第一章　遗传算法的基本概念

1.1 绪论

遗传算法(Genetic Algorithms)是一类术语的总称,是一种以进化或遗传机制为关键元素的计算机模型求解程式、方式。它包括：遗传算法(GA)、进化编程(Evolutionary Programming)、进化策略(Evolutionary Strategies)、分类器系统(Clalssifer Systems)和遗传编程(Genetic Programming)。随着 50 年代仿生学的发展，许多科学家从生物学中得到启发，按照生物进化的机理，发展出适合于现实世界复杂问题优化的模拟进化算法。遗传算法是模拟生物进化原理的过程。像进化过程一样，它作用于代表着给定问题答案的群体上，通过存在的优良个体的组合，寻求更好的个体。遗传算法是借用生物界术语衍生而来的。它是一种迭代、自适应、启发式的随机搜索算法，是基于自然选择和自然遗传的全局收敛算法。它通过遗传基因代码，利用再生、交叉、变异等基本算子，在点群中进行全局寻优。

遗传算法是对自然进化论的类比与数学抽象。尽管目前人们对生物进化的某些机制(超自然力量支配)还不完全清楚,但大都承认，自然进化是一种基于群体的优化过程。物种在环境中优胜劣汰，适者生存，最终留下优秀的个体。遗传算法正是模拟自然进化中的自然选择、基因重组和基因变异来进行自学和寻优。算法中代表问题答案的个体（染色体）通常也是二进制编码组成，但实际上表达形式

需要全本请访问 www.ertongbook.com

可多种多样,例如自然数编码、实数(浮点数)编码、队栈符号编码、结构形式编码，乃至矩阵、树形编码等。因为这一表达形式的多样性，使得其应用具有广泛的适应性，也说明了遗传算法并非简单、通常意义上的算法，而是一种新的方法、思想体系，是求解复杂问题的有效工具和通用的优化技术。

1.2 遗传算法的发展历史

遗传算法萌生于 60 年代,是由美国 Michigan 大学的 J.Holland 教授发起的。他被认为是 GA 之父，至今也是遗传算法发展史上最具代表性的人物之一。起初研究自然和人工系统，他和他的学生在研究中，试图发展一种用于创造通用程序和机器学习的理论系统，使其具有适应环境的能力。1962 年他发表了关于适应系统理论方面的系列文章，开始使用群体方法搜索以及选择（Selection）、变异（Mutation）等遗传操作。由于当时基于语言智能和逻辑—数学智能的传统人工智能非常盛行，因而他的基于自然进化的思想遭到怀疑和反对。

J.D.Bagley 于 1967 年使用‘遗传算法(GA)’这一术语，在其博士论文中发表了 GA 在博弈应用方面的文章。他采用双倍体编码，发展了与目前类似的复制（Reproduction）、交换(Crossover)、变异(Mutation) 显性、倒位等基因操作，提出了防止早熟收敛的问题并发展了自组织遗传算法的概念。R.S.Rosenberg1967 年则侧重于生物领域，在计算机上实现单个细胞的模拟。 60 年代可以说是遗传算法的萌生期。

Cavicchio 1970 年研究了基于遗传算法（GA）的子程序选择和模式识别问题，采用整数编码，提出了以预选择策略来保证群体的多样性的观点。

大约在 1972 年,Holland 提出了重要的模式定理。这是遗传算法重要的数学理论基础之一。他首次采用二进制编码研究函数优化

问题。在编码问题上，自此也出现了两派：一派是以数值优化计算的方法和精度为准，采用一个基因一个参数的方式，并把相应的基因操作改造成适合实数操作的形式；另一派是根据模式定理，建议用尽量少的符号编码。值得一提的是美国科学家 Koza 将程序树形编码用于进化编程，开创了编码形式深层化的先例及新的 GA 应用途径——遗传编程。

1973 年是遗传算法发展史上的里程碑，一是 Holland 出版了经典著作《Adaption in Nature and Artificial Systems》。它凝结了作者几十年的许多思想及实践的结晶，详细阐述了遗传算法的理论，使其具备严格的数学基础，发展了一整套模拟生物自适应系统的理论；二是 Dejong 的博士论文《 An Analysis of the Behaviour of a Class of Genetic Adapive Systems》。它深入研究了模式定理，并做了大量严格的计算实验，给出了明确的结论。他的工作为后继的广泛应用奠定了理论基础。可以说 70 年代是遗传算法的形成期。

80年代是遗传算法大发展的时期。由于这一阶段，传统的人工智能陷入困境，人们开始从生物底层模拟智能。神经网络、机器学习和遗传算法重新繁荣。 Goldberg 在 1983 年的博士论文中第一次将 GA 用于实际工程系统——煤气管道系统优化。该系统模拟了从西南向东北输送天然气的管道系统（该问题没有分析解）。 1984 年，Fitzpatrick，Grefcnstette 和 Van Gucht 利用遗传算法处理了医学图像变换问题。Axelrod 和 Forrest 研究了社会科学中应用遗传算法的问题——囚犯困境问题，等等。从此，遗传算法的理论研究更为深入和丰富，应用研究也更为广泛和完善。自 1985 年，遗传算法及应用国际学术会议两年召开一次。在一些机器学习、人工智能、神经网络、运筹学为主题的国际会议上，也设有 GA 分组会议。这标志着遗传算法开始蓬勃发展，成为国际学术界的一个热门课题。

进入 90 年代，遗传算法发展进入高速发展的时期。应用研究呈

异彩纷呈的形式展开。 Goldberg 的《Genetic Algorithms in Search, Optimization, and Machine Learning 》为遗传算法研究奠定了坚实的科学基础。1991 年 Lawrence Davis 出版了《Handbook of Genetic Algorithms》。该书对 GA 应用具有重要指导作用。从发表的文章来看，国外约 5000 篇,国内约 200 篇。现在其应用领域几乎涉及到所有学科和工程领域，例如文学艺术(GA 作曲)、工程设计、数学、物理、化学、计算机科学、生物、医学、社会科学及军事等体系。1991 年 John. Koza 获得美国关于遗传算法软件的专利。几家商业机构已推出遗传算法方面的集成软件,其一称为 EOS(Evolutionary object systems),基于 C + + 语言开发成面向目标的软件，具有丰富的单元并可集成为遗传算法应用软件包，为用户提供扩展工具。

因为人类的大部分改造现实世界的过程实际上是一种优化过程。遗传算法正是这样一种合适的方法。同时现代以不确定性、非线性、时间不可逆性为内涵，以复杂问题为对象的复杂系统，寻求其精确的最优解其实是不现实的，故而转向求近优解。遗传算法也正是最佳的工具之一。因此遗传算法的发展繁荣不是偶然的，是由其本质和潜在的优越性决定的。

1.3 遗传算法的基本概念

1.3.1 遗传算法的概念

遗传算法是一种基于自然基因选择机制的搜索算法。它同时使用适应度好的个体结构和伪随机信息交换原理，形成一种具有某种类似人类进化特性的问题求解算法。

——它不是直接操作问题本身的参数，而是需要原参数的一种编码。

——遗传算法不是依靠单个体而是依靠点群体工作。

——遗传算法依靠一个简单的评价准则去指导搜索，不需要响

应面的知识特别是梯度信息，因而限制较少。

——遗传算法使用随机搜索，但不是随机算法，而是一种启发式、自学习机制的智能算法。

这些特性决定了遗传算法具有高度鲁棒性及有效性（隐并行性）。当传统方法无效时，它显示了较好的优越性。我们注意到，生物进化并不是一种带有目的和直接的过程。没有证据表明进化的目标只产生人类。自然的过程产生了形形色色的生物种群。生物的进化由自然选择或由不同个体在环境中的竞争决定。一些个体比其他要好，这些好的个体，更可能生存和传播遗传物质。自然界中，遗传信息的编码是有性复制完成的。有性复制导致子代与其父代有相似的基因，因为子代携带了父代的遗传信息。而两性结合再生是依据其适应自然环境的能力决定的。这种能力刻画为适应度（Fitness）。对进化产生作用的通常认为是杂交（Crossover）与基于适应度的选择（Selection）、再生（Reproduction）。为保证群体的多样性，变异是不可缺少的遗传操作。

按照进化论的观点，显著有意义的进化改变需要很长的时间跨度，以至于用地质年代的时间跨度计算。因此有人怀疑遗传算法不是一个良好的计算方法。因为进化论表明，在数千年的过程中，进化是没有实质性的变化的，那么在计算机上用数秒、数小时或者数天的计算时间按照进化原理发展的算法，是不太可能得到有意义的结果的。但进化的缓慢变化是因为地质变化的缓慢而不是遗传变化的缓慢。这一点在实验室已得到证明。微组织机体可以在几天（几小时）或几周的时间内引起显著的基因变化，其关键在于组织器官简单且可以很快地复制再生。在微观上看，现实世界中问题的个体表达往往比生物染色体简单得多。

我们希望用 GA 解决的大多数问题可以表达成长度为 10 比特到几千比特的比特串。这与人类所含 23 对染色体，每一对染色体包

需要全本请访问 www.ertongbook.com

含约 10^8 基本对相比的确简单得多。此外，我们的计算机速度发展得越来越快。这些人工比特串的组织可以以每秒百万乃至上亿次的速度杂交。因此在一个很短的时间，如 10 分钟,可以有 6 亿个个体消亡或产生。假定对大小 100 的比特串，其搜索空间将超过 $2^{100} = 10^{30}$。即使使用每秒 10 亿次搜索，要穷尽这样的空间也将花上 300 亿年。然而，进化和遗传算法的成功在于它不是搜索全部空间，而是高效地搜索很少的部分的可能的空间。它通过适者选择杂交和随机变异等遗传操作,很快地收敛到极其优秀的个体上(即近优解)。

遗传算法在很大程度上使用随机数。然而，遗传算法并非只是随机搜索程式。遗传算法应用很高的概率产生一代比一代好的解；相反地，一个纯随机搜索，例如模拟退火算法，其成功地找到好解的概率很低，而且极易陷入一点局部解，而很难找到更好的解。

只有对那些已知的传统技术方法无法解决而遗传算法可以成功地解决的问题时，遗传算法才显示出其有效性。一种所谓的“强壮性”和“适应性”特征使得遗传算法能很好地应用到广泛的问题中。这些特征是其他算法技术所缺少的（神经网络可能例外）。什么样的问题需要这种力量？其中之一是人工智能。智能学习及创造是 AI 研究者自 1990 年以来一直要求计算机去完成的。然而，这些只在较小的范围内取得成功，而一种强壮的推理计算机仍然是一种科幻目标。

1.3.2 遗传算法的分类

GA 的研究范围大体分为：GA(Genetic Algorithms), EP (Evolutionary Progamming), ES (Evolutionary Strategy), CS (Classifier Systems), GP (Genetic Programming)等。进化(规划)编程,最初由 Lawrence.J.Fogel 于 1960 年设计,是一种类似于 ES、同 GA 一样的随机优化策略技术。它不强调父代与子代间的相似行为，而是寻求仿效自然界观察到的特殊遗传操作——变异操作。它没有像 GA 一样对解表达上的限

制。1962 年 Fogel, Dwens and Walsh 的《Artificial Intelligence Through Simulated Evolution》一书是 EP 应用的重要里程碑。在本书中，有限状态自动机被用来预测由 Markov 过程及非平稳时间序列产生的符号串，发展了适应行为的智能结构。该方面吸收了各方面的学术组织如科学、商业、军事等，致力于 EP 的理论与应用研究。

通常 EP 算法的伪指令如下：

```
//start with an initial time
t: = 0;
//initialize a usually random population of individuals initpopulation p (t);
//evaluate fitness of all initial individuals of population evaluate p(t);
//test for termination criterion(time, fitness, etc) while not done do
//perturb the whole population stochastically
p'(t): = mutute p(t);
//evaluate it's new fitness
evaluate P'(t);
//stochasticalluy select the survivors from actual firness p(t + 1) = survive p(t), p'(t)
//increase the time counter
t: = t + 1;
End.
```

分类器系统(CS)是 Holland 等人建立的。它由一些规则组成，通过自适应来学习控制和解释一些外部环境的行为。分类器系统中，有三个互相联系的子系统：规则和信息部分、规则评价部分以及遗传算法部分。基本上分类是执行下面的操作循环：

(1)将外部环境中输入的信息处理成固定长度的位串，并放入到一个位串表中。

(2)将表中所有的位串与每个分类的条件部分进行比较，并把行为部分的匹配结果视为一个新的位串表中可能的单元。

(3) 通过规则评价和遗传算法作用后，产生一个新的位串表，并替换旧的位串表。这个新表中的位串被处理成供外部环境的输出信号。

规则评价部分负责判断哪些添加的分类将形成新的位串表。这个判断是基于分类强度——一个指定给每个分类的有效性动态度量。越好的分类具有越大的强度，且相对于较差的分类，规则评价系统优先选择好的分类的行为部分。一个分类被选择的机会越多，它的强度也会变得越大。

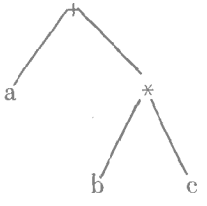
遗传算法部分负责生成新的分类。较前的分类被那些新的分类所代替。

其伪代码如下(带学习的 CS)

```
//start with an initial time
t: = 0;
//an initially empty message list init message list ML(t)
//randomly generate popuation of classifiers init Classifier Populations p(t);
//test for cycle termination criterion (time,fitnesss etc).
while not done do
//increase the time counter
t: = t + 1;
//1.detectors check whether input message are present.
ML: = read detectors(t);
//2.compare ML to the classifiers and save matches
ML': = match classifiers ML,p(t);
//3.highest bidding classifier(s) collected in/
```

```
//ML' wins the "race" and post the(ir) message(s)
ML': = select matching classifiers ML', p(t);
//4. tax bidding classifiers, reduce their strength
ML': = taxposting classifiers ML', p(t);
//5. Effectors check new Message list for output Msgs
ML: = send Effectors ML'(t);
//6. receive payoff from environment(Reinforcement) c: = receivepay-
off(t);
//7. distribute payoff/credit to classifiers
p': = distribute Credit c, p(t);
//8. Eventually (depending out), an EA(usually a GA) is applied to
the classifier population if criterion then
p: = generate new Rules p'(t);
else
p: = p'
End.
```

遗传编程(GP)是遗传学习模型到程序空间的扩展，即构成群体的个体不是固定长度的字符串，而是程序本身。它以树状形式编码。例如：简单的程序 "a + b * c" 可以表达成：



由于 LISP 语言非常适合该类问题的编程，因而 GP 通常使川

LISP 实现。但这并非必须。值得指出的是 GP 通常不使用变异为遗传操作，而只使用交叉。本书重点论述遗传算法理论及其应用。其他几种类型不再作进一步介绍。读者可以参阅相关资料和文献。关于 GA 目前发展起来的也有几种形式，通常称为简单标准遗传算法 SGA(Simple Genetic Algorithm) 和修改的高级遗传算法 modifiedGA、并行 GA 以及混和 GA 等类型。

对研究和使用遗传算法而言，有五个基本要素是值得掌握的：

①参数编码；对待求解问题参数进行适当编码。

②初始群体确定 因为 GA 是群体规模上以个体为对象操作的，这就需要确定一个适宜的群体。

③适应度函数的确定；用以评价群体中个体的好坏，用于指导进化。

④遗传操作设计；通常根据编码形式设计，主要有选择、交叉和变异三个基本操作算子。

⑤进程控制参数设定；主要是群体大小及算子使用概率以及终止条件的设定等。

具有如下特点的 GA 叫作简单标准遗传算法 SGA：

①采用轮盘赌选择；②随机配对；③一点交叉并生成两个子个体；④群体内允许有相同个体。

根据不同问题的性质，对上述要素内容的修改设计可以构造出各种修正遗传算法。

一种修正遗传算法的伪指令如下：

```
Procedure mod GA
Begin
t←0
initialize p(t),
evaluate p(t),
```

```
while(not termination - condition) do
begin
t←t + 1
select - parents from p(t - 1),
select - dead from p(t - 1),
from p(t):reproduce the parents
evaluate p(t)
end
```

另一种修正 GA,称为 CM - GA,压缩映像 GA。其伪指令如下:

```
Procedure CM - GA Begin
t = 0;
initializc p(t)
evaluate p(t)
while (not termination - Condition) do begin contractive mapping f[p
(t)→p(t + 1)]
t = t + 1
select p(t) from p(t - 1)
recombine p(t)
evaluate p(t)
if Eval[p(t - 1)] = Eval[p(t)]
then t = t - 1;
end
```

修正遗传算法 mod GA 不是从 p(t - 1)中选择 p(t),而是独立地选择 r 个个体再生、r 个染色体死亡,从而形成新的 p(t)。压缩映像遗传算法(Contractive Mapping GA)——CM - GA,其主要的一点是当产生的一代群体与上一代相比,没有改进时就不计迭代次数,而且新选择组合形成新一代;另一种是群体大小规模变化的 GA,称为 GAV

需要全本请访问 www.ertongbook.com

a PS(Genetic Algorithm with varying Population Size)。它是用‘老化’的概念取代选择机制。其程序是：

```
begin
t = 0
initialize p(t)
evaluate P(t)
while (not termination - condition) do
begin
t = t + 1
increase the age of each individual by 1
recombine p(t)
evaluate p(t)
remove from p(t) all individuals with age
great than lifetime
end
```

还有其他的一些 mod GA。

例如 最佳保留选择的 GA(elitist selection)

期望值模型；(expected value model)；

最佳保留期望值模型；(elitist expected value model)；

排挤因素模型；(crowding factor model)；

一般交叉模型(generalized crossover model)等。

混合遗传算法(Hybrid GA)是与其他启发式搜索算法或传统算法结合构成的新算法。它可以改善 GA 的局部搜索能力，弥补单一算法的不足，提高优化质量与效率。GA 常与模拟退火法 SA(simulated annealing)、列表寻优法 TABU 以及单纯形法等传统方法相结合形成混合 GA 以有效地求解各种实际问题。

1.3.3 遗传算法的常用术语及参数

遗传算法作为模拟生物进化的专门方法，它的一些基本术语常来自遗传学。这些术语对理解和正确使用 GA 是非常必要的。常用的术语和参数如下：

——群体(population)：个体(individual)的集合，代表问题的解空间子集。

——串(string)及串空间：串是个体的表达形式，对应着遗传学中的染色体，通常为二进制串，也可为整数串、实数串、表串、结构树串等。

——群体大小(population size)群体集合中个体的数量。

——串基因(gene)串中元素，表示不同的特征，其值称为等位基因(alletes)。

——例如串 S=01101 元素 0、1、1、0、1 分别为串基因。

——基因位(gene position)：某一基因在串中的位置。从左到右计算，它对应于遗传学中的地点(locus)。

——基因特征值(gene feature)若用串表示整数，S 的第一个基因表示的值是 $8(2^3)$。

——串结构空间(S^s)串基因任意组合构成的串的集合，对应于遗传学中基因型(Genetype)的集合。

——参数空间(S^p)串空间在物理系统中的映射，对应于遗传学中表现型(phenotype)的集合。

——非线性：对应于遗传学中的异位显性(Epistasis)。

——模式(schema)：表示一个基因串中某些特定位相同的结构。模式由字符{0,1,*}表示。它是 GA 中非常重要的概念。

——适应度(fitness)：表示个体对环境的适应程度，往往也是评价个体好坏的尺度。

——交叉(crossover)：一对个体的基因交换，有单点交叉和多点交叉等形式。它体现染色体间的信息与基因交换的思想，也是 GA

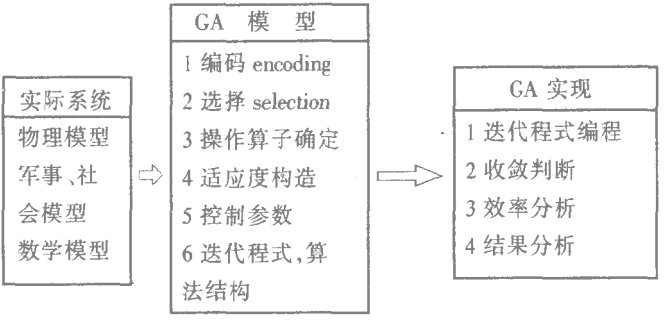
有效性的主要支持点。

——变异（mutation）：染色体中某些基因的改变，反映了生物在进化中因偶然因素引起的突变。

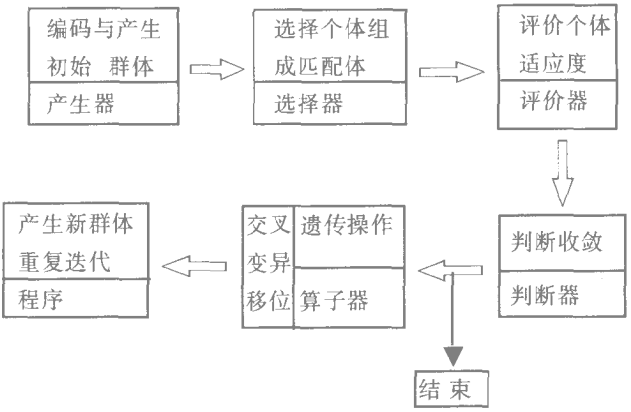
第二章　遗传算法的基本理论

2.1 遗传算法的构模与实现

对一个现实问题要想用遗传算法去解决，必须完成遗传算法的构模与遗传算法的实现两步。遗传算法构模的目的就是将实际问题抽象成遗传算法模型，然后编程加以解决。我们可以用如下的框图来表示 GA 的构模与实现过程：



从遗传算法的实现程序看，步骤不太复杂，但具体实现与操作起来却并非容易，需要多方面的知识和构模艺术，也包括实验反馈修改。这是一项创造性很强的工作。在实际进程上，其流程为：



2.1.1 编码

编码是问题表达（representation）的主要手段。它是将求解空间的解的表示映射成遗传空间解的表示。对编码的基本要求是两个空间的解需一一对应，且编码尽量简明。Holland 建议采用二进制编码。有的学者例如 Goldberg. Smith 等人使用双倍体表达（双倍体表达是高等生物染色体的主要特性，有长期记忆等作用）。

由于浮点数编码有精度高、便于大空间搜索以及比较直接等优点，浮点编码越来越受到重视。同样，自然数编码对某些问题像旅行员问题非常有效。值得指出的是 GA 不仅可以对数值符号集组成的字符串进行操作，还能够直接对结构对象进行操作。所谓结构对象泛指集合、序列、矩阵、树结构、图和表等各种结构形式为对象。有的还以物理模型乃至五线谱等更一般的形式编码。Koza 则发展了用计算机程序来编码的思路。GA 的这些特点决定了其广泛的应用性。例如，矩阵编码，GA 可以实现对神经网络或自动机的结构或系统优化 集合编码，GA 可以实现规则集合或知识库的精练而达到高质量的机器学习，这在人工智能、决策支持系统中极为有用 树结构编码，GA 可以得到用于分类的最佳决策树；任务序列编码，GA 可以用于任

务规划，特别是军事任务规划的快速完成；程序编码，GA 可以完成自动编程，带来计算机编程方面的新思路。

Vose，Battle 和 Liepins 等人扩展了 Holland 的模式定理，揭示了不同编码之间的同构性。由于编码是 GA 应用中的首要问题，因此建立完善的 GA 编码理论是必要的。现在的做法基本上是 GA 编码依据求解问题的性质而变。能否找到通用的编码技术有待于进一步研究。

编码技术：

1）一维染色体编码。

一维染色体编码是指搜索空间的系数转换到遗传空间后，其相应的基因是一维排列构成染色体。一维染色体编码中最常用的符号集是二值符号集 {0,1}。二值编码是目前 GA 中最常用的编码原则。它便于用模式定理分析。但二值编码具有一定的映射误差，特别是它不能直接反映出所求问题的本身结构特征。目前一维编码已有多种形式的改进，例如 实数表示、Gray 码表示和表表示等。二值编码的一般方法：①根据所需精度确定参数的串长；②解码，由二值串转成实数。例如：函数优化问题

$$f(x) = x \cdot \sin(10\pi x) + 1.0, x \in [-1,2]$$

若要求精度为 10^{-6}，则 $[-1,2]$ 至少要分为 $3 \cdot 10^6$ 个等区间。这意味着需要 22 比特串描述 $x(2097152 = 2^{21} < 3000000 < 2^{22} = 4194304)$，而二值串 $< b_{21}, b_{20}, \cdots b_1 b_0 >$ 到实数 x 的转换为：

$$x = -1.0 + \left(\sum_{i=0}^{21} b_i \cdot 2^i\right) \cdot \frac{3}{2^{22} - 1}$$

例如：染色体(1000101110110101000111)对应值 $x = 0.637197$。此种转换不难用计算机程度来实现。

2）多参数映射编码。

在一些多参数求解问题中，需要将这些多参数编码。基本的思

需要全本请访问 www.ertongbook.com