显示屏。在银行、医院、机场、车站等许多场所都会用到显示屏。显示屏上排列了许多灯泡。每个灯泡只有两个状态:亮与不亮。显示不同的文字或者符号其实就是让有些灯亮,让有些灯不亮。这样,显示某些

字符就归结为输出一个二进制串。例如，显示屏由 100 行、300 列灯泡组成，则一屏字符就归结为一个 30 000 位的二进制串。这就是二进制编码的基本原理。

翻花。在国庆节等许多庆祝活动中，观众可以看到天安门广场上的一幅幅巨大的标语和图案，如“庆祝国庆”“中华人民共和国万岁”等标语，以及五星红旗等美丽的图案，这就是翻花。几万个学生前后左右等间距地站在广场上，每个学生有红、黄、蓝、绿、黑、白六色翻花，每个学生拿出指定颜色的花就组成了不同的文字或者图案。这也是一种编码。与字符显示屏不同的是，翻花的每一位上的编码不只是 0 或者 1，而是数字 1～6。

遗传算法求解问题时不是直接作用于问题的解空间，而是作用于解的某种编码。因此，必须通过编码将问题的解表示成遗传空间中的染色体或者个体。它们由基因按一定结构组成。对于某个具体问题，编码方式不是唯一的。由于遗传算法的鲁棒性，对编码方式的要求一般并不苛刻，但编码方式有时对算法的性能、效率等会产生很大影响。

将问题的解编码为一维排列的染色体的方法称为一维染色体编码方法。

一维染色体编码中最常用的符号集是二值符号集{0,1}，即二进制编码。

1. 二进制编码

二进制编码用若干二进制数表示一个个体。

例如，将遗传算法应用于信号采集通道优化问题。使用二进制编码表示通道组合：0 表示未选择该通道，1 表示选择该通道，即，将 C_i 定义为通道 i 的状态，$C_i=1$ 表示选择通道 i，而 $C_i=0$ 表示不选择通道 i。$\sum_{i=1}^{16} C_i = N$ 表示在整个染色体中选择了 N 个通道。例如，选择第 2、4、6、8、9、11、12、16 这 8 个通道，可以表示为 16 位二进制编码 0101010110110001，如图 5.2 所示。通过设计这样的遗传编码，不仅给出了选择的通道数目，而且给出了选择的通道的具体分布。

通道编号	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}
染色体	0	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1

图 5.2　选择了通道 2、4、6、8、9、11、12、16 的染色体的二进制编码

二进制编码的优点如下：二进制编码类似于生物染色体的组成，从而使算法易于用生物遗传理论解释，并使得遗传操作（如交叉、变异等）很容易实现。另外，采用二进制编码时，算法处理的模式数量最多。

二进制编码的缺点如下：在求解高维优化问题时，二进制编码串将非常长，从而使得算法的搜索效率很低。相邻整数的二进制编码可能具有较大的海明距离。海明距离是

指两个数码串中对应位的值不相同的数量。例如,15和16的二进制表示为01111和10000,因此,算法要从15改进到16,就必须改变所有的位。这种缺陷造成了“海明悬崖”,将降低遗传算子的搜索效率。

2. 实数编码

为克服二进制编码的缺点,对问题的变量是实向量的情形,可以直接采用实数编码。采用实数表示法不必进行数制转换,可直接在解的表现型上进行遗传操作,从而可引入与问题领域相关的启发信息以增强算法的搜索能力。

3. 多参数级联编码

多参数优化问题的遗传算法常采用多参数级联编码。其基本思想是:对每个参数先进行二进制编码,得到子串,再把这些子串连成一个完整的染色体。多参数级联编码中的每个子串对应不同的编码参数,所以,可以有不同的串长度和参数的取值范围。

5.2.4 种群设定

由于遗传算法是对种群进行操作的,所以,必须为遗传操作准备一个由若干初始解组成的初始种群。种群设定主要包括以下两方面:初始种群的产生和种群规模的确定。

1. 初始种群的产生

遗传算法中初始种群中的个体一般是随机产生的,但最好采用如下策略设定:先随机产生一定数目的个体,然后从中挑选最好的个体加入初始种群中。这种过程不断迭代进行,直到初始种群中的个体数目达到了预先确定的规模。

2. 种群规模的确定

种群规模影响遗传优化的结果和效率。

种群规模太小,会使遗传算法的搜索空间范围有限,因而搜索有可能停止在未成熟阶段,出现未成熟收敛现象,使算法陷入局部最优解。当种群规模太小时,遗传算法的优化性能一般不会太好,这就如同动物种群太小时会出现近亲繁殖,影响种群质量一样。因此,必须保持种群中个体的多样性,即种群规模不能太小。

种群规模越大,遗传操作所处理的模式就越多,产生有意义的个体并逐步进化为最优解的机会就越大。但种群规模太大会带来若干弊病:一是遗传算法中所有的计算与操作都是由CPU完成的,所以当种群规模太大时,其适应度评估次数增加,则计算量也增加,从而影响算法效率;二是种群中个体生存下来的概率大多和适应度成正比,当种群中的个体非常多时,少量适应度很高的个体会被选择而生存下来,但大多数个体却会被淘汰,这会影响配对库的形成,从而影响交叉操作。

综合考虑以上所述的利弊,许多实际优化问题的种群规模取为20~100。

思考:为什么动物种群规模越大越好,而遗传算法中的种群规模不能太大?

5.2.5　适应度函数

遗传算法遵循自然界优胜劣汰的原则，在进化搜索中基本上不用外部信息，而是用适应度值表示个体的优劣，作为遗传操作的依据。例如，前面提到的羚羊群的优秀的评价标准就是力气大和跑得快。个体的适应度高，则被选择的概率就高，反之就低。改变种群内部结构的操作都是通过适应度值加以控制的。可见，适应度函数的设计是非常重要的。

适应度值是对解的质量的一种度量。适应度函数(fitness function)是用来区分种群中的个体优劣的标准，是进化过程中进行选择的唯一依据。在具体应用中，适应度函数的设计要结合待求解问题本身的要求而定。

1. 将目标函数映射为适应度函数

一般而言，适应度函数是由目标函数变换而来的，但要保证适应度函数是最大化问题和非负性。最直观的方法是直接将最优化问题的目标函数作为适应度函数。

若目标函数为最大化问题，则适应度函数可以取为

$$\mathrm{Fit}(f(x)) = f(x) \tag{5.1}$$

若目标函数为最小化问题，则适应度函数可以取为

$$\mathrm{Fit}(f(x)) = \frac{1}{f(x)} \tag{5.2}$$

由于在遗传算法中要对个体的适应度值进行排序，并在此基础上计算选择概率，所以适应度函数的值要取非负值。但在许多优化问题求解中，不能保证所有的目标函数值都有非负值。因此，在不少场合，当采用问题的目标函数作为个体的适应性度量时，必须将目标函数转换为求最大值的形式，而且要保证函数值必须非负。

2. 适应度函数的尺度变换

有一份试卷的题目存在一些问题，没有拉开学生的成绩。例如，学生成绩都集中在 75～85 分，这样会影响学生的学习积极性。面对这种情况，为了调动学生的积极性，有的老师对考试成绩进行变换，例如将 75 分变换到 65 分，而将 85 分变换到 95 分，这样学生成绩差距加大。遗传算法的适应度函数的尺度变换的思想与此类似。

在遗传算法中，将所有妨碍适应度值高的个体产生，从而影响遗传算法正常工作的问题统称为欺骗问题(deceptive problem)。

在设计遗传算法时，种群的规模一般为几十至几百，与实际物种的规模相差很远。因此，个体繁殖数量的调节在遗传操作中就显得比较重要。如果种群中出现了超级个体，即该个体的适应度值大大超过种群的平均适应度值，则按照适应度值进行选择时，该

个体很快就会在种群中占有很大的比例,从而导致算法较早地收敛到一个局部最优点,这种现象称为过早收敛。这是一种欺骗问题。在这种情况下,应该缩小该个体的适应度值,以降低其竞争力。另外,在搜索过程的后期,虽然种群中存在足够的多样性,但种群的平均适应度值可能会接近种群的最优适应度值。在这种情况下,种群中实际上已不存在竞争,从而搜索目标也难以得到改善,出现了停滞现象。这也是一种欺骗问题。在这种情况下,应该改变原始适应度值的比例关系,以提高个体之间的竞争力。

对适应度函数值域的某种映射变换称为适应度函数的尺度变换(scaling)或者定标。

最常用的尺度变换是线性变换,即变换后的适应度函数 f' 与原适应度函数 f 为线性关系:

$$f' = af + b \tag{5.3}$$

式中,对系数 a 和 b 可以有多种途径进行设定。例如,可由式(5.4)确定:

$$a = \frac{f_{\text{avg}}}{f_{\text{avg}} - f_{\min}}, \quad b = \frac{-f_{\min} f_{\text{avg}}}{f_{\text{avg}} - f_{\min}} \tag{5.4}$$

式中,f_{avg} 为原适应度的平均值;$f_{\min}$ 为原适应度的最小值。

线性变换法改变了适应度值之间的差距,保持了种群的多样性,计算简便,易于实现。

5.2.6 选择

遗传算子是模拟生物基因遗传的操作,从而实现优胜劣汰的进化过程。遗传算法中主要包括 3 个基本遗传算子:选择(selection)、交叉(crossover)和变异(mutation)。

选择操作也称为复制(reproduction)操作,是从当前种群中按照一定概率选出优良的个体,使它们有机会作为父代繁殖下一代。判断个体优劣的准则是各个个体的适应度值。显然这一操作借鉴了达尔文“适者生存”的进化原则,即个体适应度值越大,其被选择的机会就越大。选择操作的实现方法有很多。优胜劣汰的选择机制使得适应度值大的解有较高的存活概率,这是遗传算法与一般搜索算法的主要区别之一。

不同的选择策略对算法的性能也有较大的影响。这里介绍几种常用的选择方法。

1. 个体选择概率分配方法

在遗传算法中,选择哪个个体进行交叉和变异是按照概率进行的。适应度值大的个体被选择的概率大,但不是说一定能够选上;适应度值小的个体被选择的概率小,但也可能被选上。所以,首先要根据个体的适应度值确定其被选择的概率。个体选择概率分配方法有以下两种。

1) 适应度比例模型

适应度比例模型(fitness proportional model)也叫蒙特卡洛法,它是目前遗传算法中最基本也是最常用的选择方法。在该方法中,每个个体被选择的概率与其适应度值成正

比。设群体规模大小为 M，个体 i 的适应度值为 f_i，则这个个体被选择的概率为

$$p_i = \frac{f_i}{\sum_{i=1}^{M} f_i} \tag{5.5}$$

2）排序模型

排序模型(rank-based model)首先计算每个个体的适应度值，根据适应度值大小对种群中的个体进行排序，然后把事先设计好的概率按排序分配给个体，作为各自的选择概率。选择概率仅仅取决于个体在种群中的序位，而不是实际的适应度值。排在前面的个体被选择的机会较大。

它的优点是克服了适应度比例模型的过早收敛和停滞现象等问题，而且对于极大值或极小值问题不需要进行适应度值的标准化和调节，可以直接使用原始适应度值进行排序和选择。排序模型比适应度比例模型具有更好的鲁棒性，是一种比较好的选择方法。

最常用的排序模型是线性排序模型。该模型是由贝克(J. E. Baker)提出的。首先假设种群成员按适应度值从大到小依次排列为 $x_1, x_2, \cdots, x_M$，然后根据一个线性函数给第 i 个个体 x_i 分配选择概率 p_i：

$$p_i = \frac{a - bi}{M(M+1)} \tag{5.6}$$

式中，a、b 是常数，这两个系数是为了保证所有个体的概率之和等于 1，以满足概率的数学定义。

2. 选择个体方法

选择操作是根据个体的选择概率确定哪些个体被选择进行交叉、变异等操作。基本的选择方法有以下 4 种。

1）轮盘选择方法

轮盘选择(roulette wheel selection)方法在遗传算法中使用得最多。在轮盘选择方法中，先按个体的选择概率产生一个轮盘，轮盘每个扇形区域的角度与个体的选择概率成正比；然后产生一个随机数，它落入轮盘的某个区域，就选择相应的个体交叉。显然，选择概率大的个体被选中的可能性大，获得交叉的机会就大。

在实际计算时，可以按照个体顺序求出每个个体的累积概率；然后产生一个随机数，它落入累积概率的某个区域，就选择相应的个体交叉。例如，表 5.1 给出了 11 个个体的适应度值、选择概率和累积概率。为了选择交叉个体，需要进行多轮选择。例如，第 1 轮产生的随机数为 0.81，落在第 5 个和第 6 个个体之间，则第 6 个个体被选中；第 2 轮产生的随机数为 0.32，落在第 1 个和第 2 个个体之间，则第 2 个个体被选中。

表 5.1 11个个体的适应度值、选择概率和累积概率

个体	1	2	3	4	5	6	7	8	9	10	11
适应度值	2.0	1.8	1.6	1.4	1.2	1.0	0.8	0.6	0.4	0.2	0.1
选择概率	0.18	0.16	0.14	0.13	0.11	0.09	0.07	0.05	0.04	0.02	0.01
累积概率	0.18	0.34	0.48	0.61	0.72	0.81	0.88	0.93	0.97	0.99	1.00

2) 锦标赛选择模型

锦标赛选择模型(tournament selection model)从种群中随机选择 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,直到个体数量达到预先设定的数量为止。参数 k 称为竞赛规模。

锦标赛选择模型的优点是克服了在种群规模很大时其额外计算量(如计算总体适应度值或排序)很大的问题。它常常比轮盘选择得到更多样化的种群。

显然,这种方法也使得适应度值大的个体具有较大的生存机会。同时,由于它只使用适应度值的排序作为选择的标准,而与适应度值的大小不成正比,从而也能避免超级个体的影响,一定程度上避免了过早收敛和停滞现象的发生。

3) 随机竞争方法

作为锦标赛选择方法的一种特殊情况,随机竞争(stochastic tournament)的过程是:每次首先用轮盘选择方法选取一对个体,然后让这两个个体进行竞争,适应度值大者获胜。如此反复,直到选满为止。

4) 最佳个体保存方法

以前世界杯足球赛有一个规则:上届冠军球队直接进入决赛阶段比赛。其实,这就是遗传算法中的最佳个体保存方法。这个规则直到2002年韩日世界杯结束后才被取消。在2006年的德国世界杯上,上届冠军巴西队也参加了南美区预选赛。

前面几种选择个体的方法都存在的一个缺点是可能会把已经产生的最好的个体丢掉。为了克服这一缺点,可以采用最佳个体保存方法,即种群中适应度值最大的个体不经交叉而直接复制到下一代,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值的个体。使用这种方法能够明显提高遗传算法的收敛速度。

5.2.7 交叉

遗传算法中起核心作用的是交叉算子。交叉也称为重组(recombination)。通过交叉能够使父代将特征遗传给子代。子代应能够部分或者全部地继承父代的结构特征和