计算机理论基础与应用丛书

粒子群算法及应用

纪 震 廖惠连 吴青华 /著

科学出版社
www.sciencep.com

计算机理论基础与应用丛书

粒子群算法及应用

纪　震　廖惠连　吴青华　著

科学出版社
北京

内 容 简 介

粒子群算法是一种新的模仿鸟类群体行为的智能优化算法，现已成为进化算法的一个新的重要分支。全书共分为八章，分别论述了基本粒子群算法和改进粒子群算法的原理，并且详细介绍了粒子群算法在函数优化、图像压缩和基因聚类中的应用，最后给出了粒子群算法的应用综述和相关程序代码。

本书可以作为计算机科学与技术、控制科学与工程等信息类学科的研究生教材，也可供有关科研人员和工程技术人员参考。

图书在版编目(CIP)数据

粒子群算法及应用/纪震，廖惠连，吴青华著. —北京：科学出版社，2009
(计算机理论基础与应用丛书)
ISBN 978-7-03-023284-7

Ⅰ. 粒… Ⅱ. ①纪…②廖…③吴… Ⅲ. 电子计算机-算法理论-研究
Ⅳ. TP301.6
中国版本图书馆 CIP 数据核字(2008)第 168966 号

责任编辑：张海娜 / 责任校对：陈玉凤
责任印制：赵 博 / 封面设计：王 浩

科学出版社 出版
北京东黄城根北街 16 号
邮政编码：100717
http://www.sciencep.com

新蕾印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2009 年 1 月第 一 版 开本：B5(720×1000)
2009 年 1 月第一次印刷 印张：16 1/4
印数：1—3 000 字数：314 000

定价：48.00 元

(如有印装质量问题，我社负责调换〈新蕾〉)

前　言

粒子群算法是由社会心理学家 Kennedy 和 Eberhart 博士在 1995 年共同提出的一种新的模仿鸟类群体行为的智能优化算法，现已成为进化算法的一个新的重要分支。该算法通过初始一群随机粒子（每个粒子代表着一个潜在的解），并利用迭代方式，使每个粒子向自身找到的最好位置和群体中最好粒子靠近，从而搜索最优解。

由于粒子群算法思想直观、实现简单而且具有很高的执行效率，自从提出以来，一直受到国外相关领域的众多学者关注。IEEE Congress on Evolutionary Computation 国际年会上，粒子群算法被作为一个独立的分支，与遗传算法、进化规划等进化算法相提并论。近十年内粒子群算法被广泛地应用于优化计算、神经网络以及人工智能等多个领域。特别是进入 21 世纪以来，*IEEE Transactions on Evolutionary Computation*（IEEE）、*Evolutionary Computation*（MIT Press）、*Natural Computing*（Springer）等杂志发表的关于粒子群算法的论文数量急剧增加，许多改进型粒子群算法被提出，并应用于更为广泛的领域之中。近年来，国内对粒子群算法的研究也取得突飞猛进的成果，论文数量大幅度增加，众多学者尝试将粒子群算法与其他算法结合，在部分应用领域中取得了良好的效果。与此同时，也有许多学者致力于将粒子群算法应用于硬件实现中。

作者多年来致力于研究粒子群算法理论及其应用，在此基础上撰写了此书。本书吸纳了国内外许多具有代表性的研究成果，内容取材新颖、覆盖面广、深入浅出，注重理论联系实践，力图体现国内外在这一领域的最新研究进展。本书可作为计算机科学、控制科学等专业高年级本科生、研究生和教师的参考书，也可供从事智能优化的科技工作者阅读和参考。

全书共 8 章内容：第 1 章是绪论，主要介绍了粒子群算法的思想起源和研究现状；第 2 章阐述了基本粒子群算法的原理机制、数学模型及具体实现，并与其他进化算法进行比较异同，最后分析了其复杂度；第 3 章对粒子群算法进行了深入的分析，考查了其代数分析、解析分析以及差分方程分析，并研究了其收敛性问题；第 4 章给出了许多改进型的粒子群算法，包括作者提出的智能单粒子算法；第 5 章详细介绍了智能单粒子算法在函数优化中的应用；第 6、7 章给出了作者提出的粒子对算法，并介绍其在图像压缩和基因聚类中的应用；第 8 章概括了粒子群算法在其他领域的应用，并给出了几个应用实例。附录给出了作者提出的智能单粒子优化算法求解函数的源代码、粒子对算法应用于图像矢量量化的源代码、基准基

准函数的源代码和基因聚类常用软件。

感谢国家自然科学基金项目(60172065)、国家自然科学基金委员会与英国皇家协会中英合作项目(60711130233)和深圳大学出版基金的资助，还要感谢美国德州仪器公司，以及每天工作在深圳大学-德州仪器 DSP 实验室中、能够对最新技术保持研究激情的研究人员。感谢实验室的杜智华博士、王宜伟和周家锐为本书的第 5 章和第 7 章提供了工作基础，储颖、姜来和薛丽萍等多位老师提供了许多有益的思路。特别感谢张基宏教授，他率先将智能优化的思想应用于矢量量化的研究领域，为本书的许多工作奠定了基础。

目前国内粒子群算法的书籍和资料依旧十分匮乏，希望本书可以给大家一定的帮助，从理论和实践中进一步了解粒子群算法。但由于作者水平有限，书中难免有疏漏之处，恳请诸位专家、学者及广大读者不吝指正，共勉之。

作　者

2008 年 8 月于

深圳大学-德州仪器 DSP 实验室

目　录

第1章 绪 论

1.1 最优化问题

最优化概念反映了人类实践活动中十分普遍的现象，即要在其他各方面的前提下，争取获得在可能范围内的最佳效果。因此，最优化问题成为现代数学的一个重要课题，涉及多种不同学科，其应用涉及系统控制、人工智能、模式识别、生产调度和计算机工程等各个领域。优化方法的理论研究对改进算法、扩宽算法应用领域和完善算法体系具有重要作用，所以优化技术已作为一个重要的科学分支受到众多学者的关注，已出现了各种各样的解决最优化问题的优化方法。

所谓最优化问题[1]就是在满足一定的约束条件下，寻找一组参数值，以使某些最优性度量得到满足。用数学描述为

$$\begin{aligned} &\min\sigma = f(X) \\ &\text{s.t. } X \in S = \{X \mid g_i(X) \leqslant 0, \quad i = 1,\cdots,m\} \end{aligned} \tag{1.1}$$

其中，$\sigma = f(X)$为目标函数(objective function)，$g_i(X)$为约束函数(constraints)，约束函数可有多个，S为约束域，X为n维优化变量。其中最大化问题也可转换为上述公式描述的最小化问题。

当一个数学模型满足以下条件时，称其为线性规划问题(linear programming problem)：

(1) 所有变量都是连续的；

(2) 只有一个目标函数(最大或最小)；

(3) 目标函数和约束函数都是线性的。

线性规划问题是很重要的，因为很多实际问题都可以描述成线性规划问题，其应用可涉及炼油管理、生产计划、分布、人力资源分布计划、金融和经济计划等。解决此类问题的方法有单纯形法(simplex algorithm)[2]。

当线性规划问题的部分或所有的变量局限于整数值时，我们称上述问题即为整数规划(integer programming)或整数线性规划(integer linear programming)问题，特别是当X仅能取0或1时，上述问题即为0-1整数规划问题。这类问题在实践问题中也会经常出现，因为很多决定性问题实际上是离散的(如：是/不是、去/不去)，我们必须从一组有限集合中选择一项。

在描述线性规划和整数规划问题中，我们都强调线性，但当$f(X)$和$g_i(X)$中

至少有一个函数为非线性函数时，上述问题即为非线性规划问题(nonlinear programming problem)。由于函数的非线性，使得问题的求解变得非常困难。

上述规划问题为单目标优化问题，但在实际的经济、生产、工程应用领域中普遍存在着对多个目标的方案、计划以及设计的决策问题。在解决这类问题时，决策者要综合考虑各种因素的制约，寻求满足多个目标的最佳设计方案，这就是所谓的多目标优化问题(multi-objective optimization，MO)。当考虑 k 个目标时，此类问题可描述为

$$\begin{aligned}&\min_{\boldsymbol{X}\in R}\boldsymbol{F}(\boldsymbol{X})\\&\boldsymbol{F}(\boldsymbol{X})=(f_1(\boldsymbol{X}),f_2(\boldsymbol{X}),\cdots,f_k(\boldsymbol{X}))^{\mathrm{T}}\\&R=\{\boldsymbol{X}\mid \boldsymbol{g}(\boldsymbol{X})\leqslant 0\},\quad \boldsymbol{g}(\boldsymbol{X})=(g_1(\boldsymbol{X}),g_2(\boldsymbol{X}),\cdots,g_m(\boldsymbol{X}))^{\mathrm{T}}\\&\boldsymbol{X}=(x_1,x_2,\cdots,x_n)^{\mathrm{T}},\quad \boldsymbol{X}\in R\subset E^n\end{aligned}\tag{1.2}$$

其中，$\boldsymbol{F}(\boldsymbol{X})$为优化目标向量，$\boldsymbol{g}(\boldsymbol{X})$为约束向量，$\boldsymbol{X}$ 为决策变量。对于多目标优化问题来说，其所包含的不同目标函数之间往往存在着一定的矛盾冲突，因此在求解过程中，很难在问题的约束集合 R 中找到一个解向量，能够使得 k 个目标函数同时达到最小。也就是说，多目标优化问题中的多个目标几乎不可能在同一解上同时达到最优。因此多目标优化问题的解往往不是单一的，而是一组解的集合。

当 $g_i(\boldsymbol{X})\leqslant 0(i=1,\cdots,m)$所限制的约束空间为整个 n 维欧氏空间，即 $\mathbf{R}^n$ 时，上述最优化问题变为无约束优化问题。优化问题可通过某些方法转换成无约束优化问题进行处理，如利用惩罚函数法将约束优化问题化为无约束优化问题。

1.1.1 函数优化问题与组合优化问题

优化技术可作为求解各种工程问题优化解的应用技术[3]，从应用方面归纳而言，最优化问题又可分为函数优化问题和组合优化问题两大类。

在函数优化问题中，函数优化的对象是一定区间内的连续变量，算法的性能比较通常是基于基准函数进行展开的，目前常用的基准函数有 Rosenbrock、Griewank、Rastrigin 和 Sphere 等各种不同类型的函数。这些函数在后面章节中将会有详细介绍。

而在组合优化问题中，组合优化的对象则是解空间中的离散状态。此类问题通常可描述为：令 $\Omega=\{s_1,s_2,\cdots,s_n\}$为所有状态构成的解空间，$C(s_i)$为状态 s_i 对应的目标函数值，要求寻找最优解 s^*，使得 $\forall s_i\in\Omega,C(s^*)=\min C(s_i)$。组合优化往往涉及排序、分类、筛选等问题，是运筹学的一个重要分支。典型的组合优化问题有：

(1) 旅行商问题(traveling salesman problem，TSP)：此问题是组合优化中最典型的 NP 难问题之一，给定 n 个城市和两两城市之间的距离，一个推销员要从其

中某一个城市出发，唯一走遍所有的城市，再回到他出发的城市，求最短的路线。其图论描述为：给定图 $G=(V,A)$，其中 V 为顶点集，A 为各顶点相互连接组成的边集，已知各顶点间的连接距离，要求确定一条长度最短的 Hamilton 回路，即遍历所有顶点当且仅当一次的最短回路。

(2) 聚类问题(clustering problem)：m 维空间上的 n 个模式 $\{\boldsymbol{X}_i \mid i=1,2,\cdots,n\}$，要求聚成 k 类，使得各类本身内的点最相近，譬如要求 $x^2=\sum_{i=1}^{n} \| \boldsymbol{X}_i^{(p)}-\boldsymbol{R}_p \|$ 最小，其中，R_p 为第 P 类的中心，即 $\boldsymbol{R}_p=\sum_{i=1}^{n_p} \boldsymbol{X}_i^{(p)}/n_p (p=1,2,\cdots,k)$，$n_p$ 为第 P 类中的点数。

(3) 加工调度问题(scheduling problem，如 Flow-shop，Job-shop)：Job-shop 问题是一类比 TSP 问题更为复杂的加工调度问题，是许多实际问题的简化模型。一个 Job-shop 问题可描述为：n 个工件在 m 台机器上加工，O_{ij} 表示为第 i 个工件在第 j 台机器上的操作，相应的操作时间 T_{ij} 为已知，事先给定各工件在各机器上的加工次序(称为技术约束条件)，要求确定与技术约束条件相容的各机器上所有工件的加工次序，使得加工性能指标达到最优。在 Job-shop 问题中，除技术约束外，通常还假定每一时刻每台机器只能加工一个工件，且每个工件只能被一台机器所加工，同时加工过程为不间断。若各工件的技术约束条件相同，一个 Job-shop 问题就转化为较简单的 Flow-shop 问题。进而，若各机器上各工件的加工次序也相同，则问题可进一步转化为置换 Flow-shop 问题。

(4) 0-1 背包问题(0-1's knapsack problem)：对于 n 个体积分别为 a_i、价值分别为 c_i 的物品，如何将它们装入总体积为 b 的背包中，使得所选物品的总价值最大。

(5) 着色问题(graph coloring problem)：对于 n 个顶点的无环图 G，要求对其各个顶点进行着色，使得任意两个相邻的顶点都有不同的颜色，且所用颜色种类最少。

1.1.2 优化算法的发展

解决最优化问题的优化算法可以分为经典优化算法和启发式优化算法。经典算法始于 1947 年美国数学家 Dantzig 提出的单纯形法，此算法是求解线性规划问题一种较为方便的方法。随后，Kamaka 提出了椭球算法(多项式算法)和内点法。对于非线性问题，起初人们试图用线性优化理论去逼近求解非线性问题，但效果并不理想。后来的非线性理论大多都建立在二次(凸)函数的基础上，也就用二次函数去逼近其他非线性函数。在此基础上提出许多经典的优化算法。无约束的优化算法包括：最速下降法(steepest)、共轭梯度法、牛顿法(Newton algo-

rithm)、拟牛顿法(pseudo Newton algorithms)、信赖域法。

随着社会的发展,实际问题越来越复杂,如全局最优化问题。经典算法一般都使用局部信息,如单个初始点及所在点的导数等,这使得经典算法无法避免局部极小问题。受大自然的启发,人们从大自然的运行规律中找到了许多解决实际问题的方法。对于那些受大自然的运行规律或者面向具体问题的经验、规则启发出来的方法,人们常常称之为启发式算法(heuristic algorithm)。这种思想是由于实际需要而在20世纪40年代提出,到50年代,启发式算法的研究逐步繁荣起来。随后,人们将启发式算法的思想和人工智能领域中的各种有关问题求解的收缩方法相结合,提出了许多启发式的搜索算法。其中贪婪算法和局部搜索得到人们的关注。60年代人们对数学模型和优化算法的研究越来越重视。70年代提出计算复杂性理论,NP完全理论告诉我们,许多实际问题不可能在合理的时间范围内找到全局最优解。由此必须引入新的搜索机制和策略,才能有效地解决这些困难问题。Holland模拟地球上生物进化规律提出了遗传算法(genetic algorithm),它与众不同的搜索机制再次引发了人们研究启发式算法的兴趣,从而掀起了研究启发式算法的热潮。80年代以后,模拟退火算法(simulated annealing algorithm)、人工神经网络(artificial neural network)和禁忌搜索(tabu search)相继出现。最近,演化算法(evolutionary algorithm)、蚁群算法(ant algorithms)、拟人拟物算法和量子算法等相继兴起,掀起了研究启发式算法的高潮。由于这些算法简单有效,而且具有某种智能,因而成为科学计算和人类之间的桥梁。

1.2 几种常见的启发式算法

全局优化算法主要可以分为两大类:确定性(deterministic)和概率(probabilistic)方法。大多数的确定性方法涉及启发性的应用,如调整轨迹(轨迹方法,trajectory methods)或加入惩罚(基于惩罚的方法,penalty-based methods)进行处理,从而跳出局部最小区域。而概率方法是依靠概率调整来决定搜索是否远离局部最优区域。

启发式算法是相对于最优算法提出的,可定义为:一个基于直观或经验构造的算法,在可接受的花费(指计算时间、占用空间等)下给出待解决组合优化问题每个实例的一个可行解,该可形解与最优解的偏离程度不一定事先可以预计。

另一种定义为,启发式算法是一种使得在可接受的计算费用内去寻找最好的解的技术,但它不一定能保证所得解的可行性和最优性,甚至在多数情况下,无法阐述所得解和最优解的近似程度。

各种启发式算法有自己的特点,当结合两种不同的算法将会有出乎的效果,所以启发式算法之间的联系是很紧密的,这里将介绍几种较常见的启发式算法的

基础理论和特点，因为后面章节中将会涉及这些启发式算法。

1.2.1 遗传算法

1975年，Holland[4]提出了遗传算法，它是一种有效的解决最优化问题的方法[5,6]。

遗传算法是一种探索(exploratory)过程，在解决复杂问题中它常常能够寻找到最优解的附近区域。在把此算法应用于任何任务前，需设计计算机的表示方式或者编码，这些表示形式成为染色体(chromosomes)。其中最为常用的表示方式为二进制数组。在确定表示方式后，为了在求解过程中达到改善解的目的，此算法主要通过四个步骤的过程，分别为评估(evaluation)、复制(reproduction)、组合(recombination)和突变(mutation)。

(1) 评估：在每一代进化过程中需完成的第一步骤是估计当前的染色体，即解码群体中的每个染色体，并评估它们的解决问题的能力。这种适应度的测量将会在下一步骤中用于决定将从特殊染色体中产生多少个后代。

(2) 复制：在这一步骤中将会基于评估结果而产生新的一代。在复制过程中将遵循的原则是：好的染色体将会复制更多的后代。这原则使得算法具有适者生存的优点。可用不同方式来确定复制后代的染色体数量，最为普遍的两种方式为比例(ratioing)和排序(ranking)。

(3) 组合：在上一步骤中，复制算子操作使得群体中增添了一批能够当前较好解决问题的后代，然而很多染色体是一致的，且与上一代相差不大，因为复制仅仅是产生现存染色体的多个复制体。但是组合是从群体中把染色体结合在一起，并产生新的染色体(这些染色体在上一代中是不存在的)。此算子能够很好地保持上一代的特征。最为普遍的组合方法是交叉，即从群体中任意选择两个个体，按照一定的交叉概率或比例，选择一个交叉点，并交换两个染色体中的分段部分。在组合过程中允许好的染色体与不好的染色体进行组合，这是基于一种思想：不管个体质量如何，它都不可能包含问题的完整答案，问题的解是存在于整个群体中，只有通过组合才能够找到最好的解。

(4) 突变：由于初始群体中有可能没有包含解决问题的所有必要信息，此外，没有复制后代的某些个体有可能带有很重要的信息，这些问题将通过突变这一步骤来进行弥补。在群体中注入新的信息被称为突变。这里同样有很多种方式实现突变操作，通常是基于特定的突变概率来随机改变每一代中一定数量的比特。

简单的遗传算法的流程如表 1.1 所示。

表 1.1 遗传算法流程图

开始
1. 初始群体
2. 估计群体质量
复制
组合
突变
3. 如果不满足循环停止条件将跳到步骤 2
结束

完成遗传算法的关键在于确定较为准确的群体大小、突变率、交叉比例和其他参数。每一个参数都很重要，例如群体大小相对于搜索空间很小的情况下，此算法将会很难有效地搜索整个领域，此外，太大的突变率将会破坏进化的稳定性。遗传算法比经典算法具有很多优点：①此算法不同于一般基于微积分的方法（如梯度下降法等），因为它不需要其他信息，且不容易落入局部最小点，而梯度下降法需要计算当前位置在错误平面上朝不同方向的梯度，并向最大负梯度方向移动，如果错误平面是平滑且没有局部最小值时，这种方法是可行的，但实际数据常常是多峰且具有多个局部最小点，遗传算法正是不受局部最小值的限制。②此算法可以在解空间内进行并行搜索，而不是点对点的搜索。因为此算法通过一群的试验解，可以有效地探索空间中的许多区域，这也是此算法对局部最小值不敏感的原因之一。③此算法代表了潜在的解，而不是解本身，所以它不需要完全理解问题模型，而仅仅需要能够评估试验解的正确性即可。

1.2.2 模拟退火算法

模拟退火算法最早的思想是由 Metropolis[7] 在 1953 年提出，此思想是模拟统计物理中固体物质的结晶过程。

退火是一种物理过程，一种金属物体在加热至一定温度后，它的所有分子在状态空间 D 中自由运动。随着温度的下降，这些分子逐渐停留在不同的状态。在温度最低时，分子重新以一定的结构排列。由统计力学的研究表明，在温度为 T 的情况下，分子停留在状态 r 满足玻尔兹曼概率分布

$$\Pr\{\overline{E}=E(r)\}=\frac{1}{Z(T)}\exp\left(-\frac{E(r)}{kT}\right) \tag{1.3}$$

其中，$E(r)$为状态 r 的能量，$k>0$ 为玻尔兹曼常量，$\overline{E}$为分子能量的一个随机变量，$Z(T)$为概率分布的标准化因子：

$$Z(T)=\sum_{s\in D}\exp\left\{-\frac{E(s)}{kT}\right\} \tag{1.4}$$

我们可以把优化问题类比成退火过程。如把解类比为状态，最优解类比为退火过程中能量的最低状态(也就是温度达到最低点时)，而代价函数类比为能量。上面公式的概率分布中具有最大概率的状态。更直接的理解方式为：在一个给定的温度，搜索从一个状态随机地变化到另一个状态。每一个状态到达的次数服从一个概率分布。当温度很低时，以概率1停留在最优解。在退火的过程中，如果搜索到好的解则接受；否则，以一定的概率接受不好的解，从而实现多样化或变异的思想，达到跳出局部最优解得目的。

简单的模拟退火算法的流程[8]如表1.2所示。

表1.2 模拟退火算法流程图

开始

1. 任选一个初始解 x_0；$x_i=x_0$；$k=0$；$t_0=t_{\max}$(初始温度)

2. 若在该温度达到内循环停止条件，则到步骤3；否则，从领域 $N(x_i)$ 中随机选一个 x_j，计算 $\Delta f_{ij}=f(x_j)-f(x_i)$；若 $\Delta f_{ij}\leqslant 0$，则 $x_i:=x_j$；重复步骤2

3. $t_{t+1}:=d(t_k)$；$k:=k+1$；若满足停止条件，终止计算；否则，回到步骤2

结束

从数学模型角度出发，可把模拟退火算法描述为：在给定领域结构后，模拟退火过程是从一个状态到另一个状态不断地随机游动。我们可以用马尔可夫(Markov)链描述这一过程。当温度 t 为一确定值时，两个状态的转移概率(transition probability)定义为

$$p_{ij}(t)=\begin{cases}G_{ij}(t)A_{ij}(t), & \forall j\neq i\\ 1-\sum\limits_{l=1,l\neq i}^{|D|}G_{il}(t)A_{il}(t), & j=i\end{cases} \tag{1.5}$$

其中，$|D|$ 表示状态集合中状态的个数。$G_{ij}(t)$ 称为从 i 到 j 的产生概率(generation probability)，其表示在状态 i 时，j 状态被选取的概率。比较容易理解的是 j 是 i 的邻居，如果在领域中等概率选取，在 j 被选中的概率为

$$G_{ij}(t)=\begin{cases}1/|N(x_i)|, & j\in N(i)\\ 0, & j\notin N(i)\end{cases} \tag{1.6}$$

$A_{ij}(t)$ 为接受概率(acceptance probability)，$A_{ij}(t)$ 表示从状态 i 产生状态 j 后，接受 j 的概率，如在模拟退火算法中接受的概率是

$$A_{ij}(t)=\begin{cases}1, & f(i)\geqslant f(j)\\ \exp(-\Delta f_{if}/t), & f(i)<f(j)\end{cases} \tag{1.7}$$

$\boldsymbol{G}(t)=(G_{ij}(t))_{|D|\times|D|}$、$\boldsymbol{A}(t)=(A_{ij}(t))_{|D|\times|D|}$ 和 $\boldsymbol{P}(t)=(P_{ij}(t))_{|D|\times|D|}$ 分别称为产生矩阵、接受矩阵和一步转移概率矩阵。

模拟退火算法也需要人工调整很多参数，如起始温度、温度下降的方案、固定温度时的迭代长度及终止规则等。人为因素有可能造成计算结果的差异，所以要