博士学位论文

粒子群算法的基本理论及其改进研究

作 者 姓 名 ：刘建华
学 科 专 业 ：控制科学与工程
学院(系、所）：信息科学与工程学院
指 导 教 师 ：樊晓平 教授

中 南 大 学
2009 年 3 月

分类号 VDC_____ 密级_____

博士学位论文

粒子群算法的基本理论分析及其改进研究

The Research of Basic Theory Analysis and Improvement on Particle Swarm Optimization

作 者 姓 名 ：刘建华

学 科 专 业 ：控制科学与工程

学院(系、所）：信息科学与工程学院

指 导 教 师 ：樊晓平 教授

论文答辩日期_____ 答辩委员会主席_____

中 南 大 学

2009 年 3 月

原 创 性 声 明

本人声明，所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了论文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得中南大学或其他单位的学位或证书而使用过的材料。与我共同工作的同志对本研究所作的贡献均已在在论文中作了明确的说明。

作者签名：_____ 日期：_____年___月___日

关于学位论文使用授权说明

本人了解中南大学有关保留、使用学位论文的规定，即：学校有权保留学位论文，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以采用复印、缩印或其它手段保存学位论文；学校可根据国家或湖南省有关部门规定送交学位论文。

作者签名：_____导师签名_____ 日期：____年___月___日

摘 要

粒子群算法在仿真生物群体社会活动的基础上，通过模拟群体生物相互协同寻优能力，从而构造出一种新的智能优化算法。但粒子群算法本身来源于生物群体现象，其理论基础并不完备。而且由于其属于随机的近似优化算法，主要应用于连续区域，因此该算法存在早熟收敛和对离散性的问题难以应用的缺点。因此，对粒子群算法的理论分析、算法改进及离散性问题的研究是具有重要意义的。本文在前人工作的基础上对标准粒子群算法和离散二进制粒子群算法进行分析、改进，获得以下结果：

(1).粒子群算法是一种启发式随机优化算法，每个粒子追逐自身最优粒子和全局最优位置搜索，并且追逐时带有随机因素。粒子群算法在这种随机搜索过程中，粒子最终会收敛于群体最优粒子。本文在增加随机性和粒子最优点更新的条件下，理论上证明了粒子的轨迹是收敛于群体最优粒子位置。根据分析的理论结果，进一步说明了算法权重选择的原理。

(2).由于粒子轨迹最终收敛于群体最优粒子，本文定义一个粒子间的相似度概念，设计计算群体粒子的多样性的概念公式—聚集度—通过计算群体粒子与群体最优粒子的平均相似度，度量粒子群的多样性程度。根据群体聚集度及其与群体最优粒子相似度，每个粒子随机产生变异，由此，构造了一种对标准 PSO 算法的改进算法，提高算法的全局搜索能力、避免早熟收敛，有效地提高标准 PSO 算法的性能。

(3).标准 PSO 算法的权重是平衡算法全局搜索与局部搜索能力的参数，其取值影响算法的性能。标准 PSO 算法的权重采用从早期偏大到晚期偏小的线性递减方法，但每个粒子的权重大小一样。本文根据粒子与群体最优粒子的相似度，对不同粒子赋予不同权重，使每个粒子的权重不同，并且随着算法迭代而动态变化，这样构造一种权重动态变化的粒子群算法。经仿真实验验证，该方法有效。

(4).PSO 算法的理论分析构造了一个数学模型，数学模型从数学角度清楚地体现算法本身的数学含义。本文利用此数学模型代替原始 PSO 算

法速度及位置的更新公式，得到一种新的进化算法，并且分析新的进化算法参数的选择。新算法形式能直接体现 PSO 算法的数学思想。经仿真试验检验，新算法效果不会差于标准 PSO 算法。本文将新的算法应用于求解单交叉口信号灯的时间优化分配，实验仿真结果表明，本文的算法在静态环境是很有效的。

(5).二进制离散 PSO 算法为求解二进制的离散组合优化问题而构造一种 PSO 算法。本文从位改变率、速度的期望值及遗传算法模式概念等三个方面对其进行分析。本文得到：二进制 PSO 算法不收敛于群体最优粒子，其位值随着迭代运行而越来越随机。因此，二进制 PSO 算法缺乏局部探测性且具有偏强的全局开拓性。本文对原始二进制 PSO 算法进行改进，使其产生符合 PSO 算法思想（跟随群体最优粒子）的形式。将改进的方法应用于求解 0/1 背包问题。通过实验对比，新改进的二进制 PSO 算法提高了原始二进制 PSO 算法的性能。

关键字 粒子群算法，收敛性，相似度，进化计算，二进制 PSO

ABSTRACT

Particle Swarm Optimization is an intelligence algorithm constructed by the simulation of biologic Swarm social activity, which imitated cooperative ability of searching optimal location of biologic swarm and developed the optimization algorithm. But, because it is a stochastic approximate algorithm, the theory of Particle Swarm Optimization is incomplete, and there exist deficiencies on premature convergence and difficulty of application in discrete problem. So, the research on the theory analysis, improvement and discrete problem of particle swarm optimization is very significant. In this thesis, we have analyzed and improved standard particle swarm optimization and binary particle swarm optimization. We have following conclusions:

Particle swarm optimization is a stochastic algorithm with heuristic information, each particle fly following the self optimal location and global optimal location with stochastic factor. It has been analyzed that the particles converge to the global optimal location with stochastic iteration going on. In the condition of adding stochastic factor and the optimal particle updation, it has been proved that the particle trajectories will converge to the swarm optimal location in this thesis.

During the running of the algorithm, the particles become more and more similar, and cluster into the best particle in the swarm, which make the swarm premature convergence around the local solution. In this thesis, a new conception, collectivity, is proposed which is based on similarity between the particle and the current global best particle in the swarm. And the collectivity was used to randomly mutate the position of the particles, which make swarm keep diversity in the search space. Experiments on benchmark functions show that the new algorithm outperforms the standrard PSO and other improved PSO.

In fact, PSO is a random evolution algorithm. However, during the evolution of the algorithm, the magnitude of inertia weight has impact on the exploration and exploitation of PSO, which is a contradiction. In this thesis, a new PSO algorithm, called as DPSO, is proposed in which the inertia weight of every particle will be changed dynamically with the distance between the

particle and the current optimal position. Experiments on benchmark functions show that DPSO outperform standard PSO.

The current theory of PSO has constructed a mathematic modal, which can give a clear essence of PSO from mathematic view.In this thesis, the velocity and location updating equation are replaced by the mathematic equation, and get a new evolutionary algorithm. By select appropriate parameters, the performance of new algorithm is not inferior to standard PSO by simulation on benchmark functions. The new algorithm was applied to the real-time dynamic model on multiphase traffic flows. A simulation experiment for the traffic model at a four-phase intersection is also performed in this thesis. The results show that the method is effective.

Binary PSO is a new method to solve discrete problem, which is applied in many area. In this thesis, Binary PSO has been analyzed through bit change rate, velocity expected value and genetic algorithm pattern theory. It has been found that binary PSO is not converge to global optimal particle, and the bit is more and more stochastic with iteration going on, so it is lack of local exploration. So, an improved binary PSO are proposed which meet the idea of PSO.

KEY WORDS Particle Swarm Optimization; Convergence; Similarity; Evolutionary Computation; Binary PSO

目 录

第一章 绪论

1.1 引言

优化问题是一个古老的问题，它所研究的内容是讨论在众多的方案中什么样的方案最优以及如何找出最优方案。优化问题也定义为：在满足一定约束条件下，寻找一组参数值，使系统的某些性能指标达到最大值或最小值。优化问题也是在工程技术和经济管理领域中一种常见的问题。例如，工程设计中怎样选择设计参数，使得设计方案既满足设计要求又能降低成本；资源分配中，如何分配有限资源，使得分配方案既满足各方面的基本要求，又能获得好的经济效益；原料配比问题中，如何选择确定各种成分的比例，才能提高质量，降低成本；城建规划中，如何安排工厂、机关、学校、商店、医院、住户和其他单位的合理布局，才能方便群众，有利于城市各行各业的发展；各个领域中诸如此类的问题还有很多很多。正因为这些问题驱动，使得优化已成为一门应用广泛、实用性强的学科。

长期以来，人们对优化问题进行了不懈地探讨和研究。早在 17 世纪，在英国科学家 Newtow 发明微积分时代，已经提出极值问题，后来又提出约束优化问题并提出 Lagrange 乘数法。1939 年苏联数学家 Kantorouicz 为解决生产组织中的问题，发表了《生产组织与计划中的数学方法》等论文，这是世界上最早研究线性规划的文章。1947 年法国数学家 Cauchy 研究了函数值沿什么方向下降最快的问题，提出最快下降法，这些方法都归为传统优化方法。传统方法是借助于优化问题的不同性质，通常将问题分为线性规划问题、非线性规划问题、整数规划问题和多目标规划问题等。相应的有一些成熟的常规算法，如应用于线性规划问题的单纯形法，应用于非线性规划的牛顿法、共轭梯度法，应用于整数规则的分枝界定法、动态规划等[1] 。

目前，这种传统优化算法能够解决许多工程、社会及经济中建立的实际优化问题。但是在实际应用中，尚存在相当多的涉及因素多、规模大、难度高和影响广的优化命题，存在多个局部最优解，并且它们可能存在不同的全局最优解。例如，流量工程系统优化、运输中的最优调度、生产流程的最优排序、资源的最优分配、农作物的合理布局、工程的最优设计以及国土的最优开发等等。对于这些问题，人们无法借助于传统的优化算法来求解，因为传统的优化算法面对这样的大问题无能为力，无论是在计算速度、收敛性、初值敏感性等方面均不能满足要求。

近 30 年来，人们从不同的角度出发对生物系统及其行为特征进行了模拟，产生了一些对现代科技发展有重大影响的新兴学科，开发出具有较强通用性的优化模式和方法——智能优化方法[1, 2] ，它们在解决一些复杂的工程问题时大有用武之地。从实际应用的观点看，这类新算法不要求目标函数和约束的连续性与凸性，甚至连

有没有解析表达式都不要求；对计算中数据的不确定性也有很强的适应能力，并且计算速度快等等，这些优点使这类算法在很短的时间里就得到了广泛应用，展示出方兴未艾的强劲发展势头。

在今天，智能优化算法相对而言还是一门新学科，其理论、方法还有待于进一步发展和完善，但它所具有的先进性、实用性、有效性使我们相信，智能优化算法不仅是科学研究的重要课题，而且会成为每一个工程师及科学家必须掌握的基础知识。这类算法主要有遗传算法（Genetic Alogrithm，启发于生物种群通过遗传和自然选择不断进化的功能）[3] 、人工免疫系统（Artificial Immune Systems，模拟于生物免疫系统的学习与认知功能）[4] 、蚁群算法（Ant Colony Opimization，模仿蚂蚁群体在路径选择和信息传递方面的行为）[5] ，还有模拟退火算法、禁忌算法等等。而粒子群优化算法（Particle Swarm Optimization，简称 PSO）也是此类算法。它是 R.Eberhart 博士和 J.Kennedy 博士于 1995 年发明[6, 7] ，源于对鸟群捕食的行为研究。

粒子群算法(PSO)同遗传算法类似，是一种基于迭代的优化工具，一种基于群体的随机优化技术。系统初始化为一组随机解，通过迭代搜寻最优值。同遗传算法等其他进化算法相比[8] ，都属于人工生命计算方法，但它又不同于其他进化算法，不是采用群体解的竞争机制来迭代产生最优解，而是采用群体解的合作机制来迭代产生最优解。此外，粒子群算法概念简单、容易实现，需要调节的参数偏少。因此，粒子群算法越来越受到人们的关注，其研究成为国内外的热点，尤其应用粒子群算法对工程技术问题的优化文献呈指数增加。但粒子群算法的理论分析与应用研究目前还处于初级阶段，还有很多方面的问题值得研究。

1.2 粒子群算法产生的背景

自然界中，鸟群运动的主体是离散的，其排列看起来是随机的，但在整体的运动中它们却保持着惊人的同步性，其整体运动形态非常流畅且极富美感。这些呈分布状态的群体所表现出的似乎是有意识的集中控制，一直是许多研究者感兴趣的问题。有研究者对鸟群的运动进行了计算机仿真，他们通过对个体设定简单的运动规则，来模拟鸟群整体的复杂行为。例如，1986 年 Craig Reynols 提出了 Boid 模型[9] ，用以模拟鸟类聚集飞行的行为，通过对现实世界中这些群体运动的观察，在计算机中复制和重建这些运动轨迹，并对这些运动进行抽象建模，以发现新的运动模式。之后，生物学家 Frank Heppner 在此基础上增加了栖息地对鸟吸引的仿真条件，提出了新的鸟群模型[10] 。

上述模型的关键在于以个体间的运算操作为基础，这个操作也就是群体行为的同步必须在于个体努力维持自身与邻居之间的距离为最优，为此每个个体必须知道

自身位置和邻居的位置信息。生物社会学家 E.O.Wilson 认为[11]，“至少从理论上，在搜索食物的过程中，群体中的个体成员可以预测地零星分布时，这种协作带来的优势是决定性的，远大于以食物的竞争带来的劣势。”以上两个例子说明，群体中个体之间信息的社会共享有助于群体的进化。

受上述鸟群运动模型的影响，社会心理学博士 James Kennedy 和电子工程学博士 Russell Eherhart 于 1995 年提出了粒子群算法（PSO）[6, 7]。粒子群算法其实也是一种演化计算技术，该算法将鸟群运动模型中的栖息地类比于所求问题空间中可能解的位置，通过个体间的信息传递，导引整个群体向可能解的方向移动，在求解过程中逐步增加发现较好解的可能性。群体中的鸟被抽象为没有质量和体积的“粒子”，通过这些“粒子”间的相互协作和信息共享，使其运动速度受到自身和群体的历史运动状态信息的影响。以自身和群体的历史最优位置对粒子当前的运动方向和运动速度加以影响，较好地协调粒子本身和群体之间的关系，以利于群体在复杂的解空间中进行寻优操作。James Kennedy 和 Ruell Eberhart 在 1995 年的 IEEE International Conference on Neural Networks 和 6th International Symposium on Micromachine and Human Science 上分别发表了题为“Particle swarm optimization” 和“A new optimizer using particle swarm theory” 的两篇论文[6, 7]，标志着粒子群优化算法的诞生，国内也有人译为微粒群算法。

1.3 粒子群算法的研究现状

粒子群算法(PSO)自提出以来，已经历了许多变形和改进，包括数学家、工程师、物理学家、生物学家以及心理学家在内的各类研究者对它进行了分析和实验，大量研究成果和经验为粒子群算法的发展提供了各许多合理的假设和可靠的基础，并为实际的工业应用指引了新的方向。2003 年 IEEE 第一届国际群智能研讨会在美国印第安纳州首府召开，此后将每年举办一次群智能国际研讨会。目前，对 PSO 算法研究和运用 PSO 算法的论文逐渐增加，图 1-1 是 ISI 数据库最近收录有关 PSO 算法论文的情况，从图形中可以看出，有关 PSO 论文呈指数级地增加。目前，PSO 的研究也得到了国内研究者的重视，并已取得一定成果。

十多年来，PSO的研究方向得到发散和扩展，已不局限于优化方面研究。网站 http://www.particleSwarm.net列出目前一些的开放问题，从中可看出PSO算法研究的扩展性。PSO算法按其研究方向分为四部分：算法的机制分析研究、算法性能改进研究、算法的应用研究及离散性PSO算法研究。算法的机制分析主要是研究PSO算法的收敛性、复杂性及参数设置。算法性能改进研究主要是对原始PSO算法的缺陷和不足进行改进，以提高原始PSO算法或标准PSO算法的一些方面的性能。目前技术与方法

的改进主要是增加算法的多样性、加强局部搜索性及融合其它智能优化算法的技术；PSO算法的应用研究主要是关于如何利用PSO算法对工程技术、经济及社会等需要优化的问题求解，其中包括多目标问题、约束问题、动态问题和大量实际应用问题；离散PSO研究主要针对离散性的优化问题，PSO算法如何进行优化求解，原始PSO算法主要是解决连续性的优化问题，而离散性问题存在特殊性，因此离散性问题的求解，PSO算法需要一些特殊技术进行处理，其研究问题主要包括离散二进制问题和一般组合优化问题。

字段:出版年	记录数	%，共 4387	柱状图
2008	1318	30.0433 %	
2007	1185	27.0116 %	
2006	921	20.9938 %	
2005	442	10.0752 %	
2004	222	5.0604 %	
2009	169	3.8523 %	
2003	85	1.9375 %	
2002	41	0.9346 %	
2001	3	0.0684 %	

图 1-1 ISI 数据库收录有关 PSO 算法的论文情况

PSO 算法提出若干年后，得到学者广泛注意与研究，特别是本世纪初以来，对其进行分析、改进与应用研究日渐增长。而相关的综述性文献报道国内外也很多[12-14]。以下根据一些文献从理论分析、算法改进、离散性及算法应用四个方面对 PSO 算法的研究现状作一综述。

1.3.1 粒子群算法的理论研究

PSO 算法的理论分析主要研究算法的收敛性。PSO 也是一种迭代进化算法、随机性算法、启发式优化算法。对于这类算法的理分论析主要是研究算法最终是否收敛于问题最优解。实际上，这方面的分析包括：算法是否收敛、收敛到哪？PSO 算法的理论分析一直是 PSO 研究的难点，目前多数研究者都是通过大规模计算来对算法进行分析和研究。PSO 算法理论分析的难点在于 PSO 算法具有随机性，使得很多常规数学方法手段都对其无效。

最早 Ozcan 和 Mohan 在假设 c_1 和 c_2 为常数，p_i 和 p_g 固定不变的情况下，通过理论分析将粒子随时间变化描述为波的行为，并对不同区域进行了轨迹分析[15]。Maurice Clerc 在 IEEE Transacion on Evolutioanary Computation 发表题为 The Particle Swarm—Explosion,Stability,and Convergence in Multidimensional Complex Space 论

文[16]，对 PSO 算法迭代公式的参数进行了初步分析。其基本手段是先将随机变量变为常量，然后用状态转移矩阵的方法研究一个粒子的运动轨迹，从而可以得到使得单个粒子运动轨迹收敛的参数约束条件。但该方法存在三个明显的缺陷：(1)未考虑粒子间的相互影响，也就是 PSO 算法的“群体性”和“交互性”；(2)忽视了算法中随机量的作用，启发式算法大都引入了人为的随机因素，这种人工噪声的作用不仅不起干扰的作用，而相反地起正面的作用；(3)粒子轨迹的稳定性并不代表算法的收敛性。尽管 Maurice Clerc 的分析有着明显缺陷，但其在 PSO 算法的研究中仍然有着重要意义，该论文所给出的参数约束条件在当前 PSO 算法应用领域中得到广泛使用。Solis 和 Wets 对随机优化算法提出了其全局收敛须满足的条件[17]，基于该理论，Frans Van den Bergh 在 Maurice Clerc 分析的基础上对算法收敛性做了进一步的分析[18, 19]，研究了参数对粒子轨迹的影响，通过举反例来证明原始 PSO 算法是无法保证局部收敛的，并在 Lebesgue 和 Borel 测度空间上对收敛性做了分析，提出一种在时间无限下保证收敛到局部最优的改进算法 GCPSO，这种改进算法对原始算法的迭代公式进行了修改，粒子群中的最佳粒子采用与其他粒子不同的迭代运算公式，从而使得最佳粒子始终处于运动状态，尽管该算法可保证收敛到局部最优，但其优化性能并不佳。I.C.Trealea 应用离散动态系统理论对简化 PSO 模型的动态行为和收敛性进行了分析[20]，根据这个简化模型的收敛条件推导出对参数选择的参考准则。国内也有不少文献对参数选择作了分析，提出参数选择的理论依据[21, 22]。

潘峰等[23] 建立了标准粒子群优化算法的三种典型模型，并对三种模型情况下的粒子群动态特性进行了分析，结果表明在没有新信息引入的情况下，粒子群优化算法的搜索空间非常有限，且容易陷入停滞点。但所有的这些证明都还存在进一步深入研究的必要。华中科技大学李宁通过建立随机过程模型分析群体的随机性收敛，这是利用随机过程理论进行分析的一种尝试[24,25]。金欣磊等也用随机过程对 PSO 算法作分析[26]，但只是利用了期望值，分析中采用的数学理论也只是差分方程。

但是，目前 PSO 算法的理论分析方法基本上是在简化 PSO 算法模型的条件下进行的，将 PSO 算法的随机性去掉、粒子群的最优位置固定不变，并且不考虑粒子间的相互作用。因此，目前 PSO 算法的理论分析成果是不完善的，没有真正反映算法的运行机制及真正的收敛特性。

1.3.2 粒子群算法的性能改进研究

对于 PSO 算法的性能改进主要是提高 PSO 算法求解最优解精度问题。PSO 算法是随机性算法，主要用于优化多峰问题。PSO 算法的主要特点是简单、易理解、参数较少。但是，PSO 算法存在易陷入局部最优解的缺点，因此对 PSO 算法需要加

以改进。目前，主要从下列几方面对 PSO 算法进行改进：1.改进参数权重的取值；2.增加 PSO 算法的多样性；3. 混合其他智能优化算法。

1.改进参数权重的取值

权重作用可以调节算法的全局探测能力与局部搜索能力的平衡性。标准PSO算法的惯性权重是线性减少，这使得算法在迭代初期探索能力较强，晚期可以使算法局部搜索能力增强。然而，搜索过程是一个非线性的复杂过程，惯性权重线性过渡的方法并不能正确反映真实的搜索过程。文献[27] 对算法中的重要参数惯性权重进行了系统的实验，分析了固定权重与时变权重的选择问题，并从问题依赖性、种群大小和拓扑结构等方面详细分析了惯性权重对于算法性能的影响。结果表明，惯性权重的问题依赖性较小，随着种群的增大，其取值应适当减小，局部版本下，惯性权重的选择具有更大的自由度。

文献[28] 给出了一种用模糊规则动态调整惯性权重方法，通过对当前最好性能的评价来对惯性权重制定相应的隶属度函数和模糊推理规则，确定惯性权重的增量。实验表明，与惯性权重线性减小的方法相比，模糊自适应方法有类似或更好的结果。文献[29] 给出了一种惯性权重随着迭代代数采用余弦减小的方法，也取得了良好的效果。文献[30] 提出一种非线性权重递减的方法，其主要思想是在标准PSO算法权重的线性递减方法中增加一个指数参数，使权重非线性变化。文献[31] 提出一种权重动态变化的方法，其思想是使权重根据算法在每步的情况动态变化，随着运行的迭代步数增加，权重可能会增加也可能会减小。文献[32] 提出一种带收敛因子的粒子群算法，实验显示，与使用惯性权重的粒子群优化算法相比，使用收敛因子的粒子群算法有更快的收敛速度。其实只要恰当选取参数值，两种算法是一样的。所以，带收敛因子的粒子群算法可以看作是带惯性权重的粒子群算法的特例。

2.多样性方面的改进

标准 PSO 算法主要存在早熟收敛的缺点，即算法过早停止在局部最优点，没有很好的进行全局探索，并且种群多样性随迭代代数增加下降过快，最终算法的最优解有可能不收敛到全局最优解。为了避免早熟收敛，一些研究者提出了通过控制种群的多样性来提高算法性能的方法。所谓多样性就是种群在搜索空间位置的分散程度，越分散多样性就越强。PSO 算法早熟就是收敛过快，多样性收缩太快。对于多样性可采用一些方法来进行度量，由量化结果再控制多样性[33, 34] 。

文献[35] 通过解决微粒间的冲突和聚集，尝试在微粒开始聚集时增加群体的多样性。算法为每个微粒赋一半径 r 值，以检测两微粒是否发生碰撞，如果发生碰撞，则采取弹离的方法。对于多峰函数，该方法显著改善了 PSO 算法的性能，维持了群

体的多样性。文献[36] 分析粒子和种群最优位置的距离，用来判断粒子是否为活泼粒子，当距离小临界值时，则认为是不活泼粒子，重新用新粒子代替。文献[37] 提出种群随机多代初始化等思想，给出了不同策略来增强种群多样性，使算法不至于过早陷入局部极小。另一方面，通过引入遗传算法的“变异”操作来增强全局搜索性能也是一种十分有效的方式。文献[38] 通过对粒子的速度或位置引入了一个小概率随机变异操作来增强种群多样性(dissipative PSO，简称 DPSO)，使算法能够有效地进行全局搜索，但是过大的变异率在增加种群多样性的同时也将导致群体发生混乱，使种群不能进行精确的局部搜索，延缓了算法的收敛速度。文献[39, 40] 深入分析了不同均值和方差的高斯变异操作对增强算法性能的影响。文献[41] 提出了一种改善粒子多样性的途径—自组织临界点控制，算法对每个粒子增加了当前临界值属性，如果两个粒子间的距离小于预定阈值，则增加彼此的临界值，重新分配其在搜索空间中的位置，以达到控制种群多样性的目的。文献[42] 在自组织算法的基础上给出了一种变异操作随时间变化的自适应层次 PSO 算法(self-organizing hierarchical PSO,简称 HPSO)以进一步提高搜索性能，并给出了适合变异操作的自适应参数选择方式。但是 HPSO 算法消除了速度公式中的惯性部分，其发生变异的条件是粒子速度为 0，使粒子不能快速、有效地逃出局部极小点。

增加群体多样性的方法来改进 PSO 算法的性能，这是比较常见的思想方法。有关这方面的文献比较多，在研究文献中采用的技术也是很多种。除了上面介绍的方法外，还有采用量子 [43] 、混沌 [44] 、耗散理论 [45] 、高斯变异、分层次等思想方法 [46-48] ，目的是增加多样性，提高全局搜索能力，进而能增强搜索最优点的能力。虽然这些研究工作已经给出了提高 PSO 算法全局搜索能力的方法，但是它们很难在提高搜索速度和保持种群多样性之间达到平衡。

3.与其他进化算法混合

PSO 算法与进化算法的本质目标是一致的，但它们之间思想原理有差别，技术方法也不同，各有优缺点。与进化算法混合得到新的算法，往往能产生一些比较好的效果。文献[49]提出了广义粒子群优化模型 GPSO，使其适用于解决离散的组合优化问题。GPSO 模型本质仍然符合粒子群优化机理，但是其粒子更新策略既可根据优化问题的特点设计，也可实现与已有方法的融合。该文献以旅行商问题 TSP 为例，针对遗传算法 GA 解决该问题的成功经验，使用遗传操作作为 GPSO 模型中的更新算子，进一步提出基于遗传操作的粒子群优化模型，并以新的算子作为模型中具体的遗传操作设计了基于 GPSO 模型的 TSP 算法。与采用相同遗传操作的 GA 比较，基于 GSPO 模型的算法解的质量与收敛的稳定性都有提高，同时计算费用显著降低。

文献[50]针对粒子群优化(PSO)容易陷入局部极小，提出将模拟退火(SA)引入并行 PSO 算法。这种模拟退火并行粒子群算法，结合了并行粒子群算法的快速寻优能力和 SA 的概率突跳特性，保持了群体多样性，从而避免了种群退化。此外，也有与微分进化算法进行混合[51]及其它进化算法混合的思想[52]。

1.3.3 离散粒子群算法研究

目前 PSO 算法的研究主要针对连续型的优化问题，即描述粒子状态及其运动规律的量都是连续的(实数)，而粒子群算法对离散型优化问题的研究较少。原始的 PSO 算法主是针对连续空间的优化进行设计的。PSO 算法在连续空间的函数领域取得的巨大成功，许多学者试图用它去解决组合优化问题，并且已经取得了一些成果。最早，为解决工程实际中的组合优化问题，Jame Kennedy 和 Rull Eberhart 提出的 PSO 算法的离散二进制版 [53] ，把位置表示为离散型的 0 和 1，速度表示成一个位串转变成 1 的概率，即把求得的速度通过一个 sigmoid 函数转化区间[0，1]之间的值。二进制 PSO 算法解决用二进制编码表示解空间的优化问题，使得 PSO 应用范围扩展到离散空间，特别是一些组合优化问题。比例，背包问题、调度问题[54-56] 等一些组合优化问题，广泛应用到更多实际问题中，比如数据挖掘 [57] 、生物信息 [58] 、图形学[59] 等领域。但是，二进制 PSO 算法的理论分析文献非常少。文献[60] 对二进制 PSO 算法的时间性作了分析，得到速度限制量与维数的关系，并在一些理论分析基础上提出来了改进二进制离散 PS0 算法，这是少有的分析理论。

当 PSO 算法用来求解不能采用二进制编码来表示的离散性问题时，则 PSO 算法需要解决以下问题：粒子位置、粒子的速度、粒子运算及参数与速度的运算的含义如何定义与表达。Clerc [61] 提出了用于求解 TSP 问题的 DPSO 算法，通过重新定义微粒的位置、速度、以及它们之间的加、减、乘等操作的含义，得到一种新的离散 PSO 算法，用于求解旅行商问题。尽管该算法的效果较差，但提供了一种解决组合优化问题的新的思路，即使在简单的 TSP 问题上，其性能与其它算法(如 ACO 算法)相比仍有不小差距。国内的钟一文等[62] 对此作一变种的改进，效果有所改进。另外，国内高尚等[63] 提出的求解 TSP 问题的混合粒子群(HPSO)算法，把遗传算法的交叉算子和变异算子对染色体的作用看作是粒子位置的移动，也可以看作是一种离散 PSO 算法。

文献[64] 使用 PSO 算法去解决单一机器调度问题，它采用随机键表示法去表示粒子的位置，通过根据粒子在每一维的值对作业进行排序，就可以把粒子的位置映射为一个合法的调度。文献[65] 也用相同的方式把连续 PSO 算法应用到置换 flowshop 问题及单一机器带权调度问题的求解中。国内李宁等[29] 用 PSO 算法去解带时间窗

车辆路径问题，位置和速度都表示为整数，但运算还是采用连续量的运算法则，而对运算的结果向上取整，如果超过范围，则按边界取值。显然，这些方法还局限在对经典的连续 PSO 算法进行针对问题的修改，并没有充分考虑到离散型组合优化的特点，不可避免地存在表示冗余大，搜索效率低等缺点。

文献[12] 指出：当处理整形变量，PSO 有时很容易陷入局部最优，似乎 PSO 能定位最优区域，但不能详细探测。原始 PSO 原理是从单个个体经验和同伴经验中来学习。如何更有效的运用这些规则到离散问题还是一个开放性问题，原始 PSO 算法直接转换到离散问题不是唯一选择。

1.3.4 粒子群算法的应用研究

PSO 算法能应用于广泛的领域。对于那些非线性的多峰问题和没有特别方法可使用或用特别的方法不能得到满意结果的问题，PSO 算法具有很好的应用前景。PSO 算法的应用是非常多样性，难以加以全面概括。在 google 和 IEEE Xplore 数据库中，可搜索大量 PSO 应用的论文。Poli 对此作了比较全面综述[66, 67] ，他把 PSO 算法的应用领域分为 26 个不同类别，根据 Xplore 中搜索到的 1100 篇有关 PSO 算法的文献作数据统计，其中有 700 篇是有关 PSO 算法应用的。他把这些应用文献归纳出以下应用领域：图像与视频分析；电子网络分布；控制工程应用；电子应用；天线设计；电力系统：调度；设计；通讯设计与优化；生物医药；数据挖掘；模糊系统与控制；信号处理；神经网络；组合优化；机器人；预测与预报；模型；故障诊断与恢复；传感器网络；计算机图形与可视化；发机动设计或优化；治金；音乐制作与游戏；安全与军事应用；财经与金融。

最近，Alec Bank 等在文献[68, 69] 对 PSO 算法的应用作了介绍。其主要包括：人工神经网络应用：主要用来训练神经网络中权重及一些参数 [70] ，也有用来对神经网络集成构造 [71] ；数据挖掘的应用：主要应用于数据挖掘中的优化问题，其中有分类任务 [72] 、决策树改进 [73] 、贝叶斯网络训练 [74, 75] 、数据聚类[76, 77] 等；机器人路规划的应用：主要应用于寻找机器人的最短路径问题 [78] ；此外应用于生物信息的文献越来越多 [58,79-81] 。只要是有关复杂性的优化问题，PSO 算法基本都能发挥作用。

1.4 本文研究思路与意义

虽然 PSO 算法得到快速的研究与应用，但 PSO 算法的发展历史尚短，在理论基础与应用推广上都还存在以下问题:

第一，PSO 算法理论研究不够深入，也不够完善。PSO 算法收敛模型的建立和收敛性的分析是 PSO 算法的基础，目前收敛模型的建立和收敛性的分析都是在去掉

随机性、单维的基础上进行分析，并且没有考虑目标函数对算法的影响，这些分析方法并不能完全揭示 PSO 算法的运行本质。如何在考虑 PSO 算法的随机性问题，考虑目标函数对算法的影响下，对 PSO 算法行为进行分析是 PSO 算法理论研究的一个重要推进方向。

第二，PSO 算法存在早熟收敛的问题。PSO 算法是一种随机性智能搜索算法，算法需要在早期全局开拓与晚期局部搜索之间平衡。PSO 算法应用于优化复杂问题时，往往会遇到早熟收敛的问题，也就是种群在还没有找到全局最优点时已经聚集到一点停滞不动。早熟收敛不能保证算法收敛到全局极小点，这是由于 PSO 算法早期收敛速度较快，但到寻优的后期，其结果改进则不甚理想，即缺乏有效的机制使算法逃离极小点。因而，算法早熟收敛研究也是一个值得研究的问题。

第三，PSO 算法不太适用对离散性问题进行优化。PSO 算法主要应用于连续性问题的优化，而当面对离散问题优化时，原始 PSO 算法需要改进。目前离散 PSO 算法研究相对较少，而且对存在的离散 PSO 算法分析更少见。因此，离散 PSO 算法的分析是一个十分有意义的工作。

第四，任何一个算法都有自己的应用局限性，PSO 算法也不例外。它如何克服自己的应用局限性，如何结合其它算法去解决实际问题，也是大量学者研究的课题。

针对上述问题，本文展开了细致的研究，对不同问题提出了相应的改进。并将改进的算法应用到实际优化问题中。

1.5 本文研究内容与创新点

在本文中，根据提出的问题，对 PSO 算法进行了较为深入的研究。本文的主要研究内容与创新点可归纳如下:

(l).PSO 算法的理论分析是对 PSO 算法性能评证的基础，也是改进 PSO 算法的依据。现有的理论分析方法主要在去掉随机性、单维、最优点静态及不考虑目标函数的情况下进行的，其理论研究还不够完善。本文在增加随机性和最优点变化的条件下，利用优化的基本理论，对 PSO 算法的收敛性作进一步分析，所得的结论有助于进一步理解 PSO 算法的行为本质，也有助于进一步改进 PSO 算法。

(2).基于 PSO 算法的理论分析结果，本文根据其收敛的数学形式，设计一种新的迭代进化算法，其基本原理与标准 PSO 算法相同。但该方法理论基础更为直观、有效。利用多峰函数实验测试，与标准 PSO 算法作比较，新的迭代进化算法有效实用。

(3).随着粒子群体不断进化，粒子群体逐渐向群体最优点聚集，随着聚集程度的越来越强，PSO 算法多样性就会降低，PSO 的算法全局优化能力就会减弱。据此，

本文提出聚集度概念，用以量化 PSO 算法的多样性程度，根据多样性的量度动态地对粒子的位置产生变异，增强算法全局搜索能力，提高算法寻找全局最优点的能力。

(4).PSO 算法的权重是平衡全局搜索能力与局部探测能力的参数。标准 PSO 算法的权重是线性递减，但各粒子的权重是相同。本文根据粒子与群体最优粒子的相似度，对不同粒子赋予不同权重，使每个粒子权重不同，并且每个粒子随着算法迭代动态变化，这样构造一种权重动态变化的粒子群算法。

(5).二进制 PSO 算法是求解离散问题的 PSO 算法，本文分别从位改变率、速度的期望值及模式定理等三方面对二进制 PSO 算法作分析。结果表明，二进制 PSO 算法的随机搜索能力很强，但算法局部搜索能力较弱。因此，对该算法作了改进，改变其概率取 1 值公式，增强算法最终收敛能力。并将算法应用 0/1 背包问题实验对比。

(6).在实际应用中，单交叉路口的信号灯的时间分配是一个实用的约束优化问题。在一个固定的时间周期内，需要根据不同方向车流量动态分配走向的时间。最优的时间分配是在固定的时间周期内使得单交叉路口滞留的车辆越来越少，或者趋于稳定。本文利用改进的 PSO 算法来求解单交叉路口的信号灯的最优配时。

1.6 本文内容安排

本文共七章。第二章主要介绍 PSO 的相关基础知识。第三章主要在现有的理论分析基础上，对标准 PSO 算法进行分析。第四章根据第三章的分析，提出了粒子间的相似度概念，根据粒子之间相似度量，提出一种变异方法来增强粒子群的多性，提高算法的求优能力。并根据惯性权重对 PSO 算法性能的影响，提出权重动态变化的 PSO 算法。把改进的 PSO 算法应用到基准函数上进行实验对比。第五章根据 PSO 算法的数学模型，提出一种代替 PSO 算法位置与速度更新公式的进化算法，分析了新算法的参数设置。对两算法作对比实验。第六章从位改变率、速度期望值及模式定理等三方面对离散性二进制 PSO 算法作了分析，根据分析结论对离散的二进制算法进行改进。最后一章给出了论文的总结，并指出了未来的研究期望。

第二章 粒子群算法研究基础

2.1 优化算法的基本概念

2.1.1 优化问题

优化技术涉及的工程领域很广，问题种类与性质繁多。优化问题研究的是目标函数在某个区域上全局最优点的特征和计算方法。

定义 2.1：优化问题(Optimization Problem)一般可以描述为：

$$\min f(x), \quad (2\text{-}1)$$

$$s.t.\ x \in S.$$

其中 S 为可行域，也称为解空间，是一个非空集合；$f(x)=(x_1, x_2, \ldots, x_n)$为目标函数，是定义在 S 上的实值函数。$\min f(x)$表示求目标函数在解空间 S 中的最小值。

优化问题也可以表示为求目标函数最大值，即 $\max f(x)$。但求最大值可以等价地转化为最小值 $\min(-f(x))$，所以在本文若没有特别说明，优化问题都指求目标函数的最小值。

对一个问题要进行优化求解，第一步就是要建立优化问题的数学模型，一个优化问题的数学模型主要包括三个部分：目标函数、可行域（搜索空间）及优化的参数所需要满足的约束条件。当一个问题没有约束条件时，就称这个优化问题为无约束优化问题，否则称为约束优化问题。本文更多是处理无约束优化问题。

优化问题的最小值可分局部最小值和全局最小值，它们的定义给出如下：

定义 2.2：如果存在点 $x^* \in B$，使得$\forall\ x \in B$ 有 $f(x^*) \leqslant f(x)$，其中 $B \subset S \subseteq R^n$，$S$ 为目标函数限定的搜索空间，则点 $x^* \in S$ 称为在 B 内的局部最小点，$f(x^*)$为局部最小值。

定义 2.3：如果存在点 $x^* \in S$，对于$\forall x \in S$, 有 $f(x^*) \leq f(x)$，其中 $S \subseteq R^n$，S 为目标函数限定的搜索空间，则点 $x^* \in S$ 是一个全局最小点， $f(x^*)$为全局最小值。

优化问题可分为函数优化问题和组合优化问题两大类，其中函数优化的对象是一定区间内的连续变量，而组合优化的对象则是解空间的离散状态。前者优化问题的解空间 S 是连续的，如一些称为 Benchmark 的典型问题及函数优化问题；后者的解空间是离散状态的，如组合优化中的背包问题、装箱问题、Job-shop 和 Flow-shop 问题、集合覆盖问题、TSP 问题等。

函数优化问题通常可以描述为：令 S 为 R^n 上的有界子集（即变量的定义域），$f: S \rightarrow R$ 为 n 维实值函数，所谓函数 f 在 S 域上全局最小化就是寻求点 $x^* \in S$ 使得 $f(x^*)$在 S 域上全局最小，即 $\forall x \in S$：$f(x^*) \leq f(x)$。

组合优化问题通常可描述为：令$\Omega=\{s_1,s_2,...,s_n\}$为所有状态构成的解空间，$C(s)$为状态 s_i $(i=1,2,...,n)$对应的目标函数值，要求寻找最优解 s^*，使得 $\forall s_i \in \Omega$，$C(s^*)=min\ C(s_i)$。组合优往往涉及排序、分类、筛选等问题，它是运筹学的一个重要分支。

随着工程优化问题的日趋复杂，优化问题的目标函数的性态也变得越来越复杂，通常是不连续、不可微、高度非线性的，没有明确的解析表达式，且具有多个峰值、多目标的特征。因此，对优化问题的求解困难得多，一般的全局优化问题是 NP-hard 的，检验是否存在满足约束条件的可行解更是 NP-hard，对于一般情况下大规模的 NP-hard 问题，传统优化方法在电子计算机平台上都是很难解

2.1.2 优化算法及其分类

所谓优化算法，其实就是一种搜索过程或规则，它是基于某种思想和机制，通过一定的途径或规则得到满足用户要求的优化解。对于优化算法分类从不同角度来分，有不同分类方法。下面介绍两种分法：

按优化算法发展历史分：传统优算法和现代优化算法。传统优算法主要根据目标函数的解析式，得出求解目标函数的全局最小值，主要包括：单纯法、最速下降法、牛顿法、共轭梯度法等等。现代优化算法主要是针对传统优化算法难以解决的问题而开发设计出来的优化算法。传统优化依赖于目标函数的解析式，而很多目标函数解析很复杂的，难以利用其导数等性质来求解其最优点，只是利用其目标函数值。因此，现代优化算法主要是一些直接方法 [82] ，例如，模式搜索、Rosenbrock 算法、Powell 方法，尤其是近几十年来所出现的随机性智能优化算法 [2] ，例如，进化算法、神经网络、群智能算法、模拟退火算法等等。粒子群优化算也属于现化优化算法。

按优化算法的求解机制来分：非智能优化算法和智能优化方法。非智能优化方法主要指传统优化算法及现代优化方法中的一些直接方法。而智能优化算法主要是通过模拟某些人类智能或自然现象或过程而发展起来的算法，这些算法具有人类智能或自然生物社会现象的直观性，通过模拟这些现象来构造出一些优化算法，通常称之为智能优化算法。这些算法包括：遗传算法、模拟退火算法、蚁群算法、神经网络算法等等，本文研究的粒子群算法也是属于智能优化算法。

按求解搜索的目标点来分：全局优化算法及局部优化算法

2.1.3 局部优化算法

求解局部最优解的算法叫局部优化算法。智能随机优化算法通常要求从解空间 S 一个起始点 $x_0 \in S$ 开始，局部优化算法应当保证算法能够发现起始点 x_0 所属一个子搜索空间的局部最优解。

局部优算法，也称为局部搜索算法，指寻找初始点附近的最好可能解的算法。该类算法利用梯度信息、特定问题的启发信息或简单地在邻域里随机寻找一个位置，如果该试探位置比当前所处位置更好，则前进到该位置。局部搜索算法不能保证发现最好解，它只能发现初始点领域内的最好解，也即局部最优解。

这类算法实际更有用，因为他能找到目前位置领域中的局部最优点。提高局部搜索的一个方法是简单地选择另一个随机点作为初始点再次开始搜索。然而，经过多次试验后，发现更好解的可能性很小，尤其是在变量的个数即搜索空间的维度很高的时候。

2.1.4 全局优化算法

求解全局最优解的算法叫全局优化算法。不考虑起始点 x_0，算法能够发现搜索空间 S 的全局最优解。这种算法往往包括两个步骤：全局步骤和局部步骤。这里局部步骤通常是局部优化算法的运用，而全局步骤则被设计成确保算法将移向区域 B_i，在区域 B_i 局部步骤能够找它的局部最小值。这些方法将被称为全局收敛算法，意味着他们能够收敛到他们的起始点 x_0 局部最小值，这些方法能够发现全局最小值，假如给定的起始位置 x_0 被选择是正确的。

在大多数问题中，目标是发现最好的解。由于这一目标并不总是可以达到，发现尽可能好的解也是可接受的。

全局优化方法致力于发现问题的全局最好解。这通常是通过证明其遍历性，即，保证任一状态出现概率不为 0。因此，在足够的时间内，算法能够争取最好解。虽然这一性质很喜鼓舞人心，但只在无限的迭代进行搜索时才成立。

另一个性质比能在无限时间内收敛到最优点更具有吸引力，那就是保证用较少的目标函数计算次数找到满意的解。

2.2 智能优化算法

智能优化算法属于现代优化算法，这些算法主要是启发于人类智能、生物群体的社会性或自然现象的规律。大多数智能优化算法包含随机性因素，因此也可归为随机性的不确定性优化算法。智能优化算法在搜索空间进行随机性搜索时，受一些启发式信息指导，因此智能优化算法称为启发式的随机搜索优化算法。随着人工智能学科的发展，这类算法到目前为止产生很多种。比较典型的算法有：进化计算（遗传算法、遗传规划、进化策略和进化规划）、群体算法、神经计算、模拟退火算法、免疫算法、文化算法、DNA 计算及神经网络计算等，将来还会有不同类似算法产生。下面简单介绍一些主要算法。

2.2.1 进化计算

进化计算（Evolutionary Computation）主要指采用进化原理而得到的智能优化算法。进化原理主要是指达尔文的适者生存原理[83, 84]。进化计算(Evolutionary Computation,EC)定义了一些模拟进化的方法，这些方法都是基于种群自然选择和随机变异等生物进化机制的全局性概率搜索算法。目前进化计算领域存四种不同的方法[85]:遗传算法(Genetic Algorithms, GAs)、遗传规划 (Genetic Programming, GP)、进化策略(Evolutionary Strategies, ES)以及进化规划(Evolutionary Programming, EP)。

1.遗传算法

遗传算法是仿真生物遗传学和自然选择机理，通过人工方式所构造的一类搜索算法，从某种程度上说遗传算法是对生物进化过程进行的数学方式仿真。霍兰德(Holland)于1975年在他的著作《Adaptation in Natural and Artificial Systems》[3,86]中首次提出遗传算法，并主要由他和他的学生发展起来的。

遗传算法类似于自然进化，通过作用于染色体上的基因寻找好的染色体来求解问题。与自然界相似，遗传算法对求解问题的本身一无所知，它所需要的仅仅是对算法所产生的每个染色体进行评价，并基于适应值来选择染色体，使适应性好的染色体有更多的繁殖机会。在遗传算法中，通过随机方式产生若干个所求解问题的数字编码，即染色体，形成初始群体；通过适应度函数给每个个体一个数值评价，淘汰低适应度的个体，选择高适应度的个体参加遗传操作，经过遗传操作后的个体集合形成下一代新的种群。对这个新种群进行下一轮进化。这就是遗传算法的基本原理。遗传算法实质上是一个迭代计算过程，其实施的主要步骤包括编码、群体初始化、遗传操作、解码、评价和终止判定六步。

编码的主要任务是建立解空间与染色体空间一一对应关系。将问题结构变换为位串形式，这种表示的过程叫做编码；相反地，将位串形式的编码表示变换为原问题结构的过程叫做解码或译码。把位串形式的编码表示叫做染色体，有时也叫个体。评价主要是量化染色体的适应能力，引入对问题中的每一个染色体都能进行度量的函数，这个函数叫做适应度函数。通过适应度函数来决定染色体的优劣程度，它体现了自然进化中的优胜劣汰原则。简单遗传算法的遗传操作主要有三种：选择、交叉、变异。许多改进的遗传算法大量扩充了遗传操作，以达到更高的效率。

2.遗传规划

遗传规划(Genetic Programming, GP)的思想是由Stanford大学的J.R.Koza在

20 世纪 90 年代初提出 [87,88] 。遗传传规划是用字符串作为染色体去表达所研究的问题，而且字符串的长度常常是固定的。然而，现实中的问题往往很复杂，有时不能用简单的字符串表达问题的所有性质，为解决这个问题遗传规划应运而生。遗传规划用广义的计算机程序形式表达问题，它的结构和大小都是可以变化的，从而可以更灵活地表达复杂的事物性质。

与遗传算法类似，遗传规划都是从随机产生的初始群体出发，用适应度衡量个体的优劣，并采用复制、交换、突变等操作，经过一代一代的优胜劣汰，逐步得出最优的数学表达式。不同的是，它们在问题的表达方式上。GA 用定长的字符串，而 GP 则是形式可变的计算机程序结构。

3.进化策略

进化策略最早是德国工业大学的 I.Rechenberg 和 H.P.Schwefe1 于 1964 年为优化物体形状参数而提出[89]。早期的进化策略中，种群只有一个个体且只使用变异操作，具体在每一个操作中，变异后的个体与其父代个体进行比较，从中选择最优一个作为新一代个体，这就是所谓的(1+1)策略。它们所使用的变异算子主要是基于正态分布的变异操作。如用传统的实型数表达问题，其表达形式如下:

$$x(t) = x(t)+N(0,\sigma) \tag{2-2}$$

式中 $x(t)$是用实数表示的第 t 代个体，$N(0,\sigma)$是独立的随机数，服从正态分布，后者的数学期望为 0，标准差为σ。

由于(*1+1*)策略难以收敛到最优解，且搜索效率相对较低。其改进的方法就是增加种群内个体的数量，即(μ+1)进化策略。此时种群内含有μ个个体，随机抽取一个个体进行变异，然后取代群体中最差的个体。为了进一步提高搜索效率，后来又提出了(μ+λ)进化策略和(μ，λ)进化策略。(μ+λ)进化策略是根据种群内的μ个个体采用变异和重组产生λ个个体，然后将这μ+λ个个体进行比较选择μ个体最优者；而(μ，λ)进化策略则在新产生的λ(≥μ)个个体中比较最优者。

进化策略也是一个反复迭代的过程，它从随机产生的初始群体出发，经过突变、重组(交换)、选择等进化操作，改进群体的质量，逐渐得出最优解。

4.进化规划

20 世纪 60 年代中期，L.J.Fogel 等人为有限状态机的演化提出进化规划来求解预测问题，其基本思想是源于对自然界中生物进化过程的模拟 [90] 。进化规划与进化策略几乎同时出现，并平行发展。最早的进化策略只采用单个个体，而最早的进化规划则是采用多个个体组成群体共同进化。90 年代，D.B.Fogel 又将进化规划扩展到实数型的问题 [91] ，使其能用来求解实数空间中的优化问题，并在变异中引入

正态分布技术。其表达形式为:

$$x(t+1)=x(t)+\sqrt{f(x)}N(0,1) \tag{2-3}$$

式中 *x(t)*是用实数表示的第 *t* 代个体； *N(0, 1)*是独立的随机数，服从(0,1)标准正态分布，*f(x)*是 *x* 的适应度。

进化规划的个体表示同进化策略一样，不同之处在于它不用杂交算子，变异与选择方式与进化策略不同。

2.2.2 群智能算法

群智能算法主要灵感于生物群体的社会行为而开发出来的智能优化算法。群智能作为一种新兴的演化计算技术已成为越来越多研究者的关注焦点。群智能的群体指的是[92, 93] “一组相互之间可以进行直接通信或者间接通信（通过改变局部环境）的主体，这组主体能够合作进行分布式的问题求解”，而群智能则是指“无智能的主体通过合作表现出智能行为的特性”。群智能在没有集中控制且不提供全局模型的前提下，为寻找复杂的分布式问题求解提供了基础。目前，群智能研究领域主要有两种算法：蚁群算法、粒子群算法。而粒子群算法正是本文所研究内容，将下一节介绍，这里对蚁群算法作一个简介。

蚁群算法灵感于蚂蚁群体的社会行为特征。蚂蚁之所以被称为社会性昆虫，是因为它具备组成社会的三个要素：有组织、有分工，还有相互通信和信息的传递。生物学家和仿生学家发现，蚁群在寻找食物时，蚂蚁倾向于跟随信息素浓度高的轨迹。这些轨迹是由觅食的个体在行进过程中留下的，以指导其他个体朝相同食物源方向前进。被访问得越多的地方，其信息素的浓度就越高。由于蚂蚁在寻找食物源和返回巢穴时的访问，邻近巢穴的道路上聚集了浓度更高的信息素。

从蚂蚁群体寻找最短路径的觅食行为中受到启发,意大利学者 Dorigo 等人 1991 年提出了一种模拟自然界蚁群行为的优化算法—人工蚁群算法，简称蚁群算法(ACO)，并且成功用于求解组合优 TSP 问题 [91, 94, 95] 。蚁群算法对搜索空间的“了解”机制主要包括三个方面：（1）蚂蚁的记忆；（2）蚂蚁利用信息素进行相互通信；（3）蚂蚁的集群活动。基本蚁群算法模型由下面三个公式描述：

$$p_{ij}^{k}=\tau_{ij}^{\alpha}\eta_{ij}^{\beta}/\sum_{j\in A}\tau_{ij}^{\alpha}\eta_{ij}^{\beta} \tag{2-4}$$

$$\tau_{ij}(n+1)=\rho^{*}\tau_{ij}(n)+\sum_{k=1}^{m}\Delta\tau_{ij}^{k} \tag{2-5}$$

$$\Delta\tau_{ij}^{k}=\frac{Q}{\sum L_k}$$ 如果第 k 个蚂蚁经过了由 i 到 j 的路径

在上述三式中：*m* 为蚂蚁个数；*n* 为迭代次数；*i* 为蚂蚁所在位置；*j* 为蚂蚁可以到达的位置；*A* 为蚂蚁可以到达位置的集合；η_{ij} 为启发性信息，这里为由 *i* 到 *j* 的

路径的能见度，即 $1/d_{ij}$；L_k 为目标函数，这里为两点间欧氏距离；τ_{ij} 由 i 到 j 的路径的信息素强度；$\Delta\tau^k_{ij}$ 为蚂蚁 k 由 i 到 j 的路径上留下的信息素数量；α为路径权；β为启发式信息的权；ρ为路径上信息数量的蒸发系数；Q 为信息素质量系数；为蚂蚁 k 从位置 i 移到位置 j 的转移概率。

2.2.3 其他智能优化算法

智能优化算法主要来源人体或生物群体的生理特点或社会行为特点，通过模型化现有生理研究和社会研究的成果而灵感产生的一些优化算法。除了前面两类智能优化算法，实际还包括其他不同的智能优算法，其中主要有模拟退火算法、禁止忌算法和神经网络算法，还有人工免疫算法、文化算法、DNA 计算等等。

免疫算法 [4] ：免疫算法引入免疫算子(Immune Operator)。免疫算法在实际操作过程中，首先对所求解的问题(这里视为抗原，Antigen)进行具体分析，从中提出最基本的特征信息(即疫苗，Vaccine)；其次，对此特征信息进行处理，以将其转化为求解问题的一种方案(根据该方案而得到的各种解的集合统称为基于上述疫苗所产生的抗体，Antibody)；最后，将此方案以适当形式转化成免疫算子以实施具体的操作。这里需要说明的是，待求解问题的特征信息往往不止一个，也就是说针对某一特定的抗原所能提取出的疫苗也可能不止一种，那么在接种疫苗过程中可以随机地选取一种疫苗进行注射，也可以将多个疫苗按照一定的逻辑关系进行组合后再给予注射。所以，免疫算法主要是提取疫苗、接种疫苗和免疫选择来完成。前者提高适用度，后者则为了防止群体的退化。免疫算法也应用于典型的组优合化问题—TSP 问题 [96] 。

模拟退火算法：模拟退火算法来源于固体退火原理，将固体加温至充分高，再让其徐徐冷却，加温时，固体内部粒子随温升变为无序状，内能增大，而徐徐冷却时粒子渐趋有序，在每个温度都达到平衡态，最后在常温时达到基态，内能减为最小。根据 Metropolis 准则，粒子在温度 T 时趋于平衡的概率为 $E\text{-}\Delta E/(kT)$，其中 E 为温度 T 时的内能，ΔE 为其改变量，k 为 Boltzmann 常数。用固体退火模拟组合优化问题，将内能 E 模拟为目标函数值 f，温度 T 演化成控制参数 t，即得到解组合优化问题的模拟退火算法：由初始解 i 和控制参数初值 t 开始，对当前解重复“产生新解→计算目标函数差→接受或舍弃”的迭代，并逐步衰减 t 值。当算法终止时，当前解为所得近似最优解，这是基于蒙特卡罗迭代求解法的一种启发式随机搜索过程。退火过程由冷却进度表(Cooling Schedule)控制，包括控制参数的初值 t 及其衰减因子 Δt、每个 t 值的迭代次数 L 和停止条件 S。模拟退火算法与初始值无关，算法求得的解与初始解状态 S(是算法迭代的起点)无关；模拟退火算法具有渐近收敛性，已在理

论上被证明是一种以概率1收敛于全局最优解的全局优化算法;模拟退火算法具有并行性。

神经网络是在人工智能领域中研究最早的智能行为。在几十年发展过程中，神经网络产生了许多模型，其中递归神经网络中 Hopifield 模型能够用于优化计算，最早是用于组合优化问题––TSP 问题 [97] 。

文化算法：文化是一个将个体人的以往经验保存于其中的知识库，新的个体人可以在知识库中学到他没有直接经历的经验知识，没有这些信息,那么个体适应环境的唯一方法就是通过实验和犯错误来获取经验。受到这些想法启发，Reynolds于1994年基于文化系统的进化模型提出文化算法(Cultural Algorithm, CA) [98] 。文化算法是一种基于种群的多进化过程的计算模型,为进化搜索机制和知识存储的结合提供了一个构架。从进化角度看，任何一种符合文化算法要求的进化算法都可以嵌入文化算法框架中作为种群空间的一个进化过程。目前文化算法已应用于资源调度、函数优化、欺骗探测、数据挖掘、遗传规划、动态环境建模等领域。而国内刚刚开始关注文化算法的研究。

DNA 计算：DNA 计算的创始人是美国南加州大学的莱昂那多•阿德莱曼教授，他于 1994 年利用 DNA 计算方法解决了一个著名的数学难题“七顶点哈密尔顿路径” [99] 。最近，科学家们开始利用 DNA 计算来创造生物计算机，放在人体或生物体上工作，其计算结果可通过荧光蛋白的活动来读取。DNA 计算是利用 DNA 双螺旋结构和碱基互补配对规律进行信息编码，将要运算的对象映射成 DNA 分子链，通过生物酶的作用，生成各种数据池，再按照一定的规则将原始问题的数据运算高度并行地映射成 DNA 分子链的可控性的生化反应过程。最后,利用分子生物学技术(如聚合链反应 PCR、超声波降解、亲和分析、克隆、诱变、分子纯化、电泳、磁珠分离等）检测所需要的运算结果。

2.3 粒子群优化算法（PSO）

2.3.1 粒子群算法的基本形式

在 PSO 算法中，许多简单实体—粒子—都放在问题的搜索空间里，每一个粒子位置的目标函数值都被评价。每个粒子根据历史所处的最优位置和整个群体全优位置，带着一些随机扰动决定下一步的移动。最终，粒子群作为一个整体，象一个鸟群合作寻觅食物，很有可能靠向目标函数最优点移动。

粒子群算法（PSO）是一种基于迭代模式的优化算法，最初被用于连续空间的优化。在连续空间坐标系中，粒子群算法的数学描述如下 [100] ：

一个由 m 个粒子（particle）组成的群体在 D 维搜索空间中以一定速度飞行，每

个粒子在搜索时，考虑到了自己搜索到的历史最好点和群体内（或邻域内）其他粒子的历史最好点，在此基础上变化位置（位置也就是解）。粒子群的第 i 粒子是由三个 D 维向量组成，其三部分分别为：

目前位置： $x_i=(x_{i1},x_{i2},...,x_{iD})$；

历史最优位置： $p_i=(p_{i1},p_{i2},...,p_{iD})$；

速度： $v_i=(v_{i1},v_{i2},...,v_{iD})$；

这里 $i=1, 2, ..., n$。目前位置被看作描述空间点的一套坐标，在算法每一次迭代中，目前位置 x_i 作为问题解被评价。如果目前位置好于历史最优位置 p_i，那么目标位置的坐标就存在第二个向量 p_i。另外，整个粒子群中迄今为止搜索到的最好位置记为：$p_g=(p_{g1},p_{g2},...,p_{gD})$。

对于每一个粒子，其第 d 维（$1\le d\le D$）根据如下等式变化：

$$v_{id}=v_{id}+c_1\cdot\text{rand}()\cdot(p_{id}-x_{id})+c_2\cdot\text{rand}()\cdot(p_{gd}-x_{id}) \tag{2-6}$$

$$x_{id}=x_{id}+v_{id} \tag{2-7}$$

其中加速常数 c_1 和 c_2 是两个非负值，这两个常数使粒子具有自我总结和向群体中优秀个体学习的能力，从而向自己的历史最优点以及群体内或领域内的全局最优点靠近。c_1 和 c_2 通通常等于 2。*rand()*是在范围[0，1]内取值的随机函数。V_{max} 是常数，限制了速度的最大值，由用户设定。粒子的速度被限制在一个范围[$-V_{max}$，V_{max}]内，即在速度更新公式执行后，有：

$$if\ v_{id}<-V_{max}\quad then\quad v_{id}=-V_{max}; \tag{2-8}$$

$$if\ v_{id}>V_{max}\quad then\quad v_{id}=V_{max}; \tag{2-9}$$

当把群体内所有粒子都作为邻域成员时，得到 PSO 的全局版本；当群体内部分成员组成邻域时得到 PSO 的局部版本。局部版本中，一般有两种方式组成邻域，一种是索引号相临的粒子组成邻域，另一种是位置相临的粒子组成邻域。粒子群优化算法的邻域定义策略称为粒子群的拓扑结构。本文主要是研究全局版本。

2.3.2 粒子群算法的基本流程

基本粒子群算法流程如下:

Step1.初始化:在问题空间的 D 维中随机产生粒子的位置与速度；

Step2.评价粒子:对每一个粒子，评价 D 维优化函数的适用值；

Step3.更新最优:1).比较粒子适用值与它的个体最优值 *pbest*，如果优于 *pbest*，则将其 *pbest* 位置就是当前粒子位置。2).比较粒子适用值与群体全体最优值 *gbest*，如果目前值好于 *gbest*，则设置 *gbest* 位置就当前粒子位置

Step4.更新粒子：按照式(2.6)和(2.7)改变粒子的速度和位置

Step5.停止条件：循环回到步骤 Step2,直到终止条件满足，通常是满足好适用值和最大的迭代代数。

对应以上的算法流程，其粒子群算法的基本框架如图 2-1。

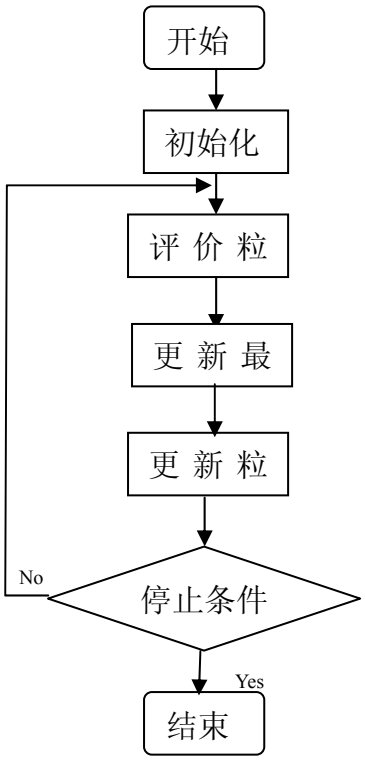

图 2-1 粒子群算法的基本框架

2.3.3 标准粒子群算法

为了进一步提高算法的性能,Y.Shi 和 R.Eberhart 在 1998 年的论文中将惯性权重 ω 引入 [101] ，将速度更新方程修改为：

$$v_{id} = \omega \cdot v_{id} + c_1 \cdot \text{rand}() \cdot (p_{id} - x_{id}) + c_2 \cdot \text{rand}() \cdot (p_{gd} - x_{id}) \quad (2\text{-}10)$$

ω 为惯性权重，它决定了粒子先前速度对当前速度的影响程度，从而起到平衡算法全局搜索和局部搜索能力的作用。文献[101] 通过仿真得了权重应该是[0.9,1.2]范围的值,优化性能具有较好。文献[101,102] 指出了权重 ω 的自适应策略，即随着迭代的进行，线性减少权重 ω 的值。这种策略使得算法在迭代初期探索能力较强，可以不断搜索新的区域，然后收敛能力逐渐增强，使算法在可能的最优解周围精细搜索。这是目前使用最广泛的标准 PSO 算法(Standard Particle Swarm Optimization),本论文所指 PSO 算法都为这种标准 PSO 算法。

这里需要引入最大权重 w_{max}, 最小权重 w_{min}, t_{max} 为运行为最大迭代代数，则权重 w 随着迭代运行代数 t 而线性减少为:

$$w = w_{\max} - \frac{t}{t_{\max}}(w_{\max} - w_{\min}) \tag{2-10}$$

文献[101] 通过实验指出当权重为 1.4 到 0.4 线性变化，优化的效果较好。然而，搜索过程是一个非线性的复杂过程，惯性权重线性过渡的方法并不能正确反映真实的搜索过程。

2.3.4 粒子群算法的控制参数

1. 惯性权重ω

对于粒子群算法的权重ω解释上节作了简述，这里再作一点详细阐述。原始粒子群算法分为三部分：v_{id}、$c_1 \cdot \text{rand}() \cdot (p_{id} - x_{id})$和$c_2 \cdot \text{rand}() \cdot (p_{gd} - x_{id})$。没有第一部分$v_{id}$，相当于是局部搜索算法（local search aglorthm）；没有第一部分，所有粒子群很容易趋向于同一样位置。只有当全局最优解刚好在初始的搜索空间，会有更大的机会使 PSO 找到最终解。因此，最后的解严重依赖于初始种群，所以没有第一部分，算法很可能是展示局部搜索能力。另一方面，如果增强第一部分v_{id}，则粒子就会扩展搜索空间的能力，也就是有能力扩展新的搜索区域。所以，如果增强第一部分v_{id}，则算法的全局搜索能就增强。局部搜索能力和全局搜索能力都有利于解决一些种类的问题。因此，算法的全局搜索与局部搜索必须存在一个平衡。对不同问题存在不同的平衡力，而惯性权重ω的引入起到一个平衡局部搜索能力和全局搜索能力的作用。从试验的效果来说，当权重固定值的范围是[0.9，1.2]时，优化会得到一个好的结果，能更好地找到全局最优解。

从试验中可以看到，大的惯性权重对初始解有更少依赖性，PS0 算法对探索新的区域有更强的能力。权重 w 起到平衡全局搜索与局部搜索能力的功能。一个大的惯性权重有利于全局探测能力（搜索新的区域），而一个小的惯性倾向于局部探索，精细搜索目前的小区域 [8] 。对于任何优化搜索算法，一个好的思想早期具有更强的探测能力（全局搜能力），能够发现好的解种。晚期需要更强的探索能力（局部搜索能力）去搜索局部好的解区域。因此，权重 w 是一个随时间递减函数而不是一个固定不变的值。开始赋给大值，而后线性减少到一定值，一个好的递减值范围是从 1.4 到 0.4。文献[102]所做的实验再一次说明采用权重 w 线性递的 PSO 算法，其性能得到很大的提高且能获得更好结果。

2. 群体规模m

显然 m 越大，相互协同搜索的粒子就越多，更能发挥 PS0 的搜索能力。然而群体过大，需要计算的时间大幅增加。有文献反应，群体增长到一定数量，再增加对

搜索算法能力并没多大增加。如果 m=1,PSO 算法就是个体搜索技术，没有全局信息可用，显然很容易陷入局部最优。当 m 很大时，PSO 算法全局优化能加强，不仅搜索时间增加长，而且收敛全局最优点的速度将非常慢。目前，在更多的文献中，这个参数取值为 20。

3.学习因子:c1和c2

在PSO算法中，学习因子是控制粒子向自我历史经验和群体最优个体学习的因子，从而控制向群体内或邻域内最优点靠近。与权重w作用类似，学习因子也能起到平衡局部搜索与全局探索能力，只不过学习因子值越大，越有利于算法收敛，增加局搜索能力，这与权重刚好相反。c_1和c_2通常等于2，不过在文献[18,20] 中也有其他的取值。但是一般c_1等于c_2，并且范围在0和4之间。在PSO算法的理论分析中，对权重与学习因子取值作了比较深入研究 [20] ，其取值大小与权重有关联。分析结果基本与相关文献采用的取值基本一致。文献[22] 利用试验也对权重与学习因子参数作了详细分析，得到结果与理论分析结果基本一致。

4.最大速度vmax

最大速度v_{max}决定粒子在一次迭代中最大的移动距离。v_{max}若较大，探索能力增强，但是粒子容易飞过最好解。v_{max}较小时，开发能力增强，但是容易陷入局部最优。有分析和实验表明，设定v_{max}的值可以通过惯性权重的调整来实现。但现在的实验基本上对v_{max}进行初始设定以后固定不变，一般将v_{max}设定为每维变量的取值范围，而不必细致地选择与调节。

5.停止准则

一般使用最大迭代次数或可以接受的满意解作为停止准则。

6.拓扑结构

对于拓扑结构,PSO 算法创始者最早就对此进行了研究。提出几中不同结构模型：其全局模型、环形模型和局部模型。全局模型 PSO 将整个群体作为粒子的邻域，速度快，不过有时会陷入局部最优；局部模型 PSO 将索引号相近或者位置相近的个体作为粒子的邻域，收敛速度慢一点，不过很难陷入局部最优。在实际应用中，可以先用全局 PSO 找到大致的结果，再用局部 PSO 进行搜索。本文所采用模型是全局模型。

7.粒子空间的初始化

与其化智能随机搜索算法一样，PSO较好地选择粒子的初始化空间，将大大缩短

收敛时间。但真正要较好选择初始化却是问题依赖的，而真正随机搜索算法与问题无关。所以更为合理的初始化是使初始的结果完全随机性的。例如，对粒子的位置初始应该让每粒子在整个搜索空间按均匀分布取样，这才是较为合理。

参数选择是算法一个很重的问题，参数选择是否恰当直接影响算法的性能。对参数选择其实是问题依赖性的，对参数分析也是PSO算法分析研究一个方向，不少文献对算法作过实验或理论分析[20, 29, 103]。

2.3.5 粒子群算法与进化算法比较

不象遗传算法、进化编程和进化策略，PSO 没有应用选择操作[102]。正是速度按照它自己最好位置和同伴的最好位置更新，粒子带着更新速度飞行。PSO 只是一种智能优化算法，并不使用“适者生存”原理。PSO 一定程度上与进化编程相似，也与文化算法相关，速度添加到当前位置，有如进化编程的变异操作。PSO 的“变异”是受粒子自的“飞行”经验和群体“飞行”经验影响。换句说，PSO 是带有“感知”执行“变异”。而进化算法不同于 PSO 算法，进化计算的工作原理是对搜索空间进行一群候选解搜索。通过在候选解之间合作与竞争，当运用到复杂优化问题时，这些技术常能更快地发现最优点。最普通使用这种基于群体进化计算技术是来自于自然界进化规律的启发式算法。四个有名的例子是遗传算法、进化编程、进化策略和遗传编程。不同于进化计算，粒子群算法(PSO)启发于社会行为的仿真。在文献[104]与五种进化算法作为了比较，五种进化算法分别是：遗传算法、蚁群算法、蛙跳算法、Menic 算法。这里主要比较 PSO 算法与遗传算法及蚁群算法。比较其与遗传算法及蚁群算法的异同点[105]。

1. PSO算法与遗传算法比较[8]

在 PSO 中，一个粒子有如于 GA 中的一个成员（染色体）。一个粒子代表求解问题的候选解。GA 中有选择、交叉和变异操作算子，而 PSO 并没有这些操作算子。

PSO算法和GA算法的相同点:

(1).都属于仿生算法。PSO算法主要模拟鸟类觅食、人类认知等社会行为而提出；GA算法主要借用生物进化中“适者生存”的规律。

(2).都属于全局优化方法。两种算法都是在解空间中随机产生初始种群，因而算法在全局的解空间进行搜索，且将搜索重点集中在性能高的部分。

(3).都属于随机搜索算法。PSO算法在认知项和社会项前都加有随机数；而GA算法的遗传操作均属随机操作。

(4).都隐含并行性。搜索过程是从问题解的一个集合开始的，而不是从单个个体开始，具有隐含并行搜索特性，从而减小了陷入局部极小的可能性。

(5).对高维复杂问题，往往会遇到早熟收敛和收敛性能差的缺点，都无法保证收敛到最优点。

PSO算法和GA算法不同点

(1).PSO算法有记忆，所有粒子对好的解作为知识保存；而在GA算法中，以前的知识随着种群的改变而被破坏。

(2).PSO算法中的粒子仅仅通过当前搜索到最优点进行共享信息，所以很大程度上这是一种单项信息共享机制。而在GA算法中，染色体之间相互共享信息，使得整个种群都向最优区域移动。

(3).GA算法的编码技术和遗传操作比较复杂，而PSO算法相对于GA算法，不需要编码，没有交叉和变异操作，粒子只是通过内部速度进行更新，因此原理更简单、参数更少、实现更容易。

(4).在收敛性方面，GA算法已经有了较成熟的收敛性分析方法，并且可对收敛速度进行估计；而PSO算法这方面的研究还比较薄弱。尽管已经有简化、确定性版本的收敛性分析，但将确定性向随机性的转化尚需进一步研究。

(5).在应用方面，PSO算法主要应用于连续问题，包括神经网络训练和函数优化等，而GA算法除了连续问题之外，还可应用于离散问题，比如TSP问题、工作车间调度等。

2.粒子群算法与蚁群算法(ACO)比较

PSO算法和ACO算法都是群体智能算法，为更清楚地认识PSO算法和ACO算法，下面对两者也做简单比较。

PSO算法和ACO算法的相同点:

(1).都属于仿生算法。PSO算法和ACO算法主要模拟觅食、人类认知等社会行为而提出。

(2).都属于全局优化方法。算法在全局的解空间进行搜索，且将搜索重点集中在性能高的部分。

(3).都属于随机搜索算法。

(4).都有记忆，好的解的知识在所有粒子中都被保存。

(5).隐含并行性。搜索过程是从问题解的一个集合开始的，而不是从单个个体开始，具有隐含并行搜索特性，从而减小了陷入局部极小的可能性。并且由于这种并行性，易在并行计算机上实现，以提高算法性能和效率。

(6).对高维复杂问题，往往会遇到早熟收敛和收敛性能差的缺点，都无法保证收敛到最优点。

PSO算法和ACO算法不同点：

(1).PSO算法的粒子通过当前搜索到的最优点进行共享信息，所以很大程度上这是一种单项信息共享机制。而在ACO算法中，每个个体只能感知局部的信息，不能直接使用全局信息。

(2).与ACO算法操作比较，PSO算法技术简单，其粒子只是通过内部速度进行更新，因此原理更简单、参数更少、实现更容易。

(3).在收敛性方面，ACO算法已经有了较成熟的收敛性分析方法，并且可对收敛速度进行估计；而PSO算法这方面的研究还比较薄弱。

(4).在应用方面，PSO算法主要应用于连续问题，包括神经网络训练和函数优化等，而全局ACO算法除了连续问题之外，还可应用于离散问题，比如TSP问题、工作车间调度等。

第三章 标准粒子群算法的分析

3.1 引言

粒子群算法（PSO）的理论分析主要回答的问题是：“算法是否最终能够收敛？是否真正收敛到问题的全局最优解？”这样的问题通常并不容易回答。因为，粒子群优化算法是随机性、带有启发式的算法，并且群体的粒子之间是相互作用的，其粒子运行轨迹很复杂。从纯理论的观点来看，随机性算法如果能以概率 1 收敛就是令人满意的。从工程实际的应用上讲，收敛的速度以及解的有效性和稳定性则更为重要。但深入的理论分析和研究能为算法在工程实际应用中的参数选择、解的表达式构造以及算法改进提供重要依据。

PSO 算法虽然提出只有十来年，对其的研究主要集中算法效率的改进及其应用研究，而算法理论性分析相对偏少，尤其是对算法的收敛性研究目前并不完善[106-109]，其研究方法主要是在简化系统条件下利用线性系统理论分析单个粒子的收敛性问题。但算法的随机性及其动态性如何影响算法运行的轨迹，在算法中如何起作用，却很少有文献分析。

本章在已有的分析理论[18] 基础上，考虑算法随机性及最优点粒子更新的条件，对 PSO 算法作进一步地深入分析，丰富 PSO 算法的理论，有利于算法的改进。本章主要是针对标准 PSO 算法进行分析。

3.2 PSO 算法的分析方法

PSO 算法的理论分析存在两个角度的分析方法[24]。一种是分析单个粒子运动轨迹，其方法主要是对单个粒子的飞行轨迹建立系统模型，考察单个粒子的动态变化状态。单个粒子的行为过程在一定程度上反映了群体的变化趋势，但这种分析方法忽略群体之间相互合作的因素，难以考虑随机性因素。另一种对整个群体的行为研究，分析群体一般性的搜索过程，建立合理的数学模型，讨论算法的收敛性。整体行为研究主要是采用随机过程模型，分析粒子群体随机状态变化。但这种方法很难建立随机过程模型。

对于单个粒子运动轨迹的分析，为了获得数学上的研究模型，PSO 理论分析方法常常做一些简化的假定：孤立的单个个体、搜索停止（即，没有更好的解发现）并且缺少随机性。因此，最后的数学模型是近似的，并且需要试验验证。

1998 年，Ender Ozcan 和 C.K.Mohan 第一个对 PSO 算法作理论分析，其分析方法在前面的简化模型上进行[15, 110]：单个粒子的行为、独立、一维，没有随机性，并且没有进化。粒子个体最优(p_i)和群体最优位置(p_g)被假定为一致。没有权重ω，

也没速度限制。文献[15] 扩展了类似的工作，多个多维粒子被覆盖， p_i 和 p_g 也没有要求一致。

文献[33, 111, 112] 运用同样的假定：一个粒子，一维，确定性行为。在这种条件下，粒子群是一个离散线性系统，粒子状态的动态性通过求解状态转移矩阵的特征值被确定的[39-41]。如果特征根的量值小于 1，则粒子将收敛于平衡点。因为系统的特征值是系统参数的函数，这种模型能建议哪个参数设置能够保证收敛。

文献[18] 也研究了粒子的简化模型：单维粒子、随机缺失及最优点静止。权重被包在模型中，考虑动态系统的特征值。其目的确定怎么设置系数能保证系统是稳定的。文献 [16] 在简化算法的模型上对单个粒子建立一个差分动态方程，分析了参数与单个粒子收敛关系。文献 [20] 建立单个粒子的差分方程，在收敛条件下分析参数的选择范围。

3.3 粒子运动轨迹分析的结论

Fan van den Bergh 在标准 PSO 系统的简化模型[36] 中，利用线性系统理论分析了粒子的收敛性，并得到粒子轨迹的显式表达式，并且粒子会收敛到一个固定点，即群体最优点与个体最优点的加权和。根据文献[36]的思想，这节利用位置分析和速度分析两种方法推导粒子的轨迹。

3.3.1 位置分析方法

为了叙述方便，对标准 PSO 算法的两个更新方程重新列出如下：

$$v_{id} = \omega \cdot v_{id} + c_1 \cdot \text{rand}() \cdot (p_{id} - x_{id}) + c_2 \cdot \text{rand}() \cdot (p_{gd} - x_{id}) \tag{3-1}$$

$$x_{id} = x_{id} + v_{id} \tag{3-2}$$

两个基本式子主要用于 D 维空间 n 个粒子的表达。因简化算法的条件，不失一般性，把两个下标 d 和 i 去掉。粒子轨迹的分析对不同粒子、在不同维数里是一样的，所以不考虑粒子下标 i 和维数下标 d。粒子的轨迹将在离散时间步骤中考虑，所以 v_{id} 和 x_{id} 可以表示为 $v(t)$ 和 $x(t)$，表示是粒子在时间步 t 时的速度与位置，并且标准 PSO 算法还有下式更新公式：

$$\begin{aligned} &If \ \ f(x(t)) < f(p), \ \ then \ \ p = x(t) \\ &If \ \ f(p) < f(p_g), \quad then \ \ p_g = p \end{aligned} \tag{3-3}$$

现假设 $\phi_1 = c_1 \cdot rand()$，$\phi_2 = c_2 \cdot rand()$，$\phi = \phi_1 + \phi_2$，如果 p，p_g，ϕ_1和ϕ_2保持固定的值，则由(3-1)式和(3-2)式消去速度相关项可以得到(3-4)式：

$$x(t+1) = (1 + w - \phi) \cdot x(t) - w \cdot x(t-1) + \phi_1 \cdot p + \phi_2 \cdot p_g \tag{3-4}$$

(3-4)式的齐次矩阵形式为：

$$\begin{bmatrix} x(t+1) \\ x(t) \\ 1 \end{bmatrix} = A \cdot \begin{bmatrix} x(t) \\ x(t-1) \\ 1 \end{bmatrix} \tag{3-5}$$

其中 $A = \begin{bmatrix} 1+w-\phi & -w & \phi_1 p + \phi_2 p_g \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

(3-5)式中系数矩阵 A 的特征多项式为：

$$(\lambda - 1)(\lambda^2 - (1+w-\phi)\lambda + w) = 0 \tag{3-6}$$

方程（3-6）存在三个特征根，除 $\lambda = 1$ 外，另两个根为：

$$\alpha = \frac{1+w-\phi+\sqrt{(1+w-\phi)^2 - 4w}}{2} \tag{3-7}$$

$$\beta = \frac{1+w-\phi-\sqrt{(1+w-\phi)^2 - 4w}}{2} \tag{3-8}$$

由此（3-4）式可以写成

$$x(t) = k_1 + k_2 \alpha^t + k_3 \beta^t \tag{3-9}$$

这里 k_1，k_2 和 k_3 是常数，由系统的初始条件决定。三个常数的确定需要三个方程，但根据(3-4)式，只要给定 x_0，x_1 就能算出 x_2。实际上只要给定初始粒子位置 x_0 和初始速度 v_0。由三个初始条件，根据(3-9)式推出下列公式：

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & \alpha & \beta \\ 1 & \alpha^2 & \beta^2 \end{pmatrix} \cdot \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix} \tag{3-10}$$

解方程组(3-10)式，再根据(3-7)和(3- 8)两式，可推得：

$$k_1 = \frac{\phi_1 p + \phi_2 p_g}{\phi_1 + \phi_2} \tag{3-11}$$

$$k_2 = \frac{\alpha(x_1 - x_0) + x_1 - x_2}{\gamma(\beta - 1)} \tag{3-12}$$

$$k_3 = \frac{\alpha(x_1 - x_0) + x_1 - x_2}{\gamma(\beta - 1)} \tag{3-13}$$

由(3-9)可知，$x(t)$要收敛，必须是α，β的绝对值小 1，而其值由(3-7)和(3-8)两式决定。因此得到，如果 $max(||\alpha||, ||\beta||)<1$，则粒子的位置 $x(t)$按下式收敛：

$$\lim_{t \to \infty} x(t) = (1-a)p + a p_g \tag{3-14}$$

这里 $a = \frac{\phi_1}{\phi_1 + \phi_2}$，所以 $a \in [0, 1]$。

(3-14)式成立的条件是 $max(||\alpha||, ||\beta||)<1$ 得到满足。根据 $max(||\alpha||, ||\beta||)<1$，可以计算参数ω，$\phi_1$，$\phi_2$ 需要满足什么关系才能保证(3-14)式成立，即粒子的轨迹收敛。计算过程如下：

根据(3-7)和(3-8)两式，当 $(1+\omega-\phi)^2<4\omega$ 时，α，β是一个复数；当 $(1+\omega-\phi)^2\geq 4\omega$ 时，α，β是一个实数。

(1).当 $(1+w-\phi)^2<4w$ 时，α，β是一个复数，则：

$$||\alpha||=||\beta||=\frac{1}{2}\sqrt{(1+\omega-\phi)^2+4\omega-(1+\omega-\phi)^2}$$

$$=\omega$$

当ω<1，且 $(1+\omega-\phi)^2<4\omega$ 时，则(3-14)式成立，

所以，当 $\begin{cases}\omega<1\\(1+\omega-\phi)^2<4\omega\end{cases}$ 成立时，粒子的轨迹收敛。

(2).当 $(1+\omega-\phi)^2\geq 4\omega$ 时，α，β是一个实数，则：

根据(3-7)和(3-8)两式，可以得到，

当 1+ω-φ>0 时，||α||>||β||；

当 1+ω-φ≤0 时，||α||≤||β||。

(a).当 1+ω-φ>0 时，||α||>||β||，则||α||<1，系统收敛。而||α||<1，根据(3-7)和(3-8)两式得到：

$$0<\frac{1+\omega-\phi+\sqrt{(1+\omega-\phi)^2-4\omega}}{2}<1$$

推得：φ>0

所以，条件总可以得到满足。因此，当 $\begin{cases}\phi>0\\1+\omega-\phi>0\\(1+\omega-\phi)^2\geq 4\omega\end{cases}$ 成立时，粒子的轨迹收敛。

(b).当 1+ω-φ<0 时，||α||≤||β||，即||β||<1，系统收敛。而||β||<1，根据(3-7)式得到：

$$0<|\frac{1+\omega-\phi-\sqrt{(1+\omega-\phi)^2-4\omega}}{2}|<1$$

推得：

$$\omega>\frac{1}{2}\phi-1$$

这个式子与文献[18]的 89 页结果(3-21)式一致。

因此，当 $\begin{cases}\omega>\frac{1}{2}\phi-1\\1+\omega-\phi<0\\(1+\omega-\phi)^2\geq 4\omega\end{cases}$ 成立时，粒子的轨迹收敛。

根据上面的推导，粒子群的轨迹收敛条件为(3-15)、(3-16)和(3-17)式，即粒子轨迹收敛的条件如下：

$$\begin{cases}\omega<1\\(1+\omega-\phi)^2<4\omega\end{cases} \tag{3-15}$$

或

$$\begin{cases}\phi>0\\1+\omega-\phi>0\\(1+\omega-\phi)^2\geq 4\omega\end{cases} \tag{3-16}$$

或

$$\begin{cases}\omega>\frac{1}{2}\phi-1\\1+\omega-\phi<0\\(1+\omega-\phi)^2\geq 4\omega\end{cases} \tag{3-17}$$

对于 $(1+\omega-\phi)^2=4\omega$ 式，其解为：

$\omega=1+\phi\pm 2\sqrt{\phi}$，对于 $\omega=1+\phi-2\sqrt{\phi}$，其曲线如图 3-1，结合直线 $\omega=\frac{1}{2}\phi-1$ 和直线 $1+\omega-\phi=0$，三条线组合图 3-1。

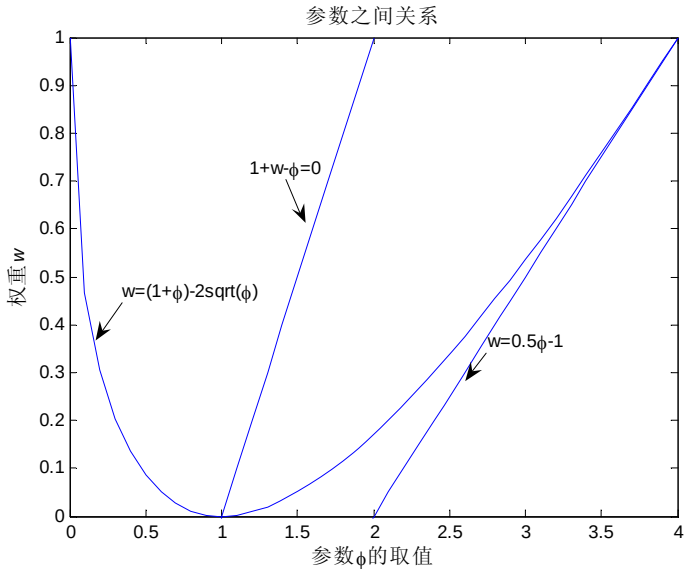

图 3-1 参数权重 w 与量 ϕ 之间的关系图

从上面的分析并结合图 3-1 来看：当 0<ω<1，0<ϕ<4 时，收敛参数范围为：

$\omega > \frac{1}{2}\phi - 1$。所以，当不考虑随机性且个体最优点和群体全局点为常数时，PS0 算法收敛条件：

$$\begin{cases} 0 < \omega < 1 \\ 0 < \phi < 4 \\ \omega > \frac{1}{2}\phi - 1 \end{cases} \tag{3-18}$$

3.3.2 速度分析方法

同样，假设$\phi_1 = c_1 \cdot rand()$，$\phi_2 = c_2 \cdot rand()$，$\phi = \phi_1 + \phi_2$，如果 p，p_g，ϕ_1和ϕ_2保持不变的值，则由（3-1）式和（3-2）式消去速度相关项可以得到下式：

$$v(t+1) - (1 + w - \varphi)v(t) + wv(t-1) = 0 \tag{3-19}$$

(3-19)式中系数矩阵的特征多项式为：

$$\lambda^2 - (1 + w - \varphi)\lambda + w = 0 \tag{3-20}$$

方程（3-20）中存在两个特征根λ_2, λ_3与(3-7)式的结果一样，即有：

$$\lambda_2 = \alpha = \frac{1 + w - \phi + \sqrt{(1 + w - \phi)^2 - 4w}}{2}$$

$$\lambda_3 = \beta = \frac{1 + w - \phi - \sqrt{(1 + w - \phi)^2 - 4w}}{2}$$

由此可以得到：

$$v(t) = q_1\alpha^t + q_2\beta^t \tag{3-21}$$

根据(3-21)式，速度$v(t)$收敛条件是$\max(\|\alpha\|, \|\beta\|) \le 1$。而上节推导保证粒子轨迹收敛的条件也是$\max(\|\alpha\|, \|\beta\|) \le 1$。因此，粒子轨迹收敛分析的结果与粒子速度收敛分析的结果是等价的，速度收敛条件也是(3-18)式。但速度收敛形式为：

$$\lim_{t\to\infty} v(t) = \lim_{t\to\infty}(q_1\alpha^k + q_2\beta^k)$$

上式意味速度$v(t)$收敛于 0。

下面对粒子运行过程作一个仿真，有一个直观理解。给出几个不同参数组合，验证粒子轨迹与速度变化与本章推导的结论符合。

参数组 1：ω=0.8，ϕ_1=1.25，ϕ_2=1.25，参数满足收敛条件(3-18)式。初始条件为 p=5，p_g=8，x(0)=1，v(0)=2。粒子的轨迹与速度的变化过程如图 3-2 及图 3-3 所示，粒子轨迹最终收敛于 6.5，而速度收敛于 0。

参数组 2：ω=0.5，ϕ_1=1.5，ϕ_2=1.5，参数满足 $\omega = \frac{1}{2}\phi - 1$ ，刚好在收敛边界。初始条件为 p=5，p_g=8，x(0)=1，v(0)=2。粒子的轨迹与速度的变化过程如图 3-4 及图 3-5 所示，粒子轨迹和速度发散。

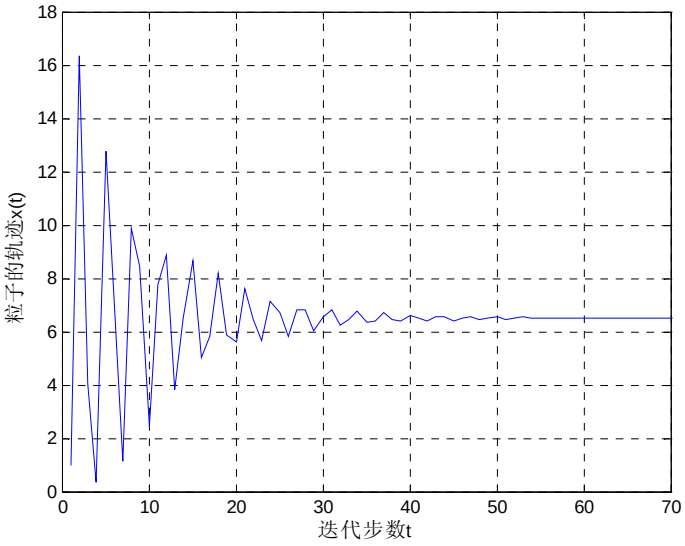

图 3-2 参数组 1 的粒子轨迹 $x(t)$迭代变化情况

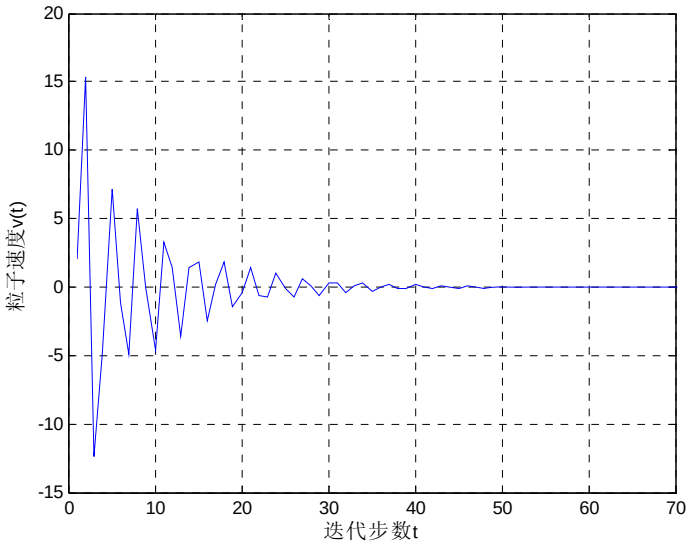

图 3-3 参数组 1 的速度 $v(t)$迭代变化情况

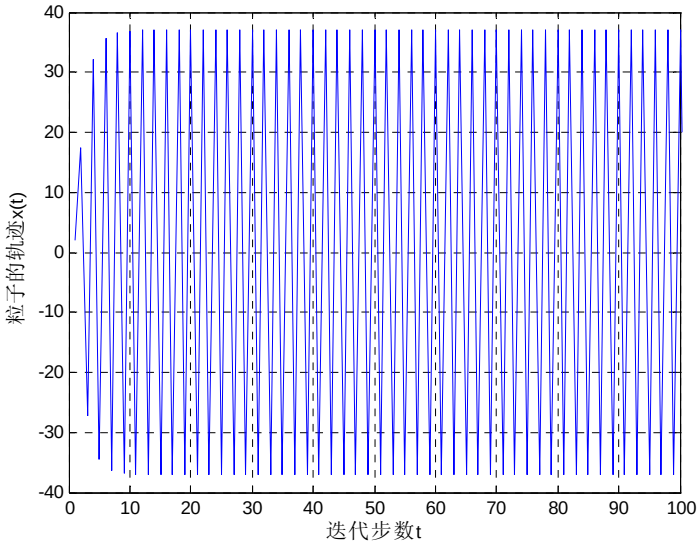

图 3-4 参数组 2 的粒子轨迹 $x(t)$迭代变化情况

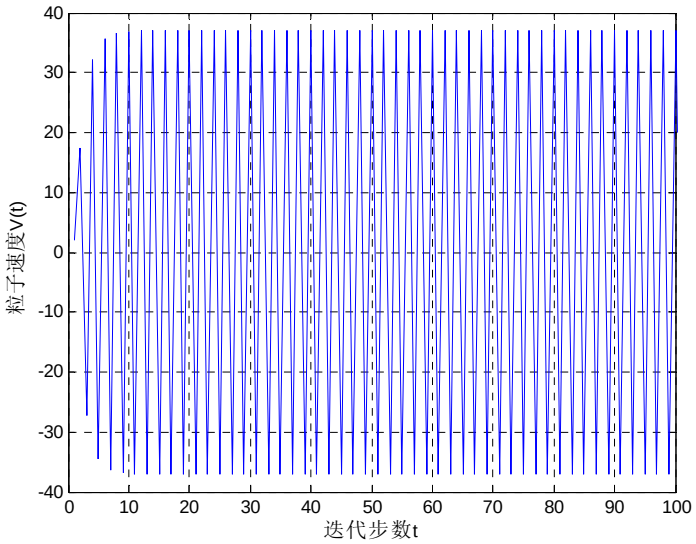

图 3-5 参数组 2 的速度 $v(t)$迭代变化情况

这节需要强调收敛条件(3-18)式成立的假设：没有随机性且粒子运动过程中个体最优点 p 和群体最优点 p_g 固定变化。这个假设条件约束粒子的运动，且与 PSO 算法真实运动有差距。因此，当 PSO 算法存在随机性且个体最优位置与群体最优位置动态变化时，需要进一步分析其运行轨迹。

3.3.3 凸集与凸函数

凸集与凸函数是优化理论中常用到的基本概念，它对于优化理论的证明和算法研究非常重要。下面简要介绍其概念：

凸集的定义：设 S 为 n 维欧氏空间 E_n 中的一个集合。若对 S 中的任意两点，它们的连线仍然属于 S，即：

对 x_1，$x_2 \in S$ 及 $\lambda x_1 + (1-\lambda)x_2 \in S$，$\lambda \in [0, 1]$

则称 S 为凸集。

结论(3-14)式是 PSO 算法在 p 与 p_g 不变的条件下获得的，即(3-3)式对结论(3-14)式并没有影响。如果考虑(3-3)式，粒子的运行轨迹是怎样？当 PSO 算法满足条件（3-3）式且优化单峰函数时，对粒子的运行轨迹进行分析。首先，给出单峰函数的定义：

定义：设 f 是定义在闭凸集 S 上的实函数，x_0 是 f 在闭凸集 S 上的极小点，并且对任意的 x_1，$x_2 \in D$，$x_1 < x_2$，有

当 $x_2 \le x_0$ 时，$f(x_1) > f(x_2)$

当 $x_0 \le x_1$ 时，$f(x_2) > f(x_1)$

则称 f 是在闭凸集 S 上的单峰函数。显然不满足上述定义的函数叫多峰函数。

一维单峰函数的例子如图 3-6，多峰函数的例子如图 3-7。根据图形，能直观理

解单峰函数和多峰函数的意义。

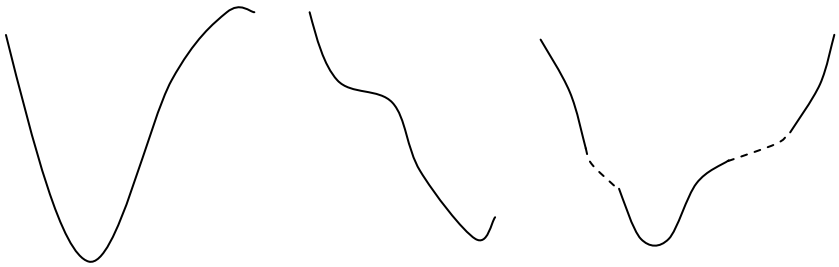

图 3-6 单峰函数示意图

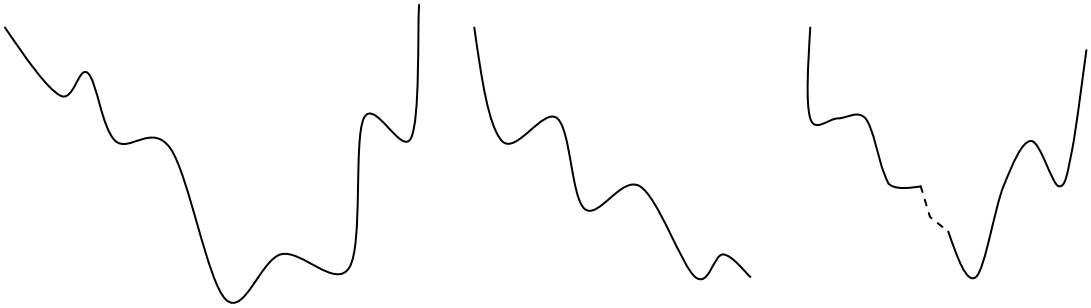

图 3-7 多峰函数示意图

根据定义，单峰函数 $f(x)$的最小值是 $f(x_0)$，没有其他的局部最小值。对单峰函数，从优化理论来说，有下列性质：

引理 3.1：当 f 是一个峰函数时，对任意 x_1，$x_2 \in S$，有下式成立：

$$f((1-a)x_1 + ax_2) < \max(f(x_1), f(x_2))$$

其中 $a \in [0, 1]$。

根据定义，单峰函数在 S 上的局部极小点，也是在 S 上的整体极小点。而对于多峰函数可以看作若干个单峰函数组成。

3.4 p 和 p_g 变化对粒子轨迹的影响

p 和 p_g 分别是粒子个体最优位置与群体最优位置，也代表粒子的“自身经验”和“社会经验”，PSO 算法的粒子通过这个两个位置实现相互协作。当 p 和 p_g 发生变化时，PSO 算法的粒子轨迹如何变化？当粒子位置更新时没有随机性，而 p 和 p_g 发生变化时，PSO 算法的粒子轨迹如何收敛？

由于 p 和 p_g 随迭代步数 t 发生变化，用 $p(t)$，$p_g(t)$表示在迭代步为 t 时的粒子个体最优位置和群体最优位置。

$p(t)$，$p_g(t)$的变化过程分别按下列两式的规律进行：

$$p(t+1)=\begin{cases}x(t) & if\ f(x(t))\geq f(p(t))\\ p(t) & if\ f(x(t))<f(p(t))\end{cases} \tag{3-22}$$

$$p_g(t+1)=\begin{cases}p_g(t) & if\ f(p(t))\geq f(p_g(t))\\ p(t) & if\ f(p(t))<f(p_g(t))\end{cases} \tag{3-23}$$

其中，$f(x)$是优化问题的目标函数。

当 $p(t)$，$p_g(t)$变化时，$(1-a)p(t)+ap_g(t)$也是变化，如果 $\lim_{x\to\infty}p(t)$ 、$\lim_{x\to\infty}p_g(t)$ 存在，则$\lim_{t\to\infty}((1-a)p(t)+ap_g(t))$也存在。那么$\lim_{x\to\infty}p(t)$、$\lim_{x\to\infty}p_g(t)$是否存在？并且粒子的轨迹是否按 $\lim_{t\to\infty}x(t)=\lim_{t\to\infty}(1-a)p(t)+ap_g(t)$收敛？

定义：对于函数 f，S 为函数 f 的定义域，则集合 $S_v=\{x\mid f(x)=v, x\in S\}$为函数 f 的同 v 值区。

对于函数 f 来说，v^*是最小值，则 S_{v^*}是函数的最小值区。

引理 3-2：$\lim_{x\to\infty}p(t)$ 、$\lim_{x\to\infty}p_g(t)$ 存在。

证明：假设优化问题的搜索空间 S 是有限时，且目标函数 $f(x)$在空间 S 中存在全局最优值。因 $p(t)$，$p_g(t)\in S$，所以 $f(p(t))$与 $f(p_g(t))$存在下界。

根据(3-22)、(3-23)式，数列$\{f(p(t))$，$t=1,2,\ldots,\infty\}$及数列$\{f(p_g(t))$，$t=1,2,\ldots,\infty\}$是一个单调非递增数列且有界。

因此，数列$\{f(p(t))$，$t=1,2,\ldots,\infty\}$及数列$\{f(p_g(t))$，$t=1,2,\ldots,\infty\}$存在极限。

令 $\lim_{t\to\infty}f(p(t))=f_p^*$，$\lim_{t\to\infty}f(p_g(t))=f_g^*$；

则有

$\lim_{t\to\infty}p(t)=S_{f_p^*}=\{x\mid f(x)=f_p^*, x\in S\}$；

$\lim_{t\to\infty}p_g(t)=S_{f_g^*}=\{x\mid f(x)=f_g^*, x\in S\}$；

根据(3-22) 、(3-23)式，$\lim_{x\to\infty}p(t)$和$\lim_{x\to\infty}p_g(t)$ 不可能分别在以上两集合的值间移动，一定是为集合某一固定值。

所以，$\lim_{x\to\infty}p(t)$、$\lim_{x\to\infty}p_g(t)$ 存在。

证毕

事实上，随着迭代步数 t 增加，$f(p(t))$，$f(p_g(t))$将逐步趋于某值，而 $p(t)$，$p_g(t)$也将趋于稳定或停止。此时，完全可以符合（3-14）式推导条件，即没有随机性，$p(t)$，$p_g(t)$也是固定值。所以，这里有下式成立：

$$\lim_{t\to\infty}x(t)=\lim_{t\to\infty}(1-a)p(t)+ap_g(t)=(1-a)p^*+ap_g^* \tag{3-24}$$

这里令$\lim_{t\to\infty}p(t)=p^*$，$\lim_{t\to\infty}p_g(t)=p_g^*$。

引理 3.3: PSO 算法在只去掉随机性条件下，(3-24)式成立。

3.4.1 收敛定理

定理 3.1：如果不考虑随机性，在单峰函数 f 下，若 p_g(t)保持不变，即为固定值 p_g；而 $p(t)$保持变化，即满足(3-22) 式；则 PSO 算法有 $\lim_{t\to\infty} p(t) = p_g$。

证明：因为 $f(p(t)) \geq f(p_g)$，根据（3-3）式可知 $f(p(t)) \geq f(p(t+1))$，所以数列{ $f(p(t))$，t=1,2,…,∞}是一个单调非递增数列且有界。因此，它的极限存在，即数列{ $f(p(t))$，t=1,2,…,∞}存在极限。

由于 PSO 算法对函数 $f(x)$的搜索空间是有界的，并从（3-22）式的含义中可以明白：

如果 $f(x(t)) < f(p(t-1))$, 那么 $p(t)=x(t)$, 否则 $p(t)= p(t-1)$;

根据引理 3.2，数列{ $p(t)$}也存在极限。

令 $\lim_{t\to\infty} p(t) = p^*$，

令 $\lim_{t\to\infty} f(p(t)) = f^*, s_{f^*} = \{x \mid f(x) = f^*, x \in S\}$ (即为函数同 f*区)，则 $\lim_{t\to\infty} p(t) \in s_{f^*}$。

接下来利用反证法：

1). 当 $f^* \neq f(p_g)$时，显然 $\lim_{t\to\infty} p(t) \neq p_g$ ，

又 $\lim_{t\to\infty} p(t) = p^*$ ，则 $p^* \neq p_g$ 。

根据引理 3.3，有

$$\lim_{t\to\infty} x(t) = (1-a)p^* + ap_g \qquad a \in (0, 1)$$

因为函数 $f(x)$是单峰函数，由已知条件可知 $f(p^*) > f(p_g)$，所以根据引理 3.1，可得 $f((1-a)p^* + ap_g) \leq f(p^*)$。再根据 $p(t)$的更新公式，$p(t)$要发生变化，即 $p(t)=(1-a)p^*+ ap_g$。因此，$p(t)$的极限不可能为 p^*，这与 $\lim_{t\to\infty} p(t) = p^*$ 发生矛盾。

所以 $f^* \neq f(p_g)$不成立。

因此 $f^* = f(p_g)$。

2）.当 $f^* = f(p_g)$时，但若 $\lim_{t\to\infty} p(t) \neq p_g$，

同样令 $\lim_{t\to\infty} p(t) = p^*$ ，则 $p^* \neq p_g$ 。

根据引理 3.3，有

$$\lim_{t\to\infty} x(t) = (1-a)p^* + ap_g \qquad a \in (0, 1)$$

因为函数 $f(x)$是单峰函数，且 $f(p^*) < f(p_g)$，所以根据引理 3.1，可得 $f((1-a)p^* + ap_g) \leq f(p^*)$。因此，$p(t)$要发生变化，即 $p(t) = (1-a)p^* + ap_g$。因此，$p(t)$的极限不可能为 p^*，这与前面条件发生矛盾。

由以上 1)和 2)，可知 $\lim_{t\to\infty} p(t) = p_g$ 成立。

证毕

注意：定理 3.1 成立条件如下:

1).目标函数 *f(x)*是单峰函数，其搜索空间有界且为凸集。也就是说，要求目标函数 *f(x)*在其搜索空间 *S* 满足引理 1 的性质，即对任意 $x_1,x_2 \in S$，有

$f((1-a)x_1 + ax_2) < \max(f(x_1), f(x_2))$。

2).要求 $f(p_g) <= f(p(t))$始终满足。由 PSO 算法对 p_g 与 $p(t)$定义，这个条件能够满足。

结合（3-12）式及定理 3.1 的结论，可以得到$\lim_{t\to\infty} x(t) = p_g$。因此有下列结论：

定理 3.2：如果不考虑随机性，在单峰函数 *f* 下，若 p_g(t)保持不变，即为固定值 p_g；而 $p(t)$变化，即满足(3-21)，(3-22)；则 PSO 算法有 $\lim_{t\to\infty} x(t) = p_g$。

证明：根据引理 3.3，有：

$\lim_{t\to\infty} x(t) = (1-a)p(t) + ap_g \quad a \in (0, 1)$;

而根据定理 1，有：

$\lim_{t\to\infty} p(t) = p_g$;

所以，有 $\lim_{t\to\infty} x(t) = p_g$。

证毕

定理 3.2 成立的条件与定理 3.1 成立的条件是相同的。因此，定理 3.2 说明 PSO 算法在简化条件（即，没有随机性，目标函数满足引理 1 性质）下，一定有$\lim_{t\to\infty} x(t) = p_g$。也就是粒子的轨迹一定会收敛群体最优位置。

3.4.2 仿真验证与分析

定理 3.1 及定理 3.2 说明 PSO 算法在不考虑随机性条件下，每一个粒子最终都会收敛于全局最优粒子。这节通过仿真试验验证定理 3.1 和定理 3.2 的有效性。为了更好地设计实验，首先注意两个定理成立的条件，列出如下：

(1).参数ω，ϕ_1，ϕ_2，p_g是固定不变的常数，且满足关系式$(1+\omega - \phi_1-\phi_2) < 4\omega$;

(2).目标函数 *f* 是单峰函数，即满足引理 3.1；

(3).$p(t)$按(3-22)式变化，即，如果 $f(x(t)) < f(p)$，则 $p(t)=x(t)$，但 $f(p_g) <= f(p)$始终满足。

3.4.3.1 实验设计

选择两个基准函数做试验，一个单峰函数，另一个多峰函数。两个函数列出如下：

(1).单峰函数 f_1 是 Spherical 函数：

$$f_1(x)=\sum_{i=1}^{n}x_i^2 \qquad -100<x_i<100$$

最优点为（0,…,0），最优值为 0。

(2).多峰函数 f_2 是 J.D.Schaffer 函数:

$$f_2(x)=\frac{\sin^2\sqrt{\sum_{i=1}^{n}x_i^2}-0.5}{(1+0.001*(\sum_{i=1}^{n}x_i^2))^2}+0.5 \qquad 5.12<x_i<5.12$$

这个函数存在很多局部最优点和局部最优值，当 $\sum_{i=1}^{n}x_i^2=n\pi$，$n\in Z$时，函数都得到局部最优值。全局最优点为（0,…,0），全局最优值为 0。

图 3-8 和图 3-9 分别是 PSO 算法在计算函数 f_1 和 f_2 时粒子轨迹图，粒子轨迹是在满足定理 3.1 和定理 3.2 的条件下所获得的，也是在没有随机性的条件下计算得到的。参数设置是ω=0.5，ϕ_1=1.4，ϕ_2=1.4，这些参数取值满足条件(3-14)式，这个条件可保证粒子的轨迹收敛。为了更好理解实验结果，实验只在两个函数的一维情况下进行的。图中 p_g 的值都设置为 0，可以使 $f(p_g)<f(p)$始终满足。$p(t)$的值按(3-22)式更新，初始值设置如下：速度 $v(0)$为 1，粒子轨迹初始位置 $x(0)$和个体最优初始位置 $p(0)$为相同，但它们的初始值对应不同轨迹图（见图）。

单峰函数 f_1 的粒子轨迹如图 3-8，而多峰函数 f_2 的轨迹如图 3-9。两个图形分别演函数 f_1 和函数 f_2 一维的粒子轨迹运行。在两个图中，粒子轨迹位置 $x(0)$和个体最优位置 $p(0)$的初始设置值三对值，即为π，12 和π+1，其分别对应图中不同的演示图 data1，data2 和 data3。

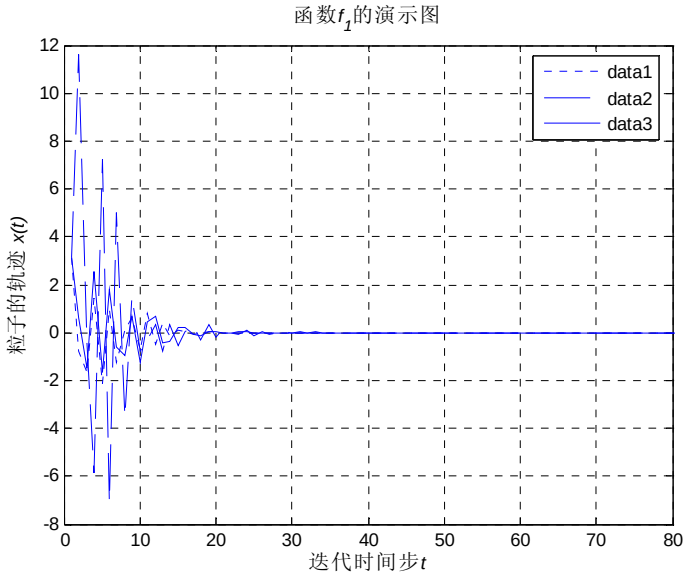

图 3-8 粒子在 PSO 计算单峰函数 f_1 时的轨迹

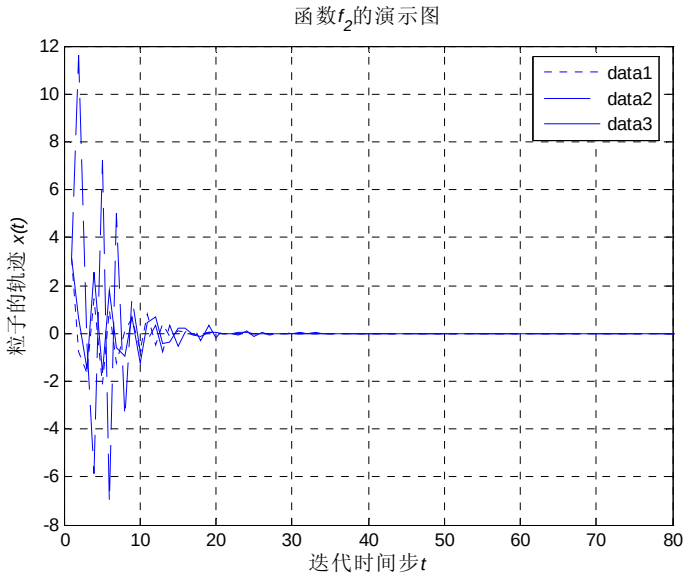

图 3-9 粒子在 PS0 计算多峰函数 f_2 时的轨迹

图 3-8 显示的粒子轨迹图表明：无论初始值 $x(0)$为什么值，粒子的轨迹收敛到单峰函数 f_1 的最优点 p_g。图 3-9 显示的粒子轨迹图表明：计算多峰函数 f_2 的粒子轨迹不一定收敛于最优点 p_g，例如，当 $p(0)=\pi$时，粒子轨迹没有到收敛到最优点 p_g(见图 3-9 中的演示线 data3)。

因此，图 3-9 演示多峰函数 f_2 的粒子轨迹图表明：当 PSO 算法计算多峰函数 $f(x)$ 时，粒子轨迹 $x(t)$可能收敛于 p_g(如 data1 和 data2)，也可能不收敛于 p_g（如 data3），这取决于初始位置 $x(0)$的值。

3.4.3.2 实验结果分析

图 3- 8 演示说明，当 PSO 算法计算单峰函数时，无论初始位置 $x(0)$和个体最优 $v(0)$的初始值是多少，粒子的轨迹都将收敛于粒子群最优点 p_g，即 $\lim_{t\to\infty} x(t) = p_g$。

图 3-9 演示多峰函数的收敛特性。PSO 算法在计算数据 *data3* 时，粒子的轨迹并不收敛于群体最优位置 p_g；而计算数据 *data2* 和 *data3* 时，粒子轨迹收敛于群体最优位置 p_g。这个不同结果在同一个多峰函数产生，只是因为初值 $x(0)$不同。图 3-9 中三个图表明：对于多峰函数，粒子轨迹 $x(t)$可能收敛于 p_g，也可能不收敛于 p_g。

在定理 3.2 的前提条件下，在 PSO 算法优化多峰函数时，粒子轨迹如何移动的？针对单峰函数，图 3-8 的结果验证定理 3. 2 的结论。对于多峰函数，定理 3.2 的结论并不成立，粒子的轨迹并不收敛于 p_g。但是，根据图 3-9，PSO 算法优化多峰函数时，粒子的轨迹也可能收敛于 p_g。因为在定理 3 成立的条件下，当 p 不变且 $p\neq p_g$ 时，$f((1-\alpha)p+\alpha p_g)$ 有可能小于 $f(p)$也有可能大于 $f(p)$，这由 p 坐标值来决定。当没有随机量时，α值固定不变，$(1-\alpha)p+\alpha p_g$ 也是不变的。此时，如果 $f((1-\alpha)p+\alpha p_g) > f(p)$

满足，则 p 不变，$f((1-\alpha)p+ ap_g)>f(p)$始终满足，$p$ 也就永远不变，这时$\lim_{t\to\infty} x(t)=p_g$不成立，这正是图 3-9 中轨迹图 data3 的情境。如果 p 不变时，不会出现 $f((1-\alpha)p+ ap_g)>f(p)$情境，则 $\lim_{t\to\infty} x(t)=p_g$一定会成立，这是图 3-9 中轨迹图 data1 和 data2 的结论。

3.5 PSO 算法的随机量分析

PSO 算法其实是一种随机启发式算法，随机量作用是非常重要，随机量的不确定性使算法具有多样性及全局的探索性。PSO 算法的随机性给算法带来什么影响？下面对此作分析。

定理 3.2 成立的条件是去掉算法的随机量。如果把随机量加进去，则定理 3.2 的α值不是固定不变的，而是随机的。因为 $\alpha=\frac{\phi_1}{\phi_1+\phi_2}$，而$\phi_1=c_1\cdot$rand( ), $\phi_2=c_2\cdot$rand( )，rand( )是区间[0，1]之间的随机数，ϕ_1，ϕ_2 是一个随机数，所以α也是一个随机数，并且是[0，1]之间的随机数。由于 rand()具有统一分布性，α是在区间［0, 1］统一分布的随机数。

上一节的分析去掉随机量，实验显示：当目标函数是多峰函数时，粒子轨迹有时未必会收敛群体全局最优位置 p_g。当算法存在随机量且目标函数是多峰态函数时，粒子轨迹是否会收敛到群体全局最优位置 p_g？这里有下列定理结论：

定理 3.3：PSO 算法在定理 3.2 前提条件下增加随机性，目标函数是多峰函数，且算法中的 p_g 不变，而 $p(t)$更新（即，如果 $f(x(t))<f(p(t))$，则 $p(t)=x(t)$），并且 $f(p_g)<=f(p)$始终满足)，则有 $\lim_{t\to\infty} x(t)=p_g$。

证明：当 $p(t)$变化，$p_g(t)$不变时，且 $p(t)\neq p_g(t)$时，由于α是区间[0，1]之间任意随机数据，所以 $(1-\alpha)p(t)+\alpha p_g(t)$在点 $p(t)$与 $p_g(t)$之间的线段随机采样。而这条线段上的点，总会存在某点 y，其函数值 $f(y)<f(p(t))$，所以 $p(t+1)$取值为 y 而产生变化。显然，此时 $p(t+1)$比前一个 $p(t)$更靠近 $p_g(t)$，在无限的迭代中，会不断出现此种情境，最终 $p(t)$收敛于 $p_g(t)$，进而 $x(t)$收敛于 $p_g(t)$。因此，在随机条件下，即使 PSO 算法求解多峰函数，粒子轨迹也一定收敛于全局最优点 p_g。

证毕

下面仿真实验验证定理 3.3 结论。实验的设置与上节完全一样，只是粒子速度的更新式子添加了随机量，即α为随机量，与标准 PSO 算法一致。其它设置与前面实验相同。实验的多峰目标函数也是上节函数 f_2。实验演示在初始值为不同时，粒子位置 $x(t)$的轨迹变化。实验结果如图 3-10。初始值 $x(0)$=π，12，π+1，与上节取的初始值完全相同。三个初始值分别对应图的轨迹 data1，data2 和 data3。

图 3-10 实验初始数据与图 3-9 是一样的。我们对比两个图形，原在图 3-9 中轨迹不收敛的初始条件在图 3-10 中变成收敛了。这说明增加随机量，会给算法带来收

敛的概率。

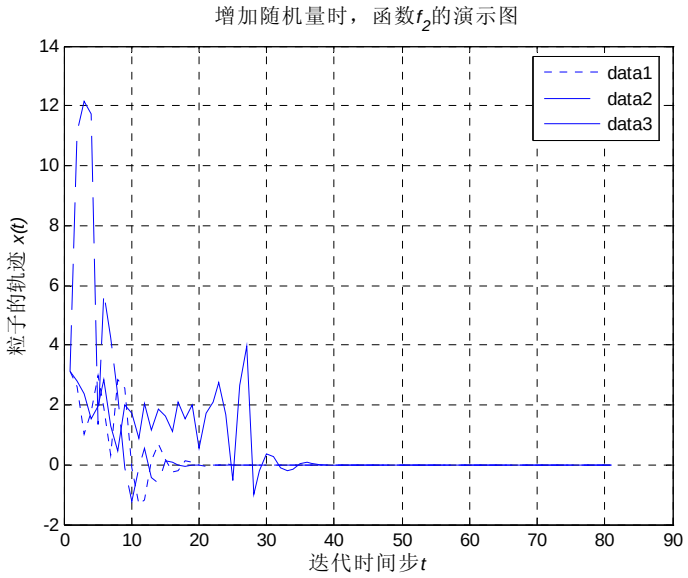

图 3-10 粒子在 PS0 带随机量计算多峰函数 f_2 时的轨迹

实验结果表明，在α固定不变，且 PSO 算法求解多峰函数时，存在不收敛的初始条件；而当α为随机量时，原来不收敛的初始条件却为收敛；而原来收敛的初始条件，这次仍然收敛。因此，实验得到出的结论：在α为随机变化值时，轨迹 $x(t)$总是收敛于 p_g。这验证了定理 3 的结论。

3.6 粒子运动轨迹与算法收敛的关系

$\lim_{t\to\infty} x(t) = p_g$ 说明 PSO 算法的每一个粒子最终收敛聚集到同一点 p_g。这种现象能说明启发式随机算法的探索性及开拓性。所谓的探索性指算法的局部搜索能力，而开拓性指算法的全局搜索能力。从前节的图中可看到，当 PSO 算法参数满足收敛条件时，粒子轨迹开始一段是随机地在空间搜索，这时粒子全局搜索最优点。而随着迭代进行到后期，粒子轨迹收敛到 p_g 附近。在一定时间以后，粒子只能在 p_g 附近探索最好点。

根据收敛性定义，存在某个 T 值，当迭代步数 t>T 时，$x_i(t)\in\{p_g$ 的邻域$\}$。即，当 t>T 时，粒子轨迹 $x_i(t)$在 p_g 的邻域探索，只能找 p_g 邻域中最优点。此时，PSO 算法是一种局部搜索算法。但根据文献[113] 的证明，PSO 算法并不能保证收敛到 p_g 邻域的最优点，即 PSO 算法并不能保证局部收敛，下面通过反例加以说明[113]。

假定在二维空间搜索且只有两个粒子，其中一个粒子 0，也是全局最优粒子。假定符号 $a_1 \ldots a_5$ 和 $p_1 \ldots p_2$ 表示任意常数。当粒子达到下列状态，粒子群将停滞搜索。

$$v_0 = a_1(0,1)$$

$$v_1 = a_2(0,1)$$

$$x_0 = (q_1, q_2)$$
$$x_1 = (q_1,\ q_2) + a_3(0,\ 1)$$
$$p_0 = (q_1, q_2) + a_4(0,1)$$
$$p_1 = (q_1, q_2) + a_5(0,1)$$

当两个粒子处于上面的状态时，根据式(3-1)和(3-2)可知:

$$x_i(t) = (q_1, q_2) + c(t)(0,1)$$

这个式子说明粒子位置处于(q_1,q_3)，q_3＝q_3+$c(t)$是任意值，而 q_1 为不变的值。这表明粒子被限制在二维空间一条直线上运动，如果目标函数的所有局部最优点位置不是(q_1,q_3)形式，那算法永远也搜索不到局部最优点。算法的粒子存在非零概率达到或初始化为状态(q_1,q_3)，因此这例子说明 PSO 算法不是局部收敛算法，因为它不能保证从任意状态下运行到最优位置。但当使用粒子数量多时，处于上面这种状态的概率是非常低的。

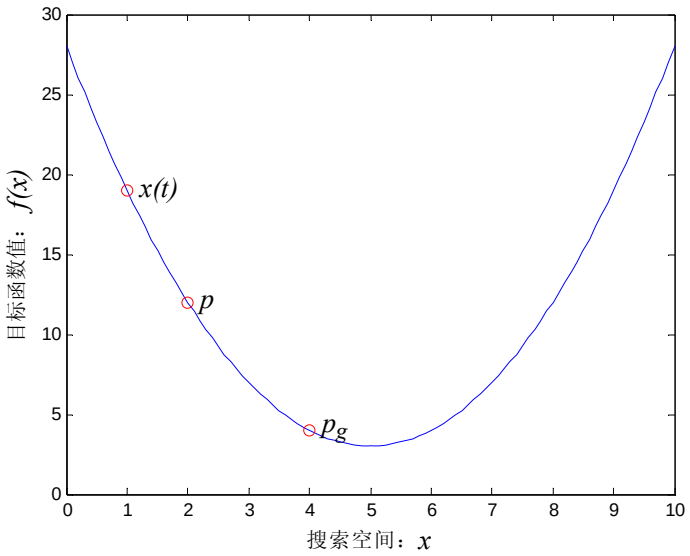

图 3-11 局部搜索时的粒子轨迹特点

当 PSO 算法运行到一定迭代步数 T 以后，某粒子轨迹 *x(t)*处于图 3-11 以上位置，粒子群算法相当于对单峰函数搜索。根据速度迭代公(3-2)式，若速度 $v(t)<p_g-x(t)$，则轨迹 *x(t)*不会改变全局最优点 p_g。如果这个条件在随后迭代步数 *t* 都能得到满足，则轨迹 *x*(t)收敛 p_g，但不会改变 p_g 的值。如果粒子群每一个粒子在迭步 T 以后都满足这种情况，则 p_g 不会收敛到局部最优点。

当 t<T 时，p_g 在动态变化并且在不同的凹区域变化。p_g 的动态变化其实在寻找更好的局部区域。此时，算法在进行全局开拓。

3.7 PSO 参数的选择

PSO 算法的参数选择是为了保证算法收敛。从前面分析可知，参数需要满足公式ω>0.5(ϕ_1+ϕ_2)-1。根据公式，ω越大，越有助于算法收敛，因为式子更容易满足。但根据标准 PSO 算法的性能，权重ω的值线性减少，ω越小，越有助于算法收敛，有助于局部探索。ω越大，越不利于收敛，但有利于算法全局开拓。这似乎有矛盾。这是因为收敛与收敛速率是两个不同的概念。收敛并不意味收敛速度快，选择的参数有利于收敛，但收敛速率不一定更快。

图 3-12 中演示 $max(||\alpha||, ||\beta||)$和ω-0.5($\phi_1$+$\phi_2$)+1 随权重ω变化的情况。这里$\phi$=$\phi_1$+$\phi_2$ 且ϕ取 2 为固定值。因为ϕ_1=c_1·rand( )，ϕ_2=c_2·rand( )，而 c_1，c_2 通常为 2，所以ϕ取 2，实际为其随机变化的期望值。

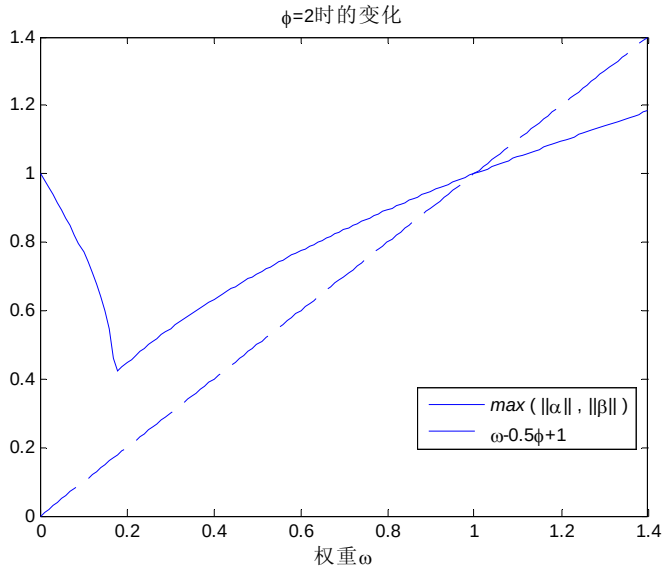

图 3-12 $max(||\alpha||, ||\beta||)$和式ω−0.5ϕ +1 的值随权重 w 的变化

图 3-12 中权重ω的取值使ω>0.5(ϕ_1+ϕ_2)-1 满足。根据前面分析，当ω<1 时，PSO 算法收敛。但 $max(||\alpha||, ||\beta||)$的值有一个折线变化，当它的值大于 1 时，算法不收敛。而它的值小于 1 时，算法收敛，并且其值越小，收敛越快。随着ω变小，至到约为 0.2，$max(||\alpha||, ||\beta||)$变成最小值，即算法收敛速度为最快，但随后 $max(||\alpha||, ||\beta||)$越变越大。因此，这个结论与一些文献的试验要求相符[102] [103]，即对权重ω的取值要求线性递减，最小为 0.2 恰当。在标准 PSO 算法，权重线性递减一般要求为 1.4 到 0.2，不能太小。观察图 3-12，$max(||\alpha||, ||\beta||)$取值的确需要权重线性递减。算法初期需要全局开拓性，不需要收敛，这样有利于全局开拓性；而后期算法需要局部探测性，希望算法收敛速率更快，因此权重的值要变小。但是，权重不能小于 0.2，此时 $max(||\alpha||, ||\beta||)$的值为最小，收敛速率最快，有利于局部探测。

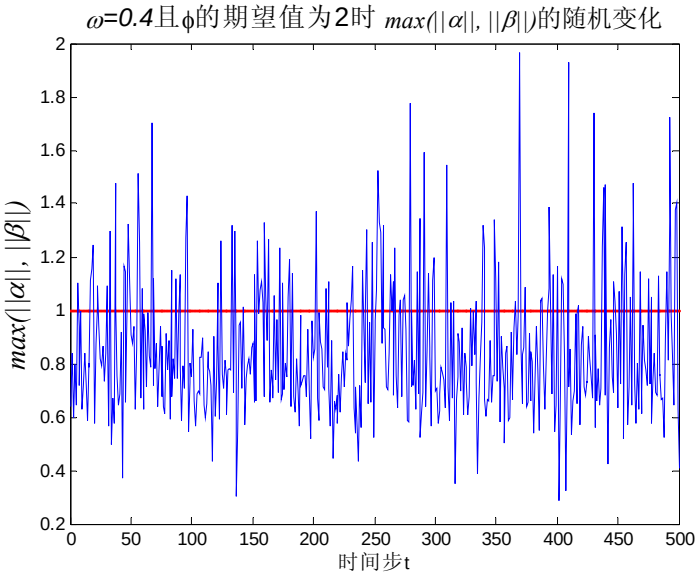

图 3-13 权重ω为 0.4 时，$max(||\alpha||, ||\beta||)$随机变化的情况

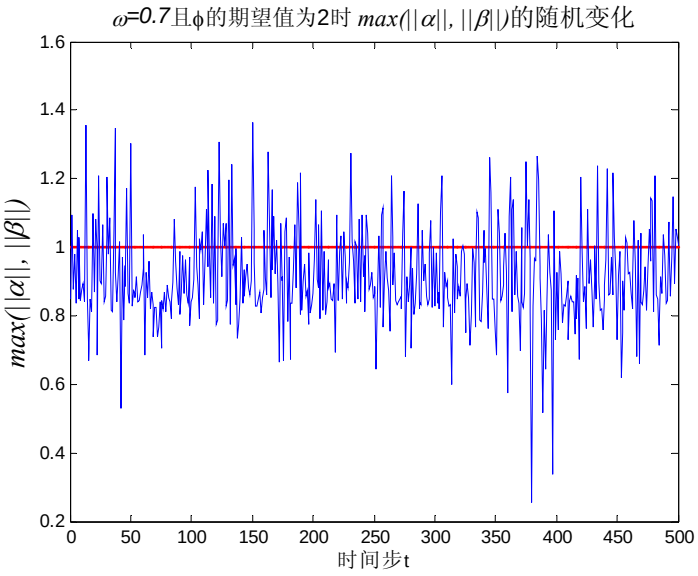

图 3-14 权重ω为 0.7 时，$max(||\alpha||, ||\beta||)$随机变化的情况

上图的分析只考虑ϕ为固定值 2。当考虑ϕ随机值时，$max(||\alpha||, ||\beta||)$的值是如何变化，也可分析随机变量给收敛带来的影响。这里$\phi=\phi_1+\phi_2$，$\phi 1=c_1 \cdot rand()$，$\phi 2=c_2 \cdot rand()$，而 c_1，c_2 为 2。图 3-13、图 3-14 和图 3-15 分别是权重ω为 0.4，0.7 和 1.4 时，$max(||\alpha||, ||\beta||)$各自随机变化的演示图。图中给出一条红(粗)线，用于判断其值小于 1，进而 PSO 算法是否收敛。显然权重ω=1.4 时，$max(||\alpha||, ||\beta||)$基本大于 1，算法完全处于发散，算法 PSO 处于全局开拓、全局搜索状态。而权重ω=0.4，$max(||\alpha||, ||\beta||)$绝对时期小于 1，算法 PSO 大部分时期处于收敛状态。这种不稳定能

够带来一定局部探测性，有利于算法在全局最优位置 p_g 的邻域内搜索最优点，处于局部寻优状态。

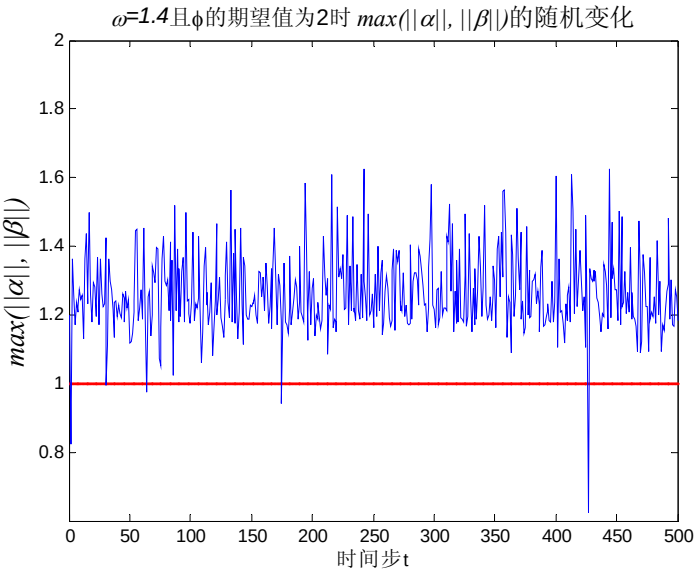

图 3-15 权重ω为 1.4 时，$max(||\alpha||, ||\beta||)$随机变化的情况

3.8 本章小结

本章在已有的理论分析基础上，进一步分析了 PSO 算法收敛性。根据优化理论分析，在不考虑随机量且计算单峰函数时，PSO 算法的粒子轨迹最终会收敛全局最优粒子位置 p_g。而当计算多峰函数时，粒子最终未必收敛全局最优位置 p_g。但如果增加随机性，算法计算的目标函数无论是单峰函数还是多峰函数，粒子都会收敛于最优位置 p_g。本章的分析结果能说明 PSO 算法的随机性及动态性给算法带来的效果。与已有的分析方法相比，本章的分析方法考虑了算法的随机性和粒子最优位置的动态性。

第四章 基于相似度改进粒子群算法

4.1 引言

正如第一章所叙，标准 PSO 算法存在早熟收敛和收敛较慢的缺点。随着算法迭代代数增加，各粒子变得越来越相似，标准 PSO 算法不能对解进行持续地优化，而在局部最优解附近徘徊，陷入局部最优解。目前有很多改进标准 PSO 算法的方法。除了与其他智能优化算法相混合的方法外，改进标准 PSO 算法的方法主要是两种，一种是对算法的权重进行改进，权重大小变化对算法的全局探索与局部探测有很大影响，有人把权重线性变化变成非线性变化或模糊化 [30, 114] 。另一种是控制种群的多样性来提高算法性能的方法，通过解决微粒间冲突 [35] 、引入速度变异和位置变异 [38] 、引入小概率重新初始化.[39] 等方法增加粒子多样性，提高全局搜索能力，避免陷入局部最优。这些改进方法主要提高算法的全局搜索能力，但要在提高全局搜索能力和收敛速度上达到平衡是很困难。

本章基于上一章对标准粒子群算法的分析结果，定义粒子之间相似度概念，分析粒子与全局最优粒子的相似性及其与算法性能的关系，并针对标准 PSO 算法存在的缺点，提出两种改进标准 PSO 算法的方法。一种根据粒子的相似性，提出计算粒子群多样性的聚集度概念，再根据聚集度以概率形式对粒子位置变异，增加粒子群的多样性；另一种方法是根据粒子与全局最优粒子相似度，动态调整每个粒子的权重，使不同的粒子赋予不同的值。

4.2 粒子之间的相似度

根据标准 PSO 算法的性能分析，随着算法迭代运行，粒子变得越来越相似，算法缺少多样性，从而影响算法的全局搜索能力。为了能够根据粒子之间的相似性来分析与改进算法，这里定义粒子间的相似度概念。

定义 4.1:两个粒子 i 和 j 的相似度 $s(i,j)$ 必须满足下列准则:

(1). $s(i,i)=1$.

(2).当 $d(i,j)\to\infty$ 时， $s(i,j)\to 0$;

(3).对任何粒子 i 和 j, 有 $s(i,j)\in[0,1]$。

本章采用如下式子来计算两个粒子i和j的相似度：

$$s(i,j)=\begin{cases}1, & d(i,j)<d_{\min}\\ 1-[\dfrac{d(i,j)}{s_{\max}}]^{\alpha}, & d_{\min}\le d(i,j)<d_{\max}\\ 0, & d(i,j)\ge d_{\max}\end{cases} \tag{4-1}$$

这里，$d(i,j)$ 表示粒子i与j在空间的距离，本章采用*Euclid*距离去计算 $d(i,j)$；参数 $d_{\max}$ 和 $d_{\min}$ 是一个正常数，需要根据目标函数的搜索区域进行确定。α也是一个正的常数，本章为了简单，取其值为1。根据相似度概念的定义，如果两个粒子越靠近，则其粒子越相似，相似度也就越大。因此，相似度概念可度量两个粒子相似程度。

根据上章分析，任何粒子i都会收敛于全局最优粒子p_g，即有 $x_i(t) \to p_g$。随着算法迭代运行，粒子i与全局最优粒子p_g的相似度$s(i,g)$会越来越小，最终会趋于0。相似度$s(i,g)$是本章一个关键信息量。

4.3 基于相似度变异的粒子群算法

4.3.1 多样性控制方法

为了避免算法的过早收敛，一些研究者提出了通过控制种群多样性提高算法总体性能的方法。Jacques Riget 和 Jakob S Vesterstorm 设计了一种以标准 PSO 为基础的，通过多样性度量控制种群特征 [34]，从而实现粒子间吸引和排斥的平衡以避免算法早熟的方法(简称为 NPSO)。这种方法在原有算法粒子位置更新的相互吸引过程之后又引入了一个排斥过程，也就是吸引的逆过程，即采用如下所示公式：

$$v_{id} = wv_{id} - c_1\text{rand}()(p_{id} - x_{id}) - c_2\text{rand}()(p_{gd} - x_{id}) \tag{4-1}$$

这种逆变过程在一定程度上抑制了吸引过程导致的系统多样性的下降。如果系统多样性下降至某个预定的指标，则将算法切换到互斥过程以增加粒子的多样性。当互斥过程使多样性恢复到预定的水平时，结束互斥操作继续标准算法运行。其中所需的多样性度量标准定义如下：

$$diversity(m) = \frac{1}{mL}\sum_{i=1}^{m}\sqrt{\sum_{d=1}^{D}(x_{id} - x_d)^2} \tag{4-2}$$

其中，m 是种群的规模大小，L 是搜索空间的最大对角线长度，D 是问题的维数，x_{ij} 是第 i 个粒子的第 d 维值，x_d 是所有粒子第 d 维的平均值。

为了解决标准PSO算法的早熟收敛问题，DPSO算法[38] 引入了随机变异操作，使微粒有更大几率逃出局部极小点，保持种群的多样性，公式的描述如下:

$$If \quad (rand() < c) \quad Then \quad x_{id} = random(l_d, u_d) \tag{4-3}$$

c为范围[0,1]内的噪声因子, $random(l_d, u_d)$为l_d和u_d之间的随机数,而l_d和u_d为位置分量的取值范围。

本节提出一种多样性度量方法，根据多样性的度量来控制种群的多样性。由粒子间的相似度，计算粒子群的聚集度，再根据聚集度以概率形式对粒子位置变异，增加粒子群的多样性。为了叙述方便，把本节提出的 PSO 算法简称为 SPSO。

4.3.2 粒子群的聚集度

改进进化算法早熟收敛的缺点的一个重要思想是减少相似个体的数量，以尽可能地保持群体的多样性。粒子群优化算法无论早熟收敛还是全局收敛，粒子群算法中的粒子都会出现“聚集”现象。根据第三章的分析，假设 PSO 算法的群体最优粒子 p_g 和个体最优位置 p_i 在进化中保持不变，当参数满足条件$\omega>0.5(\phi_1+\phi_2)-1$ 时，则 PSO 算法的粒子位置 $x_i(t)$ 收敛于 p_g 与 p_i 的加权中心，即有：

$$x_i(t) \to \frac{\phi_1 p_i + \phi_2 p_g}{\phi}$$

其中 $\phi = c_1 + c_2, \phi_1 = c_1 r_1, \phi_2 = c_2 r_2$；$c_1$，$c_2$ 和 r_1，r_2 的含义见(3-1)式和(3-2)式。

只要参数满足条件，PSO 算法的收敛是完全保证的。根据第三章的分析，当 p_i，p_g 不断变化且算法保持随机性时，$x_i(t) \to p_g$ 一定成立。这个结论表明，随着迭代运行，标准 PSO 算法会越来越聚集在一起，粒子最终都向最优点粒子 p_g 靠近。如果粒子群聚集，最终会集到最优粒子 p_g 附近。因此，用相似度度量粒子群的聚集程度，计算整个粒子群与最优粒子 p_g 的平均相似度，量化粒子群的聚集度。因此，构造如下定义:

定义 4.2：第 t 代粒子群的聚集度 $C(t)$为

$$C(t) = \frac{1}{m}\sum_{i=1}^{m} s(i,g) \tag{4-4}$$

这里 m 是种群的规模数，$s(i,g)$ 表示粒子 i 与当前群体最优粒子 p_g 的相似度，$C(t)$可度量第 t 代粒子群的聚集度。

根据聚集度的含义，$C(t)$能量化粒子群的多样性。很显然，其值越小，粒子的多样性越低。(4-4)式与(4-2)式对比，(4-4)式计算各粒子与全局最优粒子的平均相似度，而(4-2)式计算粒子与群体的平均位置的距离量化多样性。相比 (4-2) 式， (4-4) 式有第三章的理论为基础，比较合理；此外，式（4-4）不需要事先计算群体的平均位置，省了一个计算步骤。

4.3.3 基于相似度的变异

如果粒子群都收敛到目前的群体最优粒子 p_g，PSO 的进化随着迭代而会停滞，缺乏多样性。当所有粒子停在最优粒子 p_g 位置时，它们没有机会逃逸。如果这时 p_g 只是局部最优点，则粒子群没有机会再去搜索其它区域，算法也不能够求解目标函数的全局最优解。为了保持群体多样性，当粒子聚在最优位置 p_g 附近时，设计一种方法：让粒子 i 位置 x_i 按群体聚集度 $c(t)$和其与最优粒子 p_g 的相似度 $s(i, g)$随机变异。所以，设计下列式子：

$$\begin{aligned} &if \quad rand() < \alpha * c(t) * s(i,g) \\ &then \quad x_{id} = random(l_d, u_d) \end{aligned} \tag{4-5}$$

这里，α 是固定常数；$random(l_d, u_d)$为l_d和u_d之间的随机数，而l_d和u_d为位置分量的取值范围，其与(4-3)式的含义相同。

分析(4-5)式：当粒子群的聚集度 $C(t)$越大或粒子与最优粒子 g 相似度 $s(i,g)$越大时，粒子随机变异的概率也越大，增加粒子多样性的机会也越大。为了平衡全局搜索能力与局部搜索能力，参数α 随着迭代代数增加而线性减少，这样有利于算法早期提高全局搜索能力，晚期保持局部的探测能力。

4.3.4 算法的过程描述和时间性分析

为了分析算法时间性，根据前面的叙述，SPSO算法的过程描述如下：

SPS0算法过程描述:

设微粒数为n，最大迭代数为t_{max}。

Step1：初始化：随机产生n个粒子位置及其初始速度；

Step2：评价每个粒子的适应值；

Step3：确定迄今为止每个粒子找到的最好位置p_i；

Step4：确定迄今为止整个群体找到的最好位置P_g ；

Step5：利用 (3-1) 式、(3-2) 式重新计算粒子的速度和位置；

Step6：根据（4-1）式计算每个微粒与最优粒子P_g 的相似度，并根据(4-4) 式计算粒子群的聚集度，再根据 (4-5) 式依概率随机变异粒子位置；

Step7：若条件没有满足且未达最大迭代t_{max}，则转向Step2。

在本算法流程中，第 6 步多了一步计算聚集度并用公式(4-5)变异微粒的位置的工作，但只是在算法的内循环中线性增加一步计算。标准 PSO 算法的时间复杂度为 $O(n*dim*t_{max})$，而 SPSO 算法的时间复杂度仍为 $O(n*dim*t_{max})$。因此，SPSO 算法并没有增加标准 PSO 算法的时间复杂性。

与标准 PSO 算法相比，SPSO 算法只增加了步骤 6。如果(4-5)式的参数α为 0，则 SPSO 算法与标准 PSO 算法等价的。(4-5)式的目的是增加多样性，参数α的值越大，变异的概率也就越大，粒子群的多样性也越强。因此，参数α的值越大，越有利于算法全局探测，而越小却有利于算法局部探索。所以，这里采用参数α的值随迭代代数线性递减的策略，最终参数α为 0。这意味参数α早期比较大，而晚期较小，最后为 0，算法最后变成标准 PSO 算法。因此，参数α只需确定最大值，具体的值在下一节通过实验分析确定。

4.3.5 实验与分析

1 .基准测试函数

为了验证本章算法（SPSO）的有效性，分别采用标准 PSO 算法、NPSO 算法、DPSO 算法和 SPSO 算法对下列函数的最小化问题进行实验计算并进行对比。为了检验对多峰函数改进效果，选择三个多峰函数进行实验，三个函数分别列出如下：

(1) 函数 f_1 是 *J.D.Schaffer* 函数:

$$f_1(x)=\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{(1+0.001*(x_1^2+x_2^2))^2}+0.5 \qquad -5.12<x_1,x_2<5.12$$

函数 f_1 在距全局最优点大约 3.14 范围附近有许多局部最优点，其函数强烈振荡，一般算法难以得到最优解。

(2) 函数 f_2 是 *Rastrigrin* 函数

$$f_2(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10] \qquad -100<x_i<100$$

(3).函数 f_3 是 *Griewan*k 函数:

$$f_3(x)=\tfrac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1 \qquad -100<x_i<100$$

以上三个函数的维数、可行空间、函数的目标值及函数最小值列于表 4-1:

表 4-1 测试函数的基本特性

函数	维数	解空间	函数最小值	函数目标值
f_1	2	$[-5.12,5.12]^2$	0	10^{-3}
f_2	30	$[-5.12,5.12]^n$	0	50
f_3	30	$[-600,600]^n$	0	0.1

每个试验函数都重复计算 100 次，每次试验迭代计算的代数为 1000 代。对标准 PSO 算、DPSO 算法、NPSO 算法及 SPSO 算法，参数 c_1 和 c_2 设置为 1.5。权重 w 线性递减，线性变化从 1.2 变化到 0.4。对于每一个算法，粒子群数设置为 20，最大速度 $v_{\max}$ 和最小速度 v_{min} 分别设置为边界区域上下限的一半。而对于(4-5)式的参数α，将在下小节根据实验进行确定。

2 参数α的确定

在(4-5)式中，新引进的参数α对于算法 SPSO 的性能很重要。这里用试验来确定参数α的合适值，让参数α赋予区间[0，6]之间的不同值，然后计算三个函数的平均

适用值和 100 次运行成功率。其实验结果列于表 4-2。

根据表 4-2 中的数据，如果参数α为 3，则 SPSO 算法的性能相对较好。所以，在下列实验中，SPSO 算法将采用参数α的值从 3 到 0 的随着进化代数线性递减的方法。

表 4-2 参数α在不同的值下，算出函数值

参数最	函数 J.D. Schaffer		函数 Rastrigin		函数 Griewank	
大值α	平均适用值	成功率	平均适用值	成功率	平均适用值	成功率
0.5	0.0036	66	79.736	14	0.147	55
1	0.0029	76	75.274	16	0.0145	54
2	0.0025	76	55.252	26	0.0136	58
3	0.0018	84	46.618	33	0.0132	63
4	0.0021	82	48.874	30	0.0122	70
5	0.0036	68	66.022	23	0.0146	53
6	0.0037	66	79.538	22	0.1670	49

3. 几种算法的试验比较

表 4-3 PSO、DPSO、NPSO 与 SPSO 的比较

函数	算法	平均适用值	成功率	平均收敛代数.
Schaffer	PSO	0.0045	74	426.2647
	DPSO	0.0023	80	576.7382
	NPSO	0.0032	76	525.8725
	SPSO	0.0014	88	622.7500
Rastrigin	PSO	83.9446	11	542.9091
	DPSO	54.3366	27	645.4693
	NPSO	64.8749	26	614.8851
	SPSO	46.6180	33	711.5000
Griewank	PSO	0.216	2	616.4740
	DPSO	0.0165	51	715.5825
	NPSO	0.0363	34	678.4735
	SPSO	0.0132	63	872.5900

为了比较本章提出的 SPSO 算法与 PSO、DPSO、NPSO 算法性能，分别用每种

算法计算三个函数 100 次，然后得出其各自的平均适用值、成功率、平均收敛代数，其结果列于表 4-3 中。

比较表 4-3 的数据，观察四种算法计算三个函数的值，SPSO 算法性能超出其他三个算法，只有平均收敛代数，SPSO 算法更大。这说明算法在寻找最优值时，从收敛到最优点的成功率来看，本章的 SPSO 算法一般比标准 PSO 算法要好。但其收敛时间要更长，这也是用时间换来的效果。从这点说明，增加了多样性，算法的收敛速度就会降低；而 SPSO 算法的变异加强在解空间的全局搜索能力，因此，其收敛成功率好于标准 PSO 算法，而且其性能要好其他两种改进算法。

为了进一步比较两个算法的性能，在相同的参数设置下，对多峰函数 f_1 进行如下实验：每隔增加 25 代进化代数时，记录 50 次计算平均最优解和平均收敛次数，即在误差为 0.001 时，50 次计算的收敛次数。测试的最大进化代数为 500，其结果如图 4-1 和图 4-2 所示。

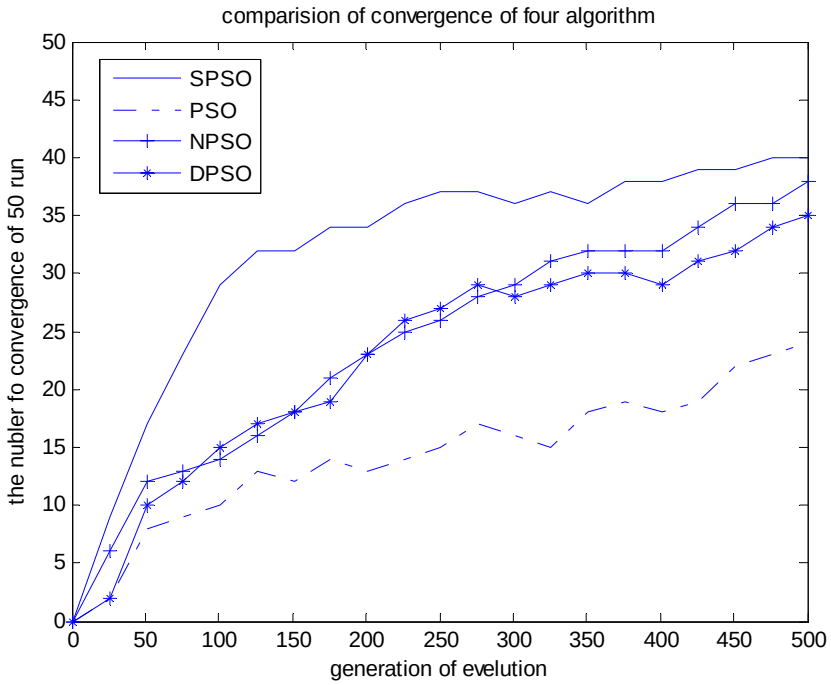

图 4-1 f_1 随进化代数的收敛变化

从图 4-1 中可以看出，SPSO 的收敛次数无论在多少个进化代数情况下，其收敛比例比另外三个算法要高。对于一个比较复杂的多峰函数来说，SPSO 算法平收敛率都比标准 PSO 算法要高。从图 4-2 可以看出，其 SPSO 算法的最优值在不同代下，其所求出最优点要比另外三个算法计算效果明显提高，速度更快。表 4-3 数据也显示，SPSO 的平均收敛代数和收敛率都比另外三个算法要好；这是由基于相似度和聚集度的粒子位置变异，有利于粒子的全局探索能力，提高算法的收敛率。所以在多峰函

数时，新算法不容易错过早期就找到的最优点，也不会使函数很容易在局部最优点收敛。所以，本算法针对多峰函数时有较好的效果。

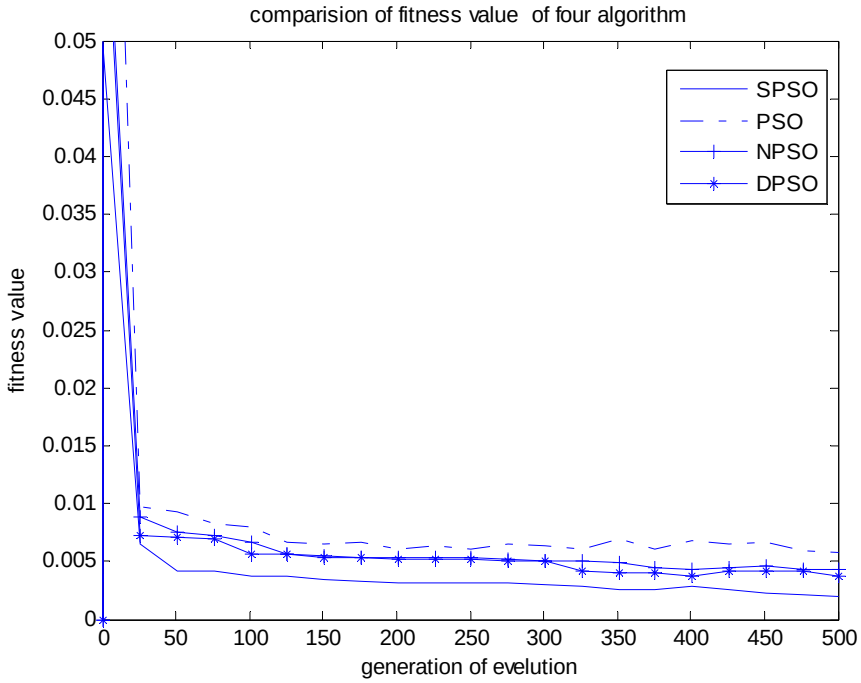

图 4-2 f_1 随进化代数的最优值变化

本节定义粒子之间的相似度，利用这个概念计算每个时刻粒子群的聚集度，根据聚集度，使微粒的位置随机地发生变异，增加粒子群的多样性，增强粒子群算法的全局搜索能力。仿真结果表明，本节提出的粒子群算法在求多峰函数时，算法收敛效果与收敛速度都有所提高。

4.4 基于相似度权重动态调整的粒子群算法

4.4.1 权重的分析与改进

PSO 算法的权重 w 影响算法的局部收敛性与全局探索性，起到平衡这两个搜索能力的作用。因此，该参数对算法的性能产生影响。大的权重有利于算法对搜索区域的全局探索，小的权重有利于算法对搜索区域的局部搜索。

文献[102] 提出权重自适应调整的策略，即随着迭代的进行，线性减小权重w的值。使算法更好地控制探索（exploration）与开发(exploitation)之间的关系。算法在迭代初期探索能力较强，可以不断搜索新的区域；而晚期收敛能力逐渐增强，算法在可能的最优解周围精细搜索，加快收敛速度。

文献[30] 对权重进行改进，把权重的线性变化改为非线性变化，引入权重变化公式如下：

$$w_t=\{(t_{max}-t)^m/t_{max}^m\}(w_{max}-w_{min})+w_{min}$$

这里，w_{max}是开始运行时初始化的权重，w_{min}是最后运行的权重，t_{max}是运行的最大迭代次数，t是当前的迭代次数，w_t是当前的权重，m是非线性调整因子。显然，当m不等于1时，权重是非线性变化。

*Shi.Y*等提出了一种用模糊规则动态调整w的方法 [28] ，通过对当前最好性能评价(*CBPE*)和当前惯性权重制定相应的隶属度函数和模糊推理规则，确定惯性权重w的增量。*CBPE*测量的是算法找到的最好候选解的性能。由于不同的优化问题有不同的性能评价值范围，所以，为了让该模糊系统有广泛的适用性，通常使用标准化的*CBPE(NCBPE)*。假定优化问题为最小化问题，则

$$NCBPE=\frac{CBPE-CBPE_{\min}}{CBPE_{\max}-CBPE_{\min}}$$

其中，$CBPE_{min}$为估计的(或实际的)最小值，而$CBPE_{max}$为非优CBPE，任何CBPE值大于或等于$CBPE_{max}$的解都是最小化问题所不能接受的解，实验结果表明，与w线性减小的策略相比，模糊自应策略具有更好的性能。

4.4.2 动态粒子群算法的基本思想

正如前面所叙，在标准PSO算法[3]中，(3-1)式的惯性权重w采用迭代线性递减的方法，早期选择很大的惯性权重w值，晚期则选择较小的惯性权重w值。虽然，这符合早期需要粒子的探索能力，而晚期需要保持其收敛性的要求；但是，这也有不利的一面，如果早期找到全局最优点，则其惯性权重过大有可能跳出这个最优点，而不在邻域探索，从而降低最优点的搜寻能力。这节提出一种基于相似度权重动态调

整的粒子群算法，简称DPSO。

在PSO进行优化求解时，群体最优粒子p_g附近很有可能存在真正的全局最优解。对当前群体最优粒子p_g来说，速度公式的后两项为0，速度更新公式变为$v_{gd}=w*v_{gd}$。如果惯性权重w很大，则最优粒子p_g有可能跳出其的邻域范围，但问题的最优解真正当前最优解p_g的邻域范围内的可能性很大，希望在此区域精细搜索。从(3-1)式右边三项可以发现，越靠近最优粒子p_g，其飞行速度越依赖惯性权重w。为了不错过每一次最优解p_g的邻域，让靠近p_g 的粒子飞行速度很小，即与当前最优粒子p_g相似度越高的粒子的权重w越小，使这些粒子在P_g领域范围进行精细探索，而不承担全局范围的探测。因此，本节设计一种惯性权重w随各粒子与当前群体最优粒子p_g相似度不同而动态变化的粒子群算法。不同粒子的惯性权重w的值不仅随迭代次数增加而递减，并且应该与当前最优粒子p_g的相似度大小而递减。

设$s(i,g)$表示第i个粒子与当前最优粒子p_g的相似度，设置两个有关距离的参数：最大距离d_{max}和最小距离d_{min}，按公式(5)计算相似度$s(i,g)$。当相似度$s(i,g)$为0时，粒子i的权重w_i为最大权重w_{max}，而当相似度$s(i,g)$为1时，粒子i的权重 w_i 为最小权重w_{min}，而当相似度 $s(i,g)$ 处于(0,1)范围之内时，其权重 w_i 应该随相似度而单调递减，其计算公式：

$$w_i = w_{\max} - s(i,g)(w_{\max} - w_{\min}) \tag{4-6}$$

$$w_i = w_{\min} + (w_i - w_{\min}) * \frac{t_{\max} - t}{t_{\max}} \tag{4-7}$$

(4-6)式和(4-7)式就是粒子i的权重计算方法。(4-7) 式使权重随迭代代数增加而线性减小。

4.4.3 DPSO 算法描述与分析

整个算法的过描述如下：

算法过程描述:

设微粒数为m，最大迭代数为t_{max}。

Step1：初始化，随机产生n个粒子位置及其初始速度；

Step2：评价每个粒子的适应值；

Step3：确定迄今为止每个粒子找到的最好位置p_i；

Step4：确定迄今为止整个群体找到的最好位置p_g；

Step5：根据式（4-1）计算每个微粒与最优粒子p_g的相似度，并根据公式(4-6)和公式(4-7)计算此粒子的权重；

Step6：利用公式(3-1)、(3-2)重新计算粒子的速度和位置；

Step7：若条件没有满足且未达最大迭代$t_{max.}$,则转向2);

为了分析本章提出的算法的有效性，这里简单地分析算法的收敛性。DPSO算法给权重的影响有下列定理：

定理4.1：DPSO算法的权重w_i的范围仍为[w_{min}，w_{max}];

证明：在 (4-6) 式中，因为$0 \le s(i，g) \le 1$,

所以，$w_{min} \le w \le w_{max}$;

由此得到：$0 \le w-w_{min} \le w_{max}-w_{min}$，

再由 $0<\frac{t_{\max}-t}{t_{\max}}<1$，

所以，由公式(4-7)，得：

$w_{min} \le w_i \le w_{max}$，

证毕

根据定理4.1，DPSO算法计算的权重能保证在标准PSO算法的权重大小范围内，所以，DPSO算法能保证标准PSO算法的收敛性。而且根据公式（4-7），DPSO迭代到最后的权重w_i的值不大于w_{max}，因此，DPSO算法有可能存在更好的局部探测性。

4.4.4 实验与分析

为了验证本文算法的有效性，分别采用标准 PSO 算法和 DPSO 算法对下列函数的最小化问题进行实例计算并进行对比。

(1) Goldstein-Price 函数

$$f_1(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]$$
$$\times[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)] \qquad |x_i| \le 2$$

其最优状态和最优值为： $\min\left(f\left(x^*\right)\right)=f(0,-1)=3$ 。

(2) J.D.Schaffer 函数

$$f_2(x)=\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{(1+0.001*(x_1^2+x_2^2))^2}+0.5 \qquad \text{-5.12} \leqslant x_1, x_2 \leqslant 5.12$$

函数 f_2 的全局极小点是(0 ,0)，其最优值是 0。

第一类实验数据如表4-4，实验方法是在相同的条件下，比较两个算法收敛最优的百分比(在计算100次中有多少次在误差范围内收敛)以及收敛时平均收敛代数。这里还需要考虑几个参数如何设置。对于函数f_1，参数设置为：c_1=1.8；c_2=1.8；$w_{\max}$=0.8；$w_{\min}$=0.2；$v_{\max}$=1；$d_{\min}$=0.5；$d_{\max}$=1；对函数f_2这里参数的设置c_1=1.8；c_2=1.8；$w_{\max}$=1.0；$w_{\min}$=0.4；$v_{\max}$=10；$d_{\min}$=10；$d_{\max}$=50。对上述问题进行了100次仿真计算，其计算结

果如表4-4所示。

表4-4 测试函数的平均寻优结果比较（在计算100次情况下）

函数	群体规模		进化	收敛	平均收敛率		平均收敛代数	
	PSO	DPSO	代数	误差	PSO	DPSO	PSO	DPSO
f_1	10	10	100	1e-20	89	93	91.191	50.677
f_1	20	20	100	1e-20	100	100	87.400	42.440
f_2	20	20	100	0.001	20	29	57.117	53.421
f_2	50	50	200	0.001	71	81	68.619	55.525

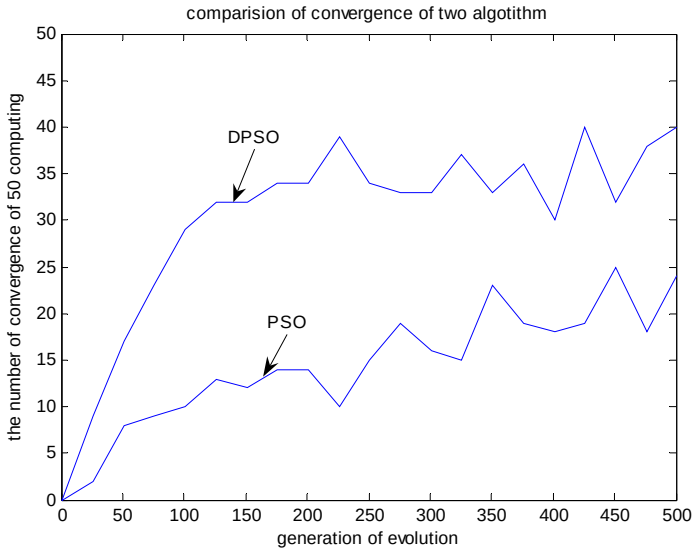

图 4-3 f_2 进化代数收敛的变化

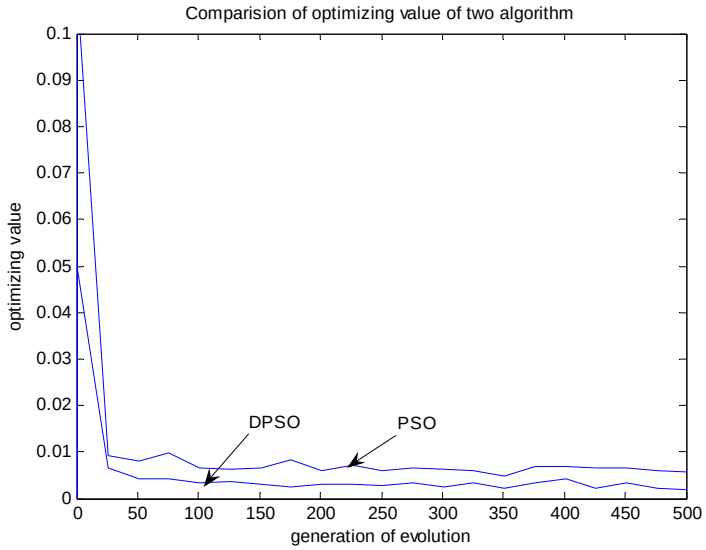

图 4-4 f_2 进化代数最优值的变化

从表 4-4 中可以看出，不论函数 f_1 还是 f_2，不论从平均收敛率还是从平均收敛代数来看，DPSO 算法比标准 PSO 算法有所提高。但性能改善程度来看，平均收敛率比平均收敛代数改善程度更好。因为 DPSO 其主要改善收敛问题，在尽量对探索能力不影响情况下，希望能够改善其收敛性能，更好性能是在粒子第一次找到最优点时，不会飞出此邻域，而产生抖动，振荡现象。从平均收敛代数性能提高中，我们可以发现 DPSO 比 PSO 性能有提高。说明 DPSO 算法能更早收敛最优点，不会轻易飞出最优点。为了进一步比较两个算法的性能，我们在相同的参数设置下，对多峰函数 f_2 进行第二类实验计算。每隔增加 25 代进化数时，记录 50 次计算平均最优解和 50 计算的平均收敛次数，即在误差为 0.001 时，50 次计算中其能够收敛的次数。测试的最大进化代数为 500，其结果如图 4-3 和图 4-4 所示。其中，PSO 表示标准的粒子群算法，DPSO 表示动态粒子群算法。从图 4-3 中可以看出 DPSO 的收敛次数无论在多少个进化代数情况下，其收敛比例比 PSO 算法要高。对于这个一个比较复杂的多峰函数来说，DPSO 算法平收敛率都比标准 PSO 算法要高。从图 4-4 可以看出，其 DPSO 算法的最优值在不同代数情况下，其所求出最优点要比 PSO 算法计算效果要明显提高，速度更快。表 4-4 数据也显示，DPSO 的平均收敛代数要 PSO 代数要小，但其在单峰函数优化问题来看，其效果与 PSO 相比并没有什么明显提高，这一点可以从表 4-4 结果中可看出这样的效果。这点可以从我们前面分析改进的思想源泉中得到解释，因为我们新算法主要目点为了改进算法早期全部进行探索，DPSO 里面无论早期还是晚期，各个粒子其实分划了三部，离最优粒子比较近的一部分，其权重很小，其主要负责在全局最优点附近进行探索，还要一部离粒子很远一部分，其权重始终是最大值，其主要负责在全局范围内进行探索，而中间部粒子其权重是随着代数变化和距离进行变化，线性变小。这样其在早期是探索，晚期是主要是收敛于全局最优点。所以在多峰函数时，新算法不容易错过早期就找到的最优点，也不会使函数很容易在局部最优点收敛。因此这就主要针对多峰函数时有很好效果。对单峰函数每一代进化只要是靠近当前最优点飞行就一定使能找到最优点的方向，而对多峰函数靠这思想飞行则不利。

本节通过分析标准粒子群算法，提出了一种能使其惯性权重随不同粒子与最优粒子距离动态变化的算法。仿真结果表明提出的动态粒子群算法在求多峰函数时，不仅在算法收敛效果与收敛速度都有所提高，本节意义是根据粒子不同来改变权重参数，其根据粒子与群体最优粒子的相似度来调整，所以可为一种改进粒子群算法的新思路。

4.5 本章小结

本章提出一个相似度的概念。根据第三章的理论分析，粒子最终会收敛到群体最优粒子。因此，每个粒子与群体最优粒子的相似度是一个很重要的信息量。根据这个信息量，本章提出一个计算粒子群体多样性的概念——聚集度，据此构造一个增加粒子群多样性的方法——变异粒子的位置。根据这个信息量，本章还提出一个让粒子权重动态变化的方法。本章方法具有理论依据，也具有实际应用。但本章算法新增加两个参数 $d_{\max}$ 与 $d_{\min}$，这个两参数取值随着问题不同取值不一样。所以，对这两参数如何设置需进一步值得讨论。

第五章 基于 PSO 思想的进化算法

5.1 引言

粒子群优化算法是一种模拟群体智能的优化算法，其形式主要迭代递推方式进行。算法在迭代中考虑了各粒子之间的相互联系（全局点）和各维之间的关系（函数适用值），所以粒子的迭代运动轨迹其实是一个复杂系统变化。正如第三章所叙，目前对 PS0 进行算法理论分析的方法是在简化系统的基础上，采用动态差分方程，利用线性系统理论进行的。这种分析方法虽然不是完善，但它描述 PS0 算法思想原理，说明了 PS0 算法是一种迭代进化系统。

本章基于 PS0 算法的思想原理，提出一种新的进化算法，并分析算法参数的选取，分析新的进化算法的收敛性。通过优化基准函数，对新算法与标准 PS0 进行实验比较。

5.2 PSO 算法的分析理论

已有不少文献对 PSO 算法作了数学模型的分析，本文在第一章和第三章已作了介绍。*Fan van den Bergh* 在其博士论文也是根据线性微分方程理论对粒子收敛性作分析 [18, 19] ，本文第三章对其的分析理论作了比较详细推导。为了本章算法构造的需要，对其结论作简要叙述如下：

假设标准PS0算法的$p(t)$和$p_g(t)$不变，即 $p = p(t)$ 和 $p_g = p_g(t)$；令$\phi_1=c_1\mathrm{rand}()$，$\phi_2=c_2\mathrm{rand}()$且$\phi_1$和$\phi_2$为常数。

根据（3-1）式和（3-2）式，利用线性差分方程推得下式 [18, 19] ：

$$x(t) = k_1 + k_2\alpha^t + k_3\beta^t \tag{5-1}$$

其中

$$k_1 = \frac{\phi_1 p + \phi_2 p_g}{\phi_1 + \phi_2} \tag{5-2}$$

$$\beta = \frac{1 + w - \phi_1 - \phi_2 - \gamma}{2} \tag{5-3}$$

$$\alpha = \frac{1 + w - \phi_1 - \phi_2 + \gamma}{2} \tag{5-4}$$

$$\gamma = \sqrt{(1 + w - \phi_1 - \phi_2)^2 - 4w} \tag{5-5}$$

$$k_2 = \frac{\beta(x(0) - x(1)) - x(1) + x(2)}{\gamma(\alpha - 1)} \tag{5-6}$$

$$k_3 = \frac{\alpha(x(1)-x(0))+x(1)-x(2)}{\gamma(\beta-1)} \tag{5-7}$$

$$x(2) = (1+w-\phi_1-\phi_2)x(1) - wx(0) + \phi_1 y_p + \phi_2 y_g \tag{5-8}$$

若 ϕ_1、ϕ_2、p 和 p_g 是固定不变的常数且(1+w- ϕ_1-ϕ_2)<4w，则粒子位置 $x(t)$的轨迹按下式收敛：

$$\lim_{t\to\infty} x(t) = (1-\lambda)p + \lambda p_g \qquad \lambda\in(0,1) \tag{5-9}$$

其中 $\lambda = \dfrac{\phi_1}{\phi_1+\phi_2}$

5.3 新算法的模型

粒子群的算法是由(3-1)、（3-2）、（3-3）和（3-4）四个式子组成，而（5-1）式根据(3-1)式和（3-2）式推导出来的。虽然是在简化的条件下推得的，但（5-1）式可以认为是标准 PSO 算法中（3-1）式和(3-2)式的等价形式。因此，如果用（5-1）式代替标准 PSO 算法（3-1）和（3-2）式，则得到一种新的算法形式。但是，标准 PSO 算法随机因素必须考虑进去。所以，下面设计一种新进化算法，新进化算法用（5-1）式为进化公式，代标准 PSO 算法的速度更新公式(3-1)和位置更新公式(3-2)，即有

$$x(t) = k_1 + k_2\alpha^t + k_3\beta^t$$

这里k_1，k_2，k_3及α，β都是参数，而t是动态变化迭代代数，即为算法的因变量，粒子位置更新变化是随进化代数 t而变化。

根据（5-2）式，可得：

$$k_1 = \frac{\phi_1 p + \phi_2 p_g}{\phi_1+\phi_2} = \frac{\phi_1}{\phi_1+\phi_2}p + \frac{\phi_2}{\phi_1+\phi_2}p_g \tag{5-10}$$

因此，当ϕ_1，ϕ_2 是随机因素时，则 k_1 为如下形式：

$$k_1 = rand()*p + (1-rand())p_g \tag{5-11}$$

（5-11）式体现的是 k_1 在粒子自身最优点 p 和群体最优点 p_g 之间线段上随机采样。把(5-11)式代入(5-1)式，则得粒子位置 $x(t)$轨迹变化式为如下：

$$x(t) = rand()*p(t) + (1-rand())p_g(t) + k_2\alpha^t + k_3\beta^t \tag{5-12}$$

同样，新进化算法还需具有粒子自身最优位置 $p(t)$和群体最优位置 p_g(t)的更新公式，即为如下两式：

$$If\ \ f(x(t)) < f(p(t)),\ \ then\ \ p(t) = x(t) \tag{5-13}$$

$$If\ \ f(p(t)) < f(p_g(t)),\ \ then\ \ p_g(t) = p(t) \tag{5-14}$$

在(5-12)式中，粒子位置迭代变化的因变量为迭代步数 t，而 k_2，k_2，α和β都是参数，其取值在下一节讨论与分析。这样，标准 PSO 算法变为新的进化算法形式，

其算法流程描述如下:

算法流程描述:

设粒子规模数为m，最大迭代数为t_{max}。

Step1：初始化：随机产生n个粒子位置；

Step2：评价每个粒子的适应值；

Step3：根据(5-13)式确定迄今为止每个粒子找到的最好位置p_i；

Step4：根据(5-14)式确定迄今为止整个群体找到的最好位置p_g；

Step5：根据(5-12)式调整粒子的位置$x(t)$；

Step7：若迭代条件没有满足且未达最大迭代步数t_{max}，则转向Step2；

与标准 PSO 算法不同，新算法流程的 Step5 发生了变化，其粒子位置更新公式为(5-12)式，不同标准 PSO 算法的(3-2)式，而且没有了速度更新公式(3-1)式。

5.4 参数设置与分析

根据(5-12)式，新的进化算法需要考虑 k_2, k_3 及α，β的参数设置。参数的设置尽量符合标准 PSO 算法性能。所以，新进化算法的参数设置需要考虑标准 PSO 的参数对它们的影响。

5.4.1 参数α，β的设置

为了确定这两个参数，分析(5-3)式和(5-4)式。参数α，β受到标准 PSO 算法的参数 w, ϕ_1和ϕ_2的影响，而ϕ_1和ϕ_2是受随机变量的影响，且$\phi_1=c_1 rand()$， $\phi_2=c_2 rand()$，这里ϕ_1和ϕ_2在$[0，c_i]$$(i=1，2)$区间进行随机均习采样。在标准 PSO 算法中，权重 w 是一个随迭代而递减的。根据（5-5）式，参数α和β存在复数值的情况。但根据(3-1)式和（3-2）式，标准 PSO 算法实际运行时，α，β的值不应该存在复数，因为迭代时间 t 是整数，所以参数应该是实数，不存在复数。为了分析参数α，β取值范围，根据（5-3）和(5-4)两个式子，观察参数α，β随权重 w 变化。由于这个两个式子很复杂，并且带有随机量 $rand()$，因此采用图形方式来观测其变化。利用 matlab，演示参数α，β随权重 w 的变化，其结果如图 5-1 和图 5-2。

观察图 5-1 和图 5-2 中的参数α，β随权重 w 的变化，当权重 w 从右向左变小时，参数α，β的值会随机性地变小。而且，当权重 w 小于 1 时，参数α，β的值也小于 1。这很有意思说明，当权重小于 1 时，(5-1)式收敛。这两个图形很形象说明标准 PSO 算法收敛性的要求。

从图 5-1 和图 5-2 可以看到，当权重在 0.4 到 1.4 变化时，参数α，β的值大概在 0.4 到 0.6 之间变化波动，波动差异是[0，0.8]。由标准 PSO 算法的权重变化一般从

1.4 到 0.4 的变化比较合适，根据图 5-1 和图 5-2，参数α，β的取值设计为：

$$0.8\times rand(\,)+w \tag{5-15}$$

这里，w 随迭代时间步数 t 线性递减，开始其值取最大值，最后为最小值。观察图 5-1 和图 5-2，新的进化算法的权重 w 从 0. 4 递减变化到 0 比较合适。如里 w 取 0.4 到 0 的值，则图 5-3 演示(5-15) 式。观察图 5-3， (5-15) 式的图形变化与图 5-1 的图形基本吻合。但要求参数 w 递减，并带有随机数 rand( )。递减目的是：在早期时，(5-15)式的值偏大；在晚期时，(5-15)式的值偏小。使早期算法具有发散性，提高全局搜索能力。晚期(5-15)式的偏小，算法具有收敛性，提高局部探测性。

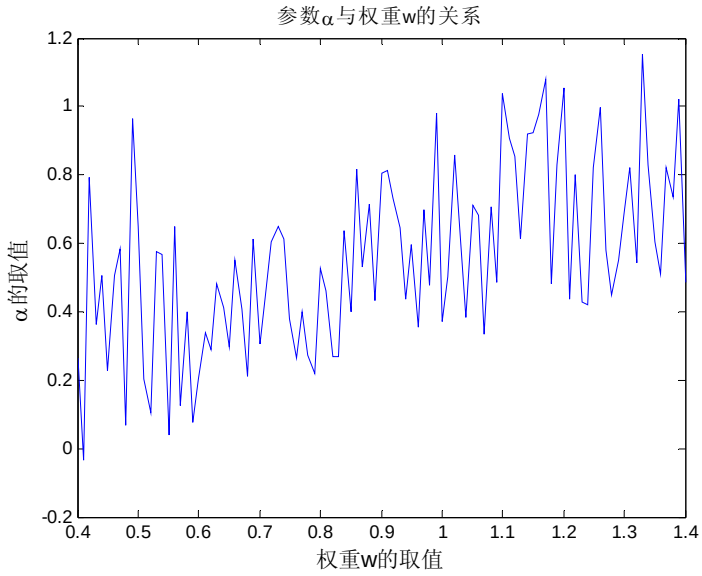

图 5-1 参数α随权重 w 变化

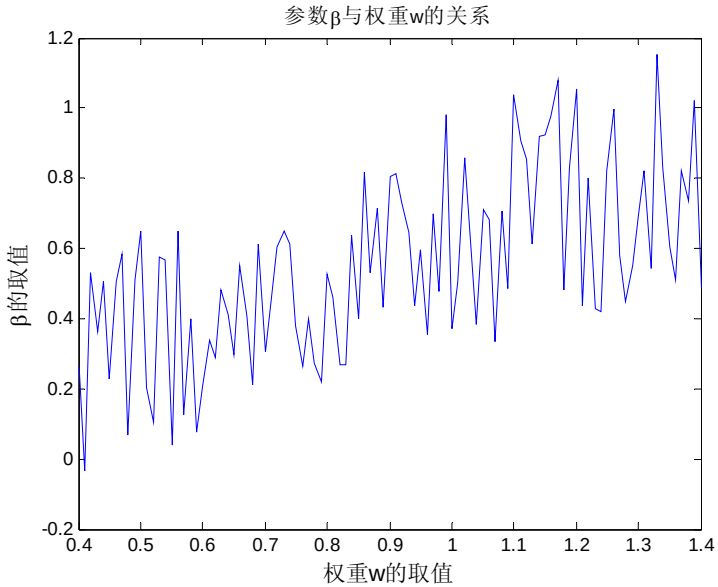

图 5-2 参数β随权重 w 变化

参数α，β的取值原则是：两个参数的绝对值早期要偏大，为大于 1 的值，而晚期为小于 1 的值，而且带有随机性。这样的取值方式容易控制算法的收敛性，也能更好地控制算法全局探测性与局部探索性的平衡

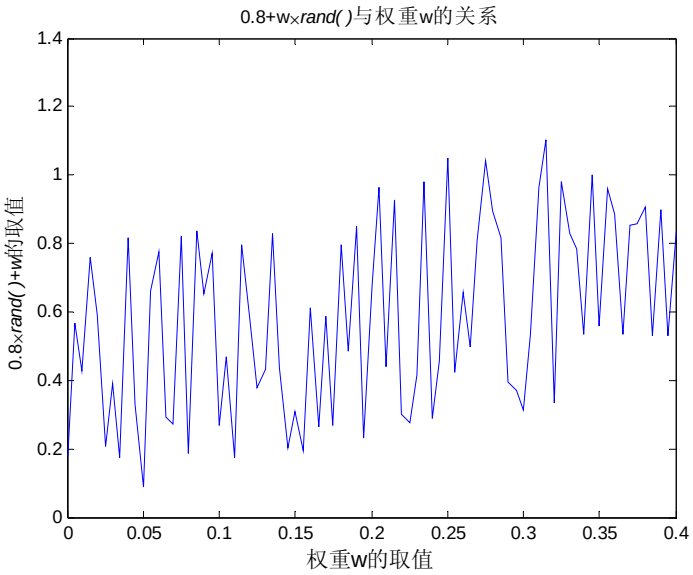

图5-3 式(5-15)取值0-0.4动态变化

5.4.2 参数 k_2，k_3 的设置

同样，为了确定参数 k_2，k_3，分析（5-6）式和（5-7）式。与参数α，β的设置类似，参数 k_2，k_3 受到标准 PSO 算法权重 w 的影响。而且还受粒子初始三代位置 $x(0)$、$x(1)$和 $x(2)$的影响。为了确定参数 k_2，k_3 的取值范围，根据（5-6）式和（5-7）式，采用图形演示的方法，分析参数 k_2，k_3 随标准 PSO 算法权重 w 的变化情况。但还需考虑初始粒子的三代位置，需要输入粒子的初始三代位置。第一代位置 $x(0)$是在搜索空间随机采样，而第二代位置 $x(1)$和第三代位置 $x(2)$在标准 PSO 算法中受初始位置 $x(0)$和初始速度 $v(0)$的影响。但这里为了简单考虑，三代初始位置都认为在搜索空间中随机采样。在进化代数较多时，这样的简单考虑还是比较合理。

当初始位置 $x(0)=5$、$x(1)=5$ 和 $x(2)=5$ 时，图 5-4 演示了（5-6）式和（5-7）式中 k_2、k_3 与权重 w 的关系；而当初始位置 $x(0)=-5$、$x(1)=-12$ 和 $x(2)=-60$ 时，图 5-5 演示（5-6）式和（5-7）式中 k_2、k_3 与权重 w 的关系。四个图形显示如下特征：(1).k_2、k_3 的值基本是在 0 附近上下波动，其值也可能为负，也有可能为正。(2).k_2、k_3 存在突然巨变的情况，绝对值达到几千。第二个特点是由于在推导(5-1)式时没有考虑标准 PSO 算法的速度限制值 v_{max}，因此会存在巨变的情况。如果粒子速度或粒子位置被限制，k_2、k_3 的值也将受到限制。在标准 PSO 算法中，不可能出现粒子位置超出搜索区域范围的情况。

结合以上的两点特征，k_2、k_3 的值是应该在 0 值的上下区域波动，因此把 k_2、k_3 的形式改为如下形式：

$$\pm c \times rand() \tag{5-16}$$

c 是一个参数，$rand()$在[0, 1]区间的随机数。

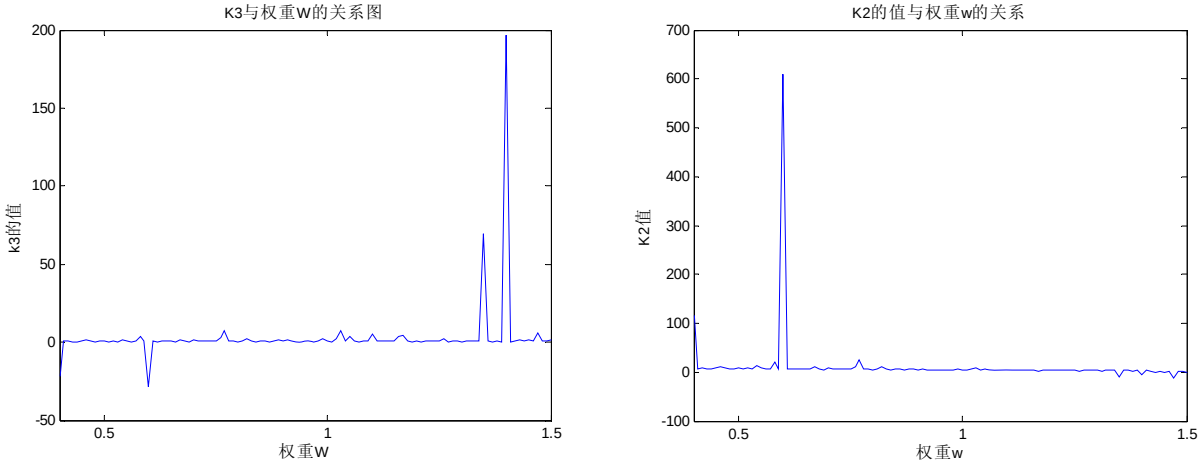

图 5-4　初始位置 $x(0) = 5$、$x(1) = 2$ 和 $x(2) = 2$

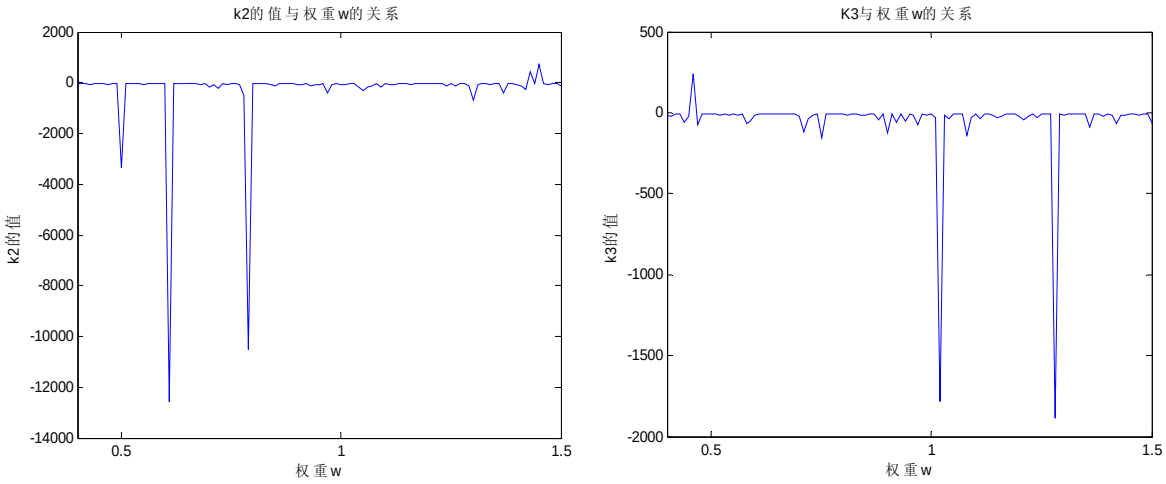

图 5-5　初始位置 $x(0) = -5$、$x(1) = -12$ 和 $x(2) = -60$

至此，再回到(5-12)式，即如下公式

$$x(t) = rand() * p(t) + (1 - rand()) p_g(t) + k_2 \alpha^t + k_3 \beta^t \tag{5-12}$$

根据前面分析的讨论，结合(5-15)式和(5-14)式，可以把(5-12)修改为下列两式：

$$\begin{aligned} x(t) &= rand() * p(t) + (1 - rand()) * p_g(t) \\ &+ (-c * rand()) * (0.8\, rand() + w)^t + (c * rand()) * \left(0.8\, rand() + w\right)^t \end{aligned} \tag{5-17}$$

显然，上式只要确定参数 c 和 w 即可。至于取什么具体值，后面作实验分析讨论。

5.4.3 初始化问题

对于随机性智能优化问题，其算法最终结果与初始值有关。本算法由于没有初始速度，只需对粒子位置进行初始化及每个个体最优与全局最优进行初始化。初始的方法是：初始位置在搜索空间随机采样，而个体最优的初始值取初始的位置，全局最优为个体最优初始值的最好值。

5.4.4 算法的分析

根据(5-17)式，算法收敛条件是(5-15)式的绝对值小于 1。很显然，这个式子的值由参数 w 来决定，只要选择合适参数，算法就会收敛于 $rand(\ )*p+(1-rand())*p_g$。而 $rand(\ )*p+(1-rand())*p_g$ 的值由下式更新：

$if \quad f(x(t))<f(p)\ then \quad p=x(t)$,

当 $x(t)$跑到函数 $f(x)$凸性区域时，会出现 $p\rightarrow p_g$。因此，算法最终会有 $x(t)\rightarrow p_g$。这个分析方法与第三章分析说明是一致的。

进化算法需要早期具有全局探测性，晚期具有局部探索性。新算法能通过设置参数 w 的值来达到这个要求。参数越大，其全局探测性就强；而参数越小，其局部探索性更强。所以，本算法也采用参数 w 随进化代数递减的方法。w 早期保持(5-17)式处于发散状态，而晚期让其处于收敛状态。这里选择 w 取值为 0.8→0.2。w 的最小值 0.2 也可使 $0.8\times rand(\ )+w$ 式小于 1，可以保持算法收敛。在算法实验运行中，进化代数一般会大于 500，而 $(0.9)^{500}$= 1.3221e-023 是足够小的数。事实上，最小值可以根据算法的最大进化代数动态调整，最大进化代数越大，其值应变大，但需要保持小于 1。这样，可以避免算法早熟收敛。

5.5 实验与分析

为了验证本章算法的有效性，对标准PSO算法和新算法进行比较。选定 5 个基准测试函数来进行实验，其形式见表5-1。

参数设置如下：经过一定实验测试，新算法权重w设置0.8到0.2，标准PSO算法的权重设置为1.2到0.4，两个算法的权重都随进化迭代而线性递减。参数c_1, c_2还是为1.8。种群数和进化代数根据各个测试的函数不同而设置不同的值，见各表。

为了与标准PSO算法比较，每个算法运100次，计算出每次算法的收敛次数，平均最优值及平均收敛代数（见表5-2到表5-6）。

表5-2和表5-3是两个多峰函数的实验效果。表中数据说明新进化算法的优化效果不逊于标准PSO算法，因此完全可以看作标准PSO算法的等价算法。新算法与标准PSO算法相比，在种群粒子数越多的情况下，新进化算法效果更好。根据两表的数据，

每次实验都满足于种群数×进化代数=20000。这样的实验条件可以保持每次实验的时间性不变。这个效果说明新算法可能更容易收敛，其收敛性偏快，更易陷入局部最优。

表 5-1 五个测试函数

函数名	表达式	维数	初值范围	目标值
Schaffer	$f_1(x)=\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{(1+0.001*(x_1^2+x_2^2))^2}+0.5$	2	$[-5.12, 5.12]^2$	10^{-3}
Rastrigin	$f_2(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	n	$[-5.12,5.12]^n$	10^2
Griewank	$f_3(x)=\frac{1}{4000}\sum_{1}^{n}x_i^2-\prod_{1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	n	$[-100,100]^n$	10^{-1}
Sphere	$f(x)=\sum_{i=1}^{n}x_i^2$	n	$[-100,100]^n$	10^{-2}
Rosenbrock	$f(x)=\sum_{i=1}^{n-1}(100(x_{i+1}-x_i^2)^2+(x_i-1)^2)$	n	$[-30,30]^n$	10^2

若最大进化代数为250；因为$(0.9)^{250}$=3.6360e-012,这样权重的指数非常小。粒子群的数为80；更多粒子数有利于早期提高全局探索性，因本算法的局部收敛性更强，局部搜索能力强，需要增加粒子数增加全局探索能力，同时进化代数减少（为250代），这样时间性复杂性不会会增加。

表 5-2 两种算法对 Schaffer's 函数 2 维的优化结果

Schaffer's f_6				
种群数	进化代数		PSO	NPSO
20	1000	收敛次数	31	31
		平均最优值	0.0067	0.0064
		平均收敛代数	873.5806	580.3548
40	500	收敛次数	54	53
		平均最优值	0.0045	0.0043
		平均收敛代数	435.6481	286.0189
80	250	收敛次数	61	69
		平均最优值	0.0040	0.0033
		平均收敛代数	182.9836	141.1594

表 5-3 两种算法对 Rastrigrin 函数 30 维的优化结果

种群数	进化代数		PSO	NPSO
20	1000	收敛次数	72	86
		平均最优值	102.5627	98.5646
		平均收敛代数	678.2083	961.7791
40	500	收敛次数	84	83
		平均最优值	99.9066	99.3631
		平均收敛代数	379.2024	484.1928
80	250	收敛次数	83	69
		平均最优值	99.6170	102.9864
		平均收敛代数	223.1205	244.7241

表5-4和表5-6演示了三个函数用标准PSO和新算法分别在不同维数计算的结果。从三个表计算出的数据可以看到，新算法的效果基本上与标准算法相当。再次验证本章新算法基本可以替代标准PSO算法。但对算法参数具体取值，需要作进一步的精确分析。

表 5-4 两种算法对 Sphere 优化的结果

Sphere					
种群数	进化代数	维数		PSO	NPSO
20	1000	10	收敛次数	100	99
			平均最优值	0.0083	0.0064
			平均收敛代数	490.5000	580.3548
40	1000	20	收敛次数	100	93
			平均最优值	0.0089	0.0093
			平均收敛代数	574.8700	686.0189
80	1000	30	收敛次数	100	95
			平均最优值	0.0091	0.0099
			平均收敛代数	632.0900	691.7826

表 5-5 两种算法对 Rosenbrock 优化的结果

Rosenbrock					
种群数	进化代数	维数		PSO	NPSO
20	1000	10	收敛次数	81	80
			平均最优值	111.8752	207.8816
			平均收敛代数	497.0123	869.2500
40	1000	20	收敛次数	77	73
			平均最优值	157.7079	162.55
			平均收敛代数	589.6494	443.4583
80	1000	30	收敛次数	74	79
			平均最优值	134.5893	112.5684
			平均收敛代数	646.3784	651.7826

表 5-6 两种算法对 Griewank 优化的结果

Griewank					
种群数	进化代数	维数		PSO	NPSO
20	1000	10	收敛次数	100	100
			平均最优值	0.0067	0.0064
			平均收敛代数	950.5000	980.3548
40	1000	20	收敛次数	99	94
			平均最优值	0.0915	0.0925
			平均收敛代数	529.4343	961.4681
80	1000	30	收敛次数	100	99
			平均最优值	0.0041	0.0033
			平均收敛代数	941.6000	951.7826

5.6 新算法在单交叉口信号控制中的运用

在城市交通路口信号控制中，信号周期是一系列基本交通信号组合的循环时间。而信号相位是信号周期中具有通行权的一组车流通过路口的持续通行时间，根据路口几何状况和车流量信息可设置多个信号相位。通过给具有通行权的车流设置绿灯相位，而与其冲突的其他车流设置为红灯相位来进行路口信号控制，实现路口车流在时间上的优化分配。

交通是城市经济活动的命脉，对城市交通路口交通灯实施合理优化控制，有利于缓解日趋紧张的交通拥挤问题。对于不同的交通路口，由于道路上的交通流呈现很大的随机性，各个方向乃至各个车道车流不一样，车流行驶过程是一种随机变化的过程，所以对交通路口信号控制主要体现在对交叉路口各个相位的配时实施不同控制。一种配时方案的改变，会影响各个车道的车流。系统对单交叉口实时控制的实质在于优化控制信号配时。目前信号配时主要采用基于 F.Webster 近似公式的定时控制，这种控制方式不能针对当前变化的交通流情况实施有效的控制。为此，本节采用本章的优化算法来进行单交叉口多相位的实时控制，以实时地对交通信号进行控制。

5.6.1 单交叉口信号控制模型

1. 模型概念与参数 [115]

图 5-6 所示的城市单交叉路口六车道四相位信号控制问题。在图 5-7 中可以看出，车流从四个方向流入交叉口，并通过直行、左行和右行三个方向流出。根据交通规则，这里右行不作考虑，因此不同的相位、不同的车道的车辆放行状态如图 5-7 所示。在任一时刻，只有一个相位(即绿灯相位)的车流有权通过交叉路口，而其他相位(即红灯相位)的车辆则在交叉口停车线以外等候，直到获得通行权，即该相位由红灯转为绿灯，信号灯分红、绿、黄三种。采取的控制策略为：黄灯时间为 3s，红绿灯最小绿灯时间为 15s，最大绿灯时间为 45s。控制的方法是依据各车道的车流量信息，以路口流通能力最大，排队候车时间最短或等候的车辆数最少作为优目标函数，对交叉口交通信号进行综合优化，实时修正每个周期各相位的配时，以使交通控制获得最佳的性能指标。

图 5-7 所示的一个四相位变化控制的单交叉口，不同的相位、不同的车道的车道的车辆放行状态可用一个系数矩阵 p_e 表示：

$p_e=\{p_{ijk}\}$

其中：i 为相位序号，取值为 1、2、3、4，分别表示第一、第二、第三、第四相位；j 为方向序号，取值为 1、2、3、4，分别表示东、南、西、北方向；k 为车道序号，取值为 1、2，分别表示左行、直行车道；

$$p_{ijk}=\begin{cases}1, & \text{表示第i相位、第j方向、第k车道车辆放行}\\ 0, & \text{表示第}i\text{相位、第}j\text{方向、第}k\text{车道车辆禁止放行}\end{cases} \tag{5-18}$$

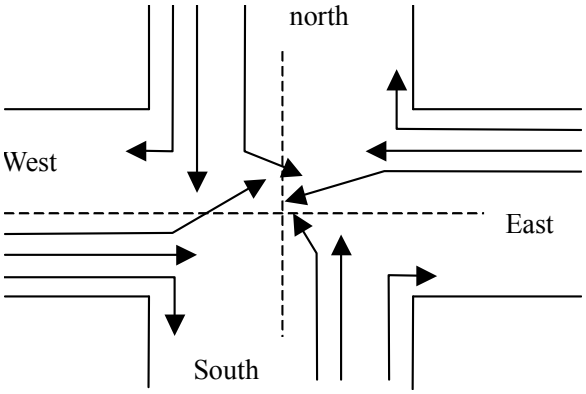

图 5-6 单交叉路口交通流分布

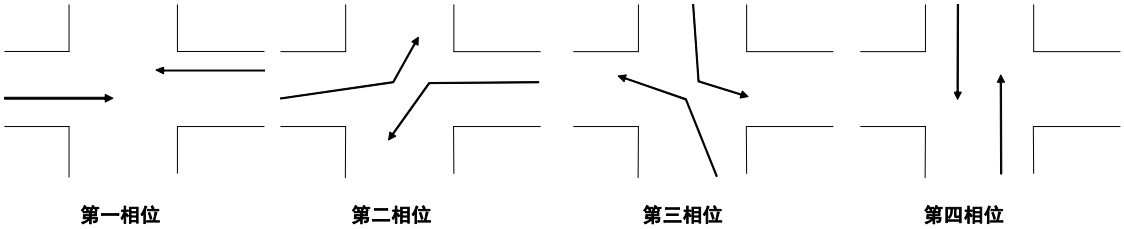

图 5-7 不同车道、不同相位放行情况

则对采用如图5-7所示的四相位信号控制的交叉路口，其放行状态系数矩阵可表示为：

$$\mathrm{P}_e=\begin{Bmatrix}(0,1,0,0,0,1,0,0) & (1,0,0,0,1,0,0,0)\\ (0,0,0,1,0,0,0,1) & (0,0,1,0,0,0,1,0)\end{Bmatrix} \tag{5-19}$$

2. 延误车辆数的计算 [116]

设：t_i(i=1、2、3、4)为交叉路口各个相位的配时；λ_{ijk} 表示第 i 个相位、第 j 个方向、第 k 个车道的车辆到达率，则一个周期内第 i 个相位、第 j 个方向、第 k 个车道到达的车辆数为

$s_1=\lambda_{ijk}t_i$

假设在绿灯期间内，放行车辆在第 i 个相位、第 j 个方向、第 k 个车道驶离路口的离开率为 u_{ijk}，则一个周期内第 i 个相位、第 j 个方向、第 k 个车道可能驶离路口的车辆数为

$s_2=p_{ijk}u_{ijk}t_i$

设 s^l_{ijk} 表示第 l 个周期、第 i 个相位、第 j 个方向、第 k 个车道滞留的车辆数，则：

$$s^l_{ijk}=s^l_{i-1jk}+\lambda_{ijk}t_i-p_{ijk}u_{ijk}t_i \quad s^l_{i-1jk}+\lambda_{ijk}t_i\geq p_{ijk}u_{ijk}t_i \tag{5-20}$$

$$s^1_{ijk}=0 \quad s^l_{i-1jk}+\lambda_{ijk}t_i<p_{ijk}u_{ijk}t_i \tag{5-21}$$

其中 i=1, 2, 3, 4； j=1, 2, 3, 4； k=1, 2, 3。

当 i=1 时，s^1_{i-1jk} 为第 i-1 个周期、第 j 个方向、第 k 个车道、第四相位滞留的车辆数。故第 1 个周期末路口总的滞留车辆数可表示为：

$$s=\sum_{j=1}^{4}\sum_{k=1}^{3}s^l_{4jk} \tag{5-22}$$

3. 系统的性能指标描述

从以上分析可知，为了使路口流通能力最大，即要求路口滞留车辆数最小。满足

$$\begin{aligned} s^* &= \min s=\min\sum_{j=1}^{4}\sum_{k=1}^{3}s^l_{4jk} \\ &= \min\sum_{j=1}^{4}\sum_{k=1}^{3}\Big[s^{l-1}_{4jk}+\sum_{i=1}^{4}\lambda_{ijk}t_i-\sum_{i=1}^{4}p_{ijk}u_{ijk}t_i\Big] \end{aligned} \tag{5-23}$$

优化过程是对路口 4 个相位进行实时优化配时 t_1、t_2、t_3、t_4，且满足

$$t_1+t_2+t_3+t_4=T \tag{5-24}$$

其中T为路口相位信号控制的周期。若控制周期不变，性能指标为在满足(5-24)式条件下求(5-24)式的极小值。考虑路口行人过马路时的安全需要，每相位最短绿灯时间不得小于某值e(一般取e>=6s)，因此每一相位的配时须满足条件(5-20)式

5.6.2.基于粒子群优化的单交叉口信号控制

由(5-23)和(5-24)组成带约束的非线性优化问题转化成无约束非线性优化问题，而粒子群算法适合于求无约束的非线性优化问题。所以，把由(5-23)和(5-24)组成带约束的非线性优化问题转化成无约束的非线性优化问题，采用加惩罚函数的方法，则原问题化为：

$$\begin{aligned} s^* &= \min s=\min\sum_{j=1}^{4}\sum_{k=1}^{3}s^l_{4jk} \\ &= \min(\sum_{j=1}^{4}\sum_{k=1}^{3}\Big[s^{l-1}_{4jk}+\sum_{i=1}^{4}\lambda_{ijk}t_i-\sum_{i=1}^{4}p_{ijk}u_{ijk}t_i\Big]+\alpha(\sum_{i=1}^{4}t_i-T)^2) \end{aligned} \tag{5-25}$$

其中 $(\sum_{i=1}^{4}t_i-T)^2$ 是添加的惩罚函数，参数α是一个充分大的一个正数，并且

同时满足：

$t_{\min} \le t_i \le t_{\max}$，$i$=1, 2, 3, 4。

这样，粒子的位置表示为（t_1，t_2，t_3，t_4），粒子适应度表示为：

$$f(t_1, t_2, t_3, t_4) = \sum_{j=1}^{4}\sum_{k=1}^{3}\left[s^{l-1}{}_{4jk} + \sum_{i=1}^{4}\lambda_{ijk}t_i - \sum_{i=1}^{4}p_{ijk}u_{ijk}t_i \right] + \alpha(\sum_{i=1}^{4}t_i - T)^2 \tag{5-26}$$

算法过程描述:

Step1：设置粒子优化算法的参数，初始化各粒子的位置和速度，对粒子群进行分组；

Step2：根据(5-26)式计算各粒子适用度；

Step3：判断是否需要更新粒子的个体极值，所在组的局部极值，以及粒子群的全局极值；

Step4：根据(5-12)式更新粒子的位置；

Step5：重复 Step2 到 Step4 的工作直到达到预先设定的迭代次数；

Step6：输出全局值值及 t_1, t_2, t_3, t_4。得到最优时间分配。

5.6.3 仿真实验

用 Matlab 编程实现上述算法的仿真。由于是四维空间，取种群为 100，迭代次数为 500 代。东南西北的左行和直行的车流到达率分别假定为：

da_e0= 15+ *12cos(kt)*； *da_el*= 15+ *5sin(2kt);*

da_w0= 12+ *16sin(4kt)*； *da_w1*= 12+ *18cos(kt);*

da_s0= 10+ *12sin(kt)*； *da_sl*= 11+ *16sin(2kt)*；

da_n0= 13+ *13sin(kt);* *da_nl*= 14+ *l0cos(kt);*

其中*k*= 0.314159，*t*是时间(单位为分钟)。而车流离开率为80辆 / 分钟。

仿真程序采用本章提出新的算法来求解目标函数(5-26)式的最优值。由于各方向、各车道的到达率是基于以上假定，所以实验时认为第一周期开始时，滞留车辆为零。由此，利用本章的算法计算得到这一周期的最佳配时，并得到各方向、各车道的滞留车辆数。实验时用第一周期的最佳配时作为以后各周期定时控制的配时方案，与本章算法实时配时作比较，看各周期两种算法滞留车辆数。不考虑黄灯的情况，信号周期均取 120s，绿灯最短时间 15s，最长时间为 45s。运行 10 个周期后，在不同交通流情况下交叉口的车辆延误数如表 5-7 所示。

表 5-7 两种配时控制的比较

周期	PSO 算法的动态配时					固定配时
	t_1	t_2	t_3	t_4	s	
1	43	41	17	19	51	51
2	44	35	23	18	54	55
3	43	35	24	18	55	59
4	43	42	18	17	54	63
5	43	38	22	17	56	67
6	44	32	20	24	57	71
7	44	32	27	17	55	75
8	44	37	22	17	55	79
9	42	39	15	24	56	83
10	42	34	24	20	54	87

从表 5-7 中的数据可以看到利用新的算法进行动态配时各周期滞留的车辆没有什么变化，随着配时周期循环，路口滞留车辆保持在一个稳定数据，不会造成路口车辆堵塞。而采用定时配时，随着周期次数增加，路口的滞留车辆数在逐渐增加，因此这样在一定时期后，会造成路口的车辆数量达一定负荷，使路口拥塞。所以从仿真实验结果来说，本算法是有效的。

本节采用新算法对单交叉口的信号进行配，由于基于 PS0 思想的新算法有利于函数型优化问题。本节虽然是针对单路口四相位交通流控制，同样可以扩展到更多相位的交通路口交通流控制。

5.7 本章小结

本章根据 PSO 算法的分析理论，提出一种新的进化算法形式，并分析新算法的参数。如果选择适当的参数，新算法效果与标准 PSO 相当。新算法具有一定独特性，其根据标准 PSO 算法的数学模型构造，新算法的参数不仅比标准 PSO 算法少，而且其参数取值的数学意义更明了。新的算法的数学含义是，粒子在自身最优点和全局最优点之间线段采样，然而加上一个迭代收敛的尾式，再附上自身最优点和全局最优点的更新公式。这种数学形式的收敛原理很容易理解。标准 PSO 模拟自然群体现象而产生的灵感得到的随机智能优化算法，本章的新算法根据它的数学模型，构造新的随机优化算法，这有助于从数学模模型中构造优化算法提供视角。

第六章 离散二进制粒子群算法分析

6.1 引言

标准 PSO 算法适应在连续搜索空间进行计算，而对于离散的搜索空间，其不能直接加以应用，必须对标准 PSO 算法改进。为了使 PS0 算法能解决离散组合优化问题，J. Kennedy 和 R. Eberhart 在 1997 年开发出一个 PSO 算法的离散二进制版本(简称 BPSO) [53] 。离散二进制 PSO 算法是用来优化离散二进制空间的问题，这扩展 PSO 算法的应用。该算法提出来已有十来年，不少文献将其应用于组合优化问题，但很少有文献对该算法进行分析。本章对离散二进制粒子群算法（BPSO）进行分析研究，在此基础上对 BPSO 算法作一些改进。

6.2 离散二进制粒子群算法

许多优化问题的搜索空间是离散的。一些典型例子就是排序问题，例如调度问题、路由问题。除了这种纯粹的组合优化问题外，研究者常常用二进制方法来处理连续问题，从而在离散数字空间来解决它。正是任何问题，无论是离散还是连续，都能用二进制数字表示，所以能操作二值函数的优化算法应该是比较有益的算法。

BPSO 本质上不同于原始的 PSO，首先粒子是由二进制编码组成，每个二进制位利用(3-1)式产生速度，而其速度值被转换成变换的概率，也就是位变量取 1 值的机会。此外，BPSO 参数之间关系也不同于正常连续 PSO 算法。J.Kennedy 和 R.Eberhart 通过函数优化仿真证明了离散二进制版本的鲁棒性，目前算法得到应用。

J.Kennedy 和 R.Eberhart 开发出来的离散二进制 PSO 算法的速度更新公式与原始 PSO 算法一样，但没有原始 PSO 的粒子位置更新公式。为了表示速度的值是二进制位取 1 的概率，速度的值被映射到区间[0，1]，映射的方法一般采用 sigmoid 函数：

$$s(v_{id}) = \frac{1}{1+\exp(-v_{id})} \tag{6-1}$$

这里 $s(v_{id})$表示位置 x_{id} 取 1 的概率，粒子通过(6-2)式改变它的位值：

$$x_{id} = \begin{cases} 1 & if\ rand() \le s(v_{id}) \\ 0 & otherwise \end{cases} \tag{6-2}$$

这里 *rand()*是一个随机数，从区间[0，1]的统一分布中随机产生。为了避免 $s(v_{id})$ 太靠近 1 或 0，一个参数 V_{max} 作为最大速度值，用于限制 v_{id} 的范围，即 $v_{id} \in [-V_{max}, V_{max}]$。速度的限制最终是限制了位 x_{id} 取 1 或 0 的概率。例如，如果 $V_{max}=6$，那么就限制 $S(v_{id})$的值在 0.9975 和 0.0025 之间。在连续 PSO 算法中，大的 V_{max} 值有利于粒子的全局开拓能力，而在离散 PSO 算法二进制版本中，小的 V_{max} 值才会增加位的变

异率。

(6-1)式的函数图形如图 6-1, 其函数值映射成区间[0,1]的值。根据(6-2)式，函数值代表了某位为 1 的概率。需要注意的是：Sigmond 函数值并不代表位变化概率，只是代表某位取 1 的概率。所以，在分析中需要考虑某位变化的概率是多少，本章将对此进行分析。

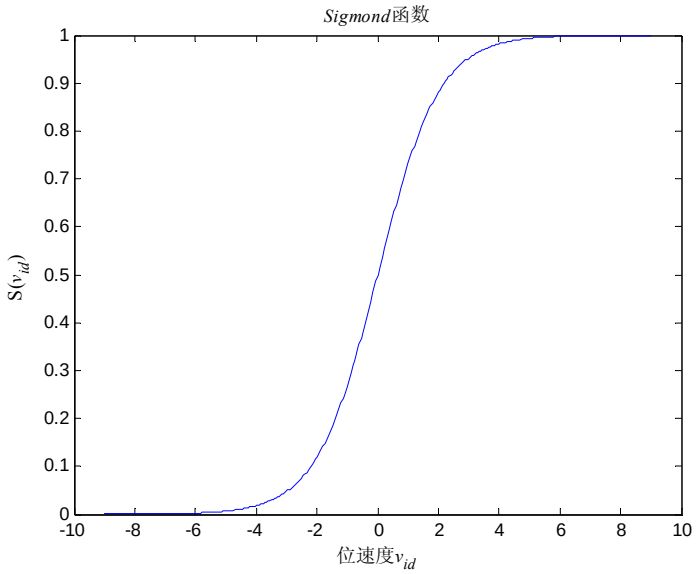

图 6-1　Sigmond 函数

为了便于分析，这里描述二进制 PSO 算法的过程：

算法过程描述:

Step 1 ：初始粒子群：将问题每一个解采用二进制编码表示；随机初始 n 个二进制数字，表示 n 个粒子；

Step 2 ：计算每个粒子的适应值：每个粒子位串解码得到参数，由参数得到目标函数值；

Step 3 ：更新个体最优值及全群最优：与现有各粒子的目标函数作比较更新个体最优和全局最优；

Step4：计算速度：对每个粒子的每位计算其速度，按(3-1)式计算；

Step5：产生新的粒子群：按(6-1)式计算每位的更新概率，按(6-2)产生新的粒子群；

Step6：若迭代条件满足，再输出全局最优粒子的目标值。否则转入 Step2

以上过程的流程图如图 6-2。

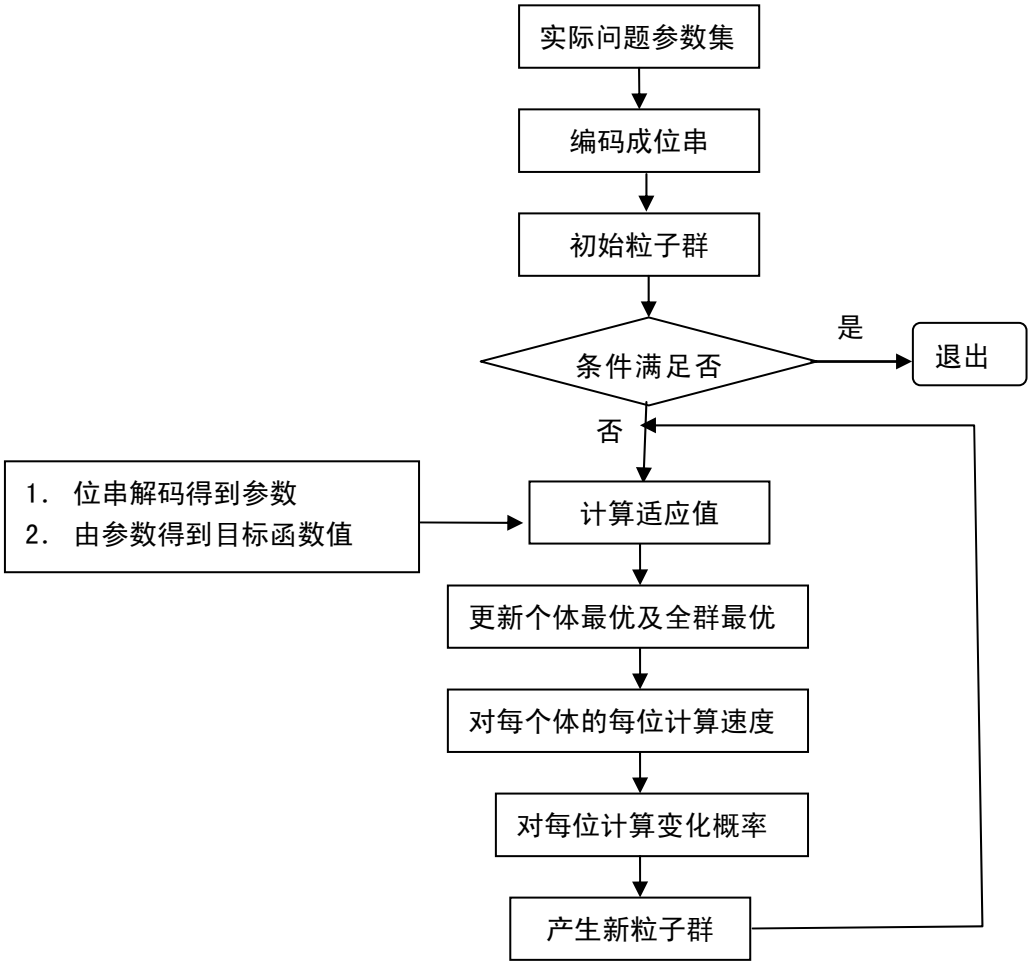

图 6-2　离散 PS0 算法流程图

6.3 二进制粒子群算法的分析

6.3.1 位改变率的分析

J. Kennedy 和 R. Eberhart 在文献[53] 提出位改变率的概念，这节介绍分析它。在二进制粒子群算法(BPSO)中，粒子轨迹是一种概率变化，其单维的速度转化为位取 1 的概率。即使 v_{id} 保持不变，根据 v_{id} 的值得到位取 1 的概率，粒子的位值也会产生变化。位为 1 的概率为 $S(v_{id})$，而位为 0 概率是 1-$S(v_{id})$。如果位已经是 0，则位发生改变的概率为 $S(v_{id})$；如果位已经是 1，则其发生改变的概率为 1-$S(v_{id})$。因此，位发生改变的绝对率为：

$$p(\Delta) = S(v_{id})(1 - S(v_{id})) \tag{6-3}$$

这等价于：

$$p(\Delta) = S(v_{id}) - S(v_{id})^2 \tag{6-4}$$

这就是当位为给定的一个速度 v_{id} 值时，其发生改变的绝对率。所以，v_{id} 值变化其实就是位改变率的变化。

结合(6-1)式和(6-4)式，得到下式：

$$p(\Delta)=\frac{1}{1+\exp(-v_{id})}-(\frac{1}{1+\exp(-v_{id})})^2 \tag{6-5}$$

(6-5)式是位的速度与位改变绝对率之间关系，其关系图见图 6-3。根据图 6-3，当位速度为 0 时，位的改变绝对率最大，最大值为 0.25。

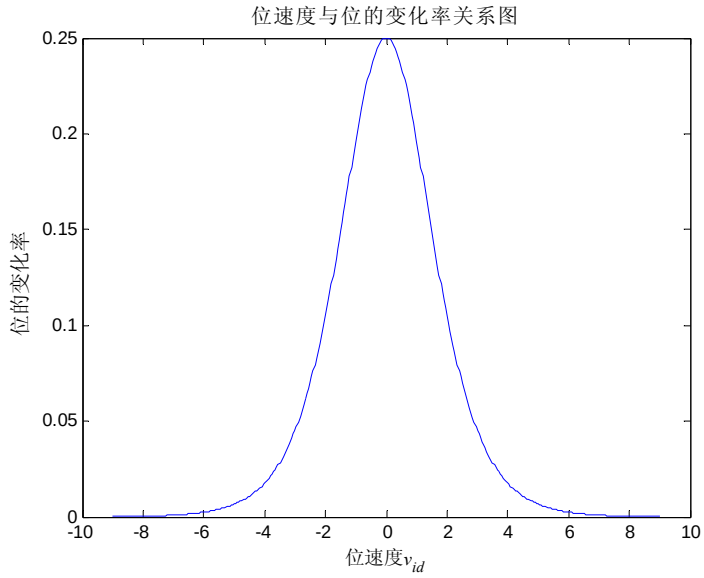

图 6-3　位速度与位的变化率之间关系

上面分析是 J. Kennedy 和 R. Eberhart 在文献[53] 分析结果，其分析方式不严格，比较简单，下面进一步分析。

6.3.2 位改变率的进一步分析

设 $v_{id}(t)$表示第 i 粒子的第 t 代在 d 维的速度。根据前面的分析，第 t 代位为 1 的概率为 $S(v_{id}(t))$，而位为 0 概率是 1-$S(v_{id}(t))$。因此，如果第 t-1 代位已经是 0，则第 t 代将发生改变的概率为 $S(v_{id}(t))$；同样，如果第 t-1 代位已经是 1，则第 t 代发生改变的概率为 1-$S(v_{id}(t))$。而第 t-1 代位是 0 的概率为 1-$S(v_{id}(t\text{-}1))$，第 t-1 代位是 1 的概率为 $S(v_{id}(t\text{-}1))$。因此，第 t 代位发生改变的概率 p(t)应该为：

$$p(t)=(1-S(v_{id}(t-1)))S(v_{id}(t))+S(v_{id}(t-1))(1-S(v_{id}(t))) \tag{6-6}$$

结合(6-1)式与(6-6)式，得到下式:

$$p(t)=\left(1-\frac{1}{1+\exp(-v_{id}(t-1))}\right)\left(\frac{1}{1+\exp(-v_{id}(t))}\right)+\left(\frac{1}{1+\exp(-v_{id}(t-1))}\right)\left(1-\frac{1}{1+\exp(-v_{id}(t))}\right) \tag{6-7}$$

(6-7)式就是第 t 代的位改变概率，其与速度的关系如图 6-4。根据图 6-4，第 t 代的位改变概率与两代的速度有关，但其最大的改变概率应该在两代速度都为 0 时，且最大值为 0.5。也就是说，最大的位改变概率不超过 0.5。作一个特殊假设，假设

$v_{id}(t)=v_{id}(t-1)$，则(6-6) 式变为：

$$p(t)=2(1-S(v_{id}(t)))S(v_{id}(t)) \tag{6-8}$$

再结合(6-7)式，则得下式：

$$p(t)=2\left(1-\frac{1}{1+\exp(-v_{id}(t))}\right)\left(\frac{1}{1+\exp(-v_{id}(t))}\right) \tag{6-9}$$

(6-9)式的演示图为图 6-5，其图形类似一个抛物线，最大值点在速度 $v_{id}(t)$为 0 时，最大值为 0.5。

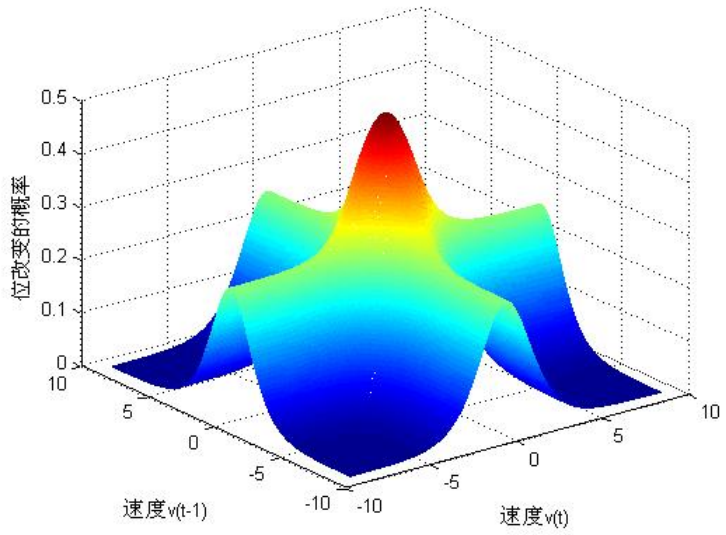

图 6-4　位改变的概率

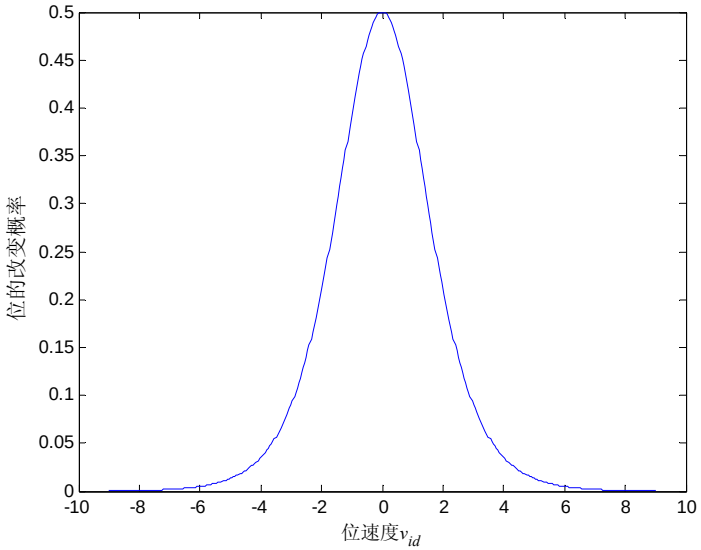

图 6-5　位改变的概率与速度关系

这个结论与文献[53] 的分析结果有差别。当速度 $v_{id}(t)$为 0 时，按文献[53] 的分析，位改变的率是 0.25，而这里得到结果为 0.5。这个结论在后面的实验中可以得到检验。

离散二进制 PSO 算法速度更新公式为(3-1)式，即

$$v_{id} = \omega \cdot v_{id} + c_1 \cdot rand() \cdot (p_{id} - x_{id}) + c_2 \cdot rand() \cdot (p_{gd} - x_{id}) \quad (3\text{-}1)$$

这里 p_{id}、x_{id} 及 p_{gd} 分别是二进制位，它们的取值为 0 或 1。因此，（p_{id}–x_{id}）和（p_{gd}–x_{id}）可能为-1，0，1。所以 $c_1 \cdot rand() \cdot (p_{id} - x_{id}) + c_2 \cdot rand() \cdot (p_{gd} - x_{id})$ 的值为区间[-(c_1+c_2)，(c_1+c_2)]的随机值。

假如(3-1)式只有一项，即(3-1)式改为 $v_{id} = c_2 \cdot rand() \cdot (p_{gd} - x_{id})$。此时，如果 p_{gd}=x_{id}，则 v_{id}=0，$S(v_{id})$=0.5，这意味位取 1 和 0 的概率相同。根据(6-7)式，$p(t)$=0.5，意味着粒子的位发生改变的概率为 0.5。当粒子的位置都收敛到全局最优粒子位置 p_{gd}，则 v_{id}=0，即粒子的位改变概率为 50%，达到最高。

根据上面的分析，得到以下两点结论：

(1).位改变率与速度的变化式应该为(6-7)式或(6-9)式，而不是(6-4)式。当速度 $v_{id}(t)$ 为 0 时，按文献[53] 分析，位改变的率最大值是 0.25，而这里得到结果为 0.5。这个结论与 J. Kennedy 和 R. Eberhart 在文献[53] 分析结果是有差别的，这可以在后面的实验中可以得到检验。

(2).离散二进制 PSO 算法的粒子不太可能收敛于全局最优粒子。如果其收敛到全局最优粒子，则其速度为 0。此时，位发生改变机会最大，即为 0.5，搜索更加随机性，没有方向性。所以，可以得出的结论是：离散二进制 PSO 算法是全局性随机搜索算法，算法随迭代运行，其随机性更强，没有收敛性。因此，离散二进制 PSO 算法缺少局部探测性。

6.3.3 实验验证

为了验证上节分析的结论，这里用实验来检验。实验利用二进制粒子群算法来求解基准函数 *J.D.Schaffer* 函数，其函数形式如下:

$$f_2(x) = \frac{\sin^2 \sqrt{\sum_{i=1}^{n} x_i^2} - 0.5}{(1 + 0.001 * (\sum_{i=1}^{n} x_i^2))^2} + 0.5 \qquad -5.12 < x_i < 5.12$$

实验在二维空间上利用离散二进制 PSO 算法计算函数的最小值。首先对搜索空间每一维编码，每一维共 2^{10} 位二进制数字表示，迭代运行步数为 1000，粒子数为 20，最大限制速度 V_{max}=9。每个粒子有二维，每一维区间[-5.12, 5.12]被编码成 2^{10} 位。经仿真演示，粒子位的平均速度随迭代变化如图 6-6，其粒子各位取 1 的平均概率随迭代变化如图 6-7，而图 6-8 是某粒子位改变率的迭代变化。对一个粒子每位其可能改变，也可能不改变，一个粒子共 2×2^{10} 二进制位，每次迭代计算其位改变数，再除以其 2×2^{10} 二进制位，得到粒子的位平均改变率。

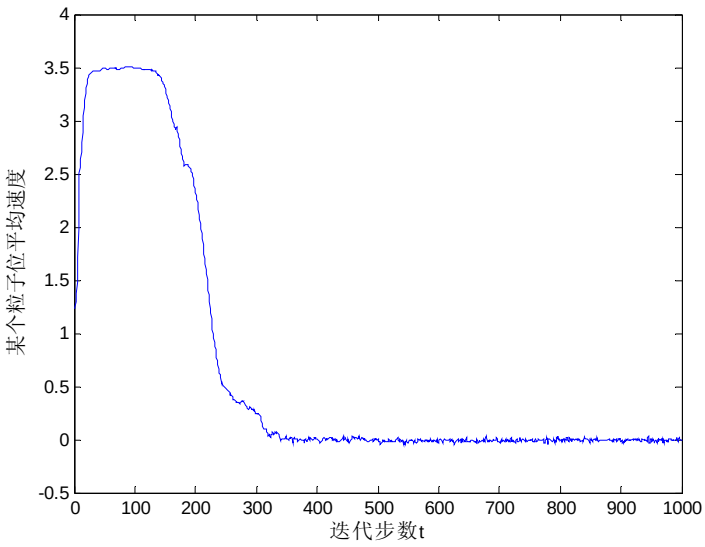

图 6-6　某粒子的平均速度迭代变化

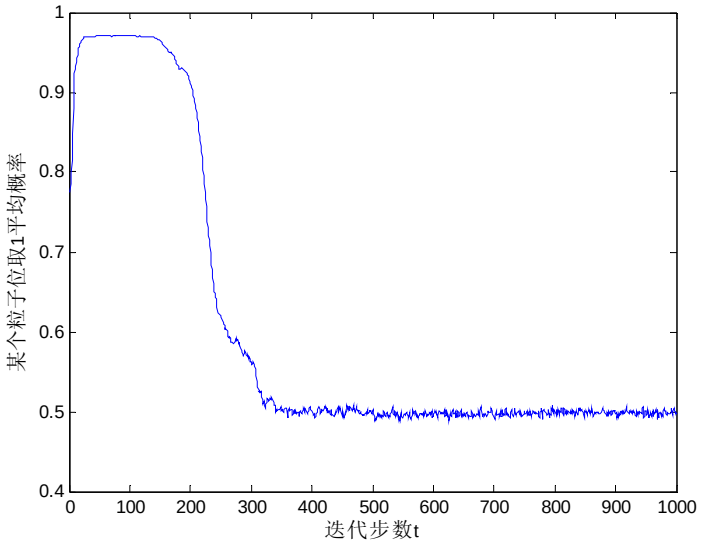

图 6-7　某粒子位取 1 的平均概率迭代变化

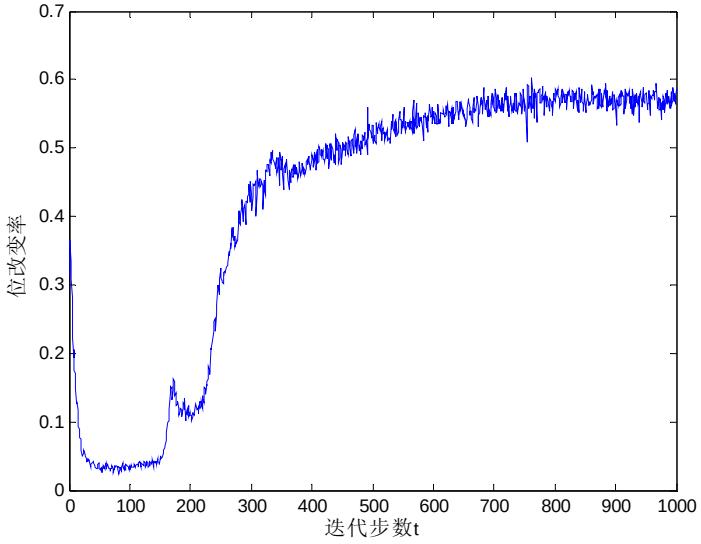

图 6-8　某粒子位改变率的迭代变化

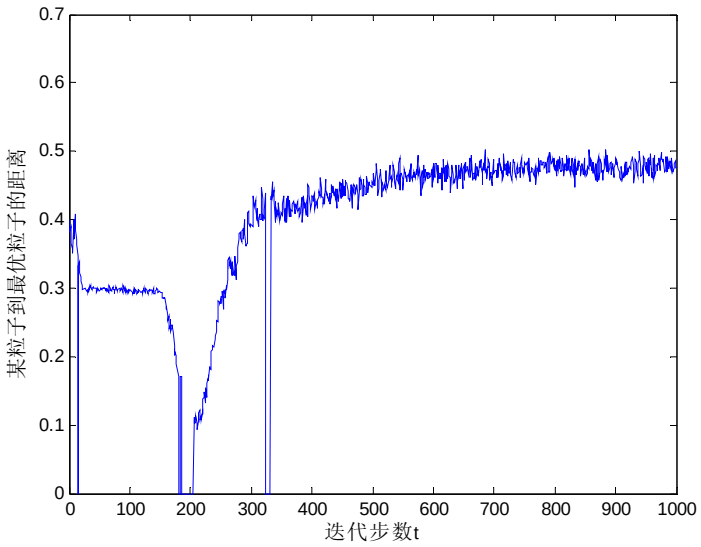

图 6-9　某粒子到最优粒子平均距离的迭代变化

图 6-6 显示粒子的速度在减小，并慢慢地趋向一个固定值上下波动。图 6-7 表明粒子取 1 的平均概率也具有图 6-6 类似的特点。图 6-8 更好地反映(6-7)式及图 6-4 的特征，从图 6-8 可以看出，位的改变是在慢慢增加，最后位改变概率靠近 0.5 左右；图 6-6 显示速度随迭代而靠近 0。再结合图 6-8 可知：当速度趋靠近 0 时，速度改变率基本靠近 0.5 左右。这些图的结果验证了对图 6-4 及(6-7)式的分析结论。图 6-9 显示某粒子到最优粒子的平均距离的迭代变化情况，从图形式可以看到：粒子与最优粒子距离是慢慢增加，而没有趋向于 0，图 6-9 的图形反应了粒子不收敛于全局最优粒子，粒子随着迭代进行越来越具有随机性，少了局部探测性。

6.3.4 速度期望值的分析

为了分析位改率随迭代步数 t 如何变化，另一种方法是分析 BPSO 算法速度的动态性，也就是分析速度向量的行为。现考虑一个位(bit)：假设自身最优和全局最优解设置为 1，并且这两个解不变，那么根据速度变换公式(3-1)式，它的速度值 v 被引向最高上限需要速度一直增加，并且速度增加必须需要解的位被设置为 0。这种情况发生的概率是：

$$p(x_b=0)=1-s(v)=1-\frac{1}{1+e^{-v}}=\frac{1}{1+e^{v}}$$

但从上式可以看到，这种概率是随着速度的增加而减小。因此，速度越靠近边界，其靠近最优解 1 的难度越大。只要速度值不是太靠近边界值，二进制 PSO 的搜索很随机，以致于不能用很高的概率发现单个最优解。

为了更好地分析位变化的随机性，这里采用期望值的方法分析速度变化，以便去掉速度公式的随机因素，使分析更加方便。

假设 $E(v(t))$表示迭代 t 时速度 $v(t)$的期望值。根据(3-1)式可得下式：

$$E(v(t+1)) = \omega \cdot E(v(t)) + E(c_1 \cdot rand() \cdot (p - x(t)) + c_2 \cdot rand() \cdot (p_g - x(t)))$$

为了叙述方便，这里不考虑粒子下标 i 及维数下标 d。并假定自身最优粒子的位 p 和全局最优粒子的位 p_g 是固定不变的。

因为 $rand(\ )$是在区间[0，1]内统一分布，并且 $x(t)$取 1 和 0 的概率分别是：

$$p(x(t)=1) = s(v(t)) = \frac{1}{1+e^{-v(t)}}$$

$$p(x(t)=0) = 1 - s(v(t)) = 1 - \frac{1}{1+e^{-v(t)}} = \frac{1}{1+e^{v(t)}}$$

所以，有

$$E(v(t+1)) = \omega \cdot E(v(t)) + E(c_1 \cdot rand() \cdot (p - x(t)) + c_2 \cdot rand() \cdot (p_g - x(t)))$$

$$E(v(t+1)) = \omega \cdot E(v(t)) + E(c_1 \cdot rand()) \cdot E(p - x(t)) + E(c_2 \cdot rand()) \cdot E(p_g - x(t))$$

$$E(v(t+1)) = \omega \cdot E(v(t)) + \frac{1}{2} c_1 \cdot E(p - x(t)) + \frac{1}{2} c_2 \cdot E(p_g - x(t))$$

$$E(v(t+1)) = \omega \cdot E(v(t)) + \frac{1}{2} c_1 \cdot p + \frac{1}{2} c_2 \cdot p_g - \frac{c_1 + c_2}{2} \cdot E(x(t))$$

而, $E(x(t)) = 1 \cdot p(x(t)=1) + 0 \cdot p(x(t)=0)$

$$= \frac{1}{1+e^{-E(v(t))}}$$

因此，得：

$$E(v(t+1)) = \omega \cdot E(v(t)) + \frac{c_1 \cdot p + c_2 \cdot p_g}{2} - \frac{c_1 + c_2}{2} \cdot \frac{1}{1+e^{-E(v(t))}} \tag{6-10}$$

上式是一个关于 $E(v(t))$的非线性差分方程，难以直接求其解，利用数值仿真图形分析 $E(v(t))$的迭代变化情况。

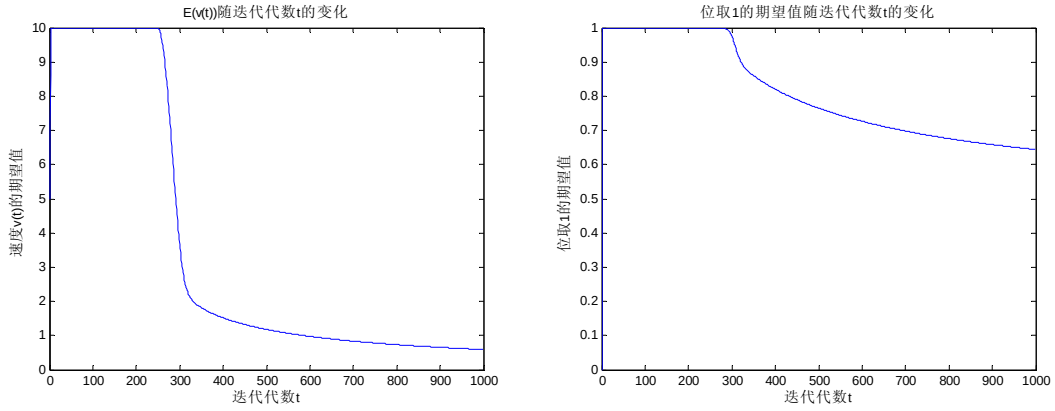

图 6-6　当 p= 1 ， p_g= 1 时，速度期望 E(*v*(*t*))及位取 1 的概率迭代变化

仿真中，参数 w 的值从 1.2 递减到 0.4，而 c_1 和 c_2 分别取值 2，p 和 p_g 分别取值

0 或 1，初始速度 $E(v(0))$=5，速度的最大值 v_{max}=10。下面分三种情况来分析 $E(v(t))$ 值的迭代变化。当 p=1，p_g=1 时，图 6-6 演示位的速度期望 $E(v(t))$及相应的位取 1 的概率迭代变化；而当 p= 0 ，p_g= 0 时，图 6-7 演示其相应的结果；当 p=0，p_g=1 时，图 6-8 其相应的结果。

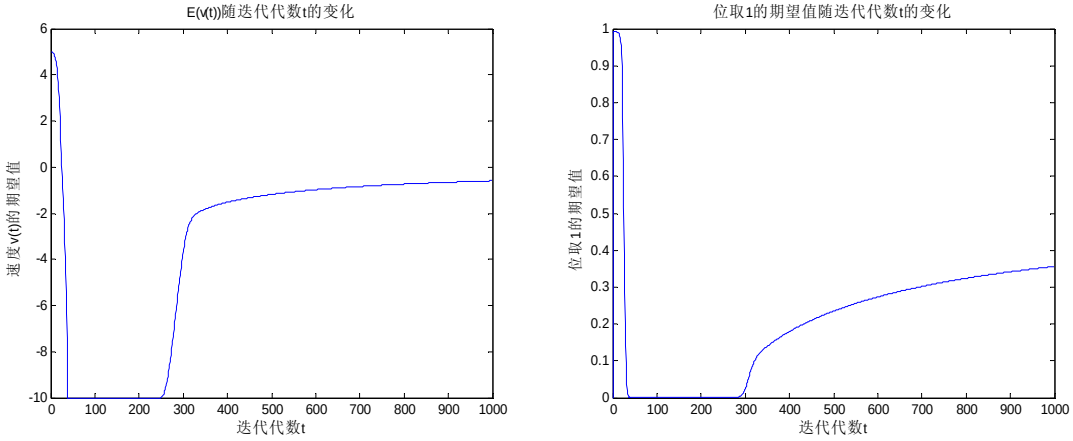

图 6-7　当 p=0, p_g=0 时, 速度期望 E($v(t)$)及位取 1 的概率迭代变化

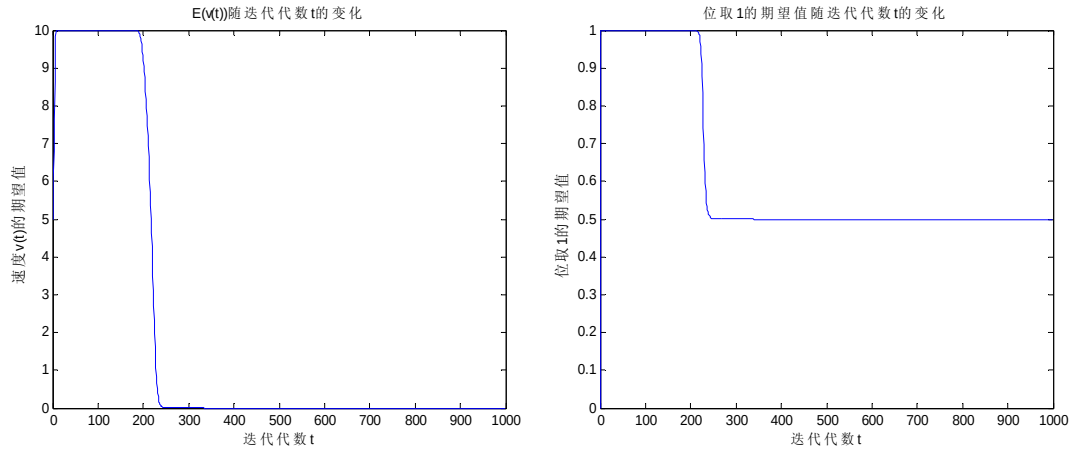

图 6-8　当 p=0, p_g=1 时, 速度期望 E($v(t)$)及位取 1 的概率迭代变化

根据上面三组图形，随着迭代代数增加，速度期望值向 0 靠近，其相应位取 1 的概率都趋向 0.5。因此，BPSO 算法随着迭代进化，粒子越来越不确定，其随机性在增加，粒子位置的取值变得更加随机。这种现象表明，BPSO 算法是一种全局搜索算法，缺少局部收敛和局部探测性，这与标准 PSO 的特性相反。这个分析结论与上一节的分析结论相吻合。因此，这节分析的结论从另外一个角度再次说明二进制 PSO 算法是一个带随机性偏强、缺乏局部搜索能力的离散性智能优化算法。

6.4 模式的分析

6.4.1 遗传算法模式定理

根据参考文献 [117]

设符号“*”取值为 0 或 1，即*∈{0, 1}，于是，字符串“*0001”代表了在基因位 2、3、4、5 位具有形式“0001”的所有字符串，即 00001 和 10001。这时，也称 00001 和 10001 按“*0001”具有结构相似性。

定义：基于字符串{0，1，*}产生的字符串，称作模式（schema）；符号“*”称作通配符或无关符。

例：模式 H=*1**0 描述如下集合内的字符串：

{01000，01010，01100，01110，1100}。

上述集合内的字符串按模式*1**0 具有结构相似性，也称具有模式 H。

定义：设 $H \in S$={0, 1, *}是一个模式，

(1)称 H 中确定位置的个数为该模式的阶，记为 $o(H)$;

(2)称 H 中第一个确定位置（简称模式位）到最后一个确定位置之间的距离为该模式的定义距，记为$\delta(H)$。

例：设模式 H=011*1*，则

$o(H)$=4，$\delta(H)$=4。

模式的阶和定义距描述了模式的基本性质。下面介绍遗传算法中 Holland 的模式定理(Schemata theorem)。记 $G(t)$是第 t 代种群，并假设在遗传过程中种群的规模，即种群中个体的数目不变，又记

$T(H，G(t))$={$A \in G(t)$| A 具有模式 H}

为 $G(t)$中具有模式 H 的所有个体集合。所谓个体 A 具有模式 H，是指个体 A 在 H 的基因上（除统配符*）对应的基因值相同，记

$m(H, t)$=| $T(H, G(t))$|，

即 $G(t)$中具有模式 H 的个体数目。

下面通过分析遗传算法的三种基本遗传操作对模式的作用来讨论模式定理。

(1). 选择算法

在选择算子作用下，与某一模式所匹配的样本数的增减依赖于模式的平均适用度，与群体平均适用度之比，平均适用度高于群体平均适用度的将呈指数级增长；而平均适用度低于群体平均适用度的模式将呈指数减少。其推导如下：

设 $G(t)$中个体的总数为 N，即$|G(t)|=N$，其个体编号依次为 1, 2, …, N。第 i 个个体 A_i 的适用应函数值为 $f_i=f(A_i)$, 选择概率为：

$$p_i = f_i / \sum_{j=1}^{N} f_j$$

其中 $f(x)$是适用度函数，称

$$\bar{f}=\frac{1}{N}\sum_{j=1}^{N}f_j$$

为 *G(t)*中个体的平均适用度，而称

$$f(H)=\frac{1}{m(H,t)}\sum_{A\in T(H,G(t))}f(A)$$

为 *G(t)*中所有具有模式 *H* 的个体的平均适用度。则有

$$m(H,t+1)=m(H,t)\frac{f(H)}{\bar{f}}$$

现在，假定模式 H 的平均适用度高于群体的平均适用度，且设高出部分为 $c\bar{f}$，c 为常数，则有

$$m(H,t+1)=m(H,t)\frac{\bar{f}+c\bar{f}}{\bar{f}}=(1+c)m(H,t)$$

假设从 *t*=0 开始，c 保持为常值，则有

$m(H, t+1)=m(H,0)(1+c)$

(2). 交叉算子

然而仅有选择算子，并不能产生新的个体，即不能对搜索空间中新的区域进行搜索，因此引入交叉操作。下面讨论模式在交叉算子作用下所发生的变化，这里我们只考虑单交叉的情况。

模式 H 只有当交叉点落在定义距之外才能生存在。在简单交叉（单点交叉）下 *H* 的生存概率 P_S=1- $\delta(H)/(t-1)$。例如，一个长度为 5 的串以及隐含其中的两个模式为

$$A=010110$$

$$H_1=*1***0$$

$$H_2=***11*$$

我们注意到模式 H_1 的定义长度为 4，那么交叉点在 6-1=5 个位置随机产生时，H_1 遭破坏的概率 P_d= $\delta(H_2)/(m-1)$=1/5，即生存概率为 4/5。

而交叉本身也是以一定的概率 P_c 发生的，所以模式 *H* 的生存概率为

$$P_s=1-P_c\cdot P_d=1-\frac{P_c\cdot\delta(H_2)}{m-1}$$

现在考虑交叉发生在定义距内，模式 *H* 不被破坏的可能性。在前面的例子中，若与 *A* 交叉的串在位置 2、6 上有一位与 *A* 相同，则 H_1 将被保留。考虑到这一点，上式给出的生存概率只是一个下界，即有

$$P_s\geq 1-\frac{P_c\cdot\delta(H_2)}{m-1}$$

可见，模式在交叉算子作用下定义短的模式将增多。

(3). 变异算子

假定串的某一位置发生改变的概率为 P_m，则该位置不变的概率为 1- P_m，而模式 H 在变异算子作用下若要不受破坏，则其中所有的确定位置(“0”或“1”的位)必须保持不变，因此模式 H 保持不变的概率为$(1-P_m)^{o(H)}$，其中 $o(H)$为模式 H 的阶数。当 $P_m<<1$ 时，模式 H 在变异算子作用下的生存概率为

$$P_s=(1-P_m)^{o(H)}\approx 1-o(H)P_m$$

综合上所述，模式 H 在遗传算子选择、交叉和变异的共同作用下，其子代的样本数为

$$m(H,t+1)\geq m(H,t)\frac{f(H)}{\bar{f}}[1-P_c\frac{\delta(H)}{l-1}][1-o(H)P_m]$$

上式中忽略了极小项 $P_c\cdot\delta(H)/(l-1)+o(H)\cdot P_m$。通过上式，可以得出模式定理：

定理：（模式定理）在遗传算子选择、交叉和变异的作用下，具有阶数低、长度短、平均适用度高于群体平均适用度的模式在子代中将以指数级增长。

统计学的研究表明：在随机搜索中，要获得最优的可行解，则必须保证较优解的样本呈指数级增长，而模式定理保证了较优的模式（遗传算法的较优解）的样本呈指数级增长，从而给出了遗传算法的理论基础。另外，由于算法总能以一定的概率遍历到解空间的每一个部分，因此在选择算子的条件下总能得到问题的最优解。

6.4.2 二进制 PSO 的模式分析

根据前面的分析，二进制 PSO 算法每一次迭代是对二进制串位进行改变。因此，从遗传算法角度来理解，二进制 PSO 采用一种特殊变异的操作，每一次迭代是对每一位求出其变异概率，从而对每一位进行变异。其变异概率就是位改变率。特殊性在于其求解变异概率时，根据当前的速度和上一代位为 0 或 1 的概率。首先利用速度(3-1)式，再根据位的速度值，利用(6-1)式再转变为位取 1 的概率。

PSO 算法在于速度是根据粒子靠近最优点和自身历史位置的距离来求解。因此，二进制粒子变异与最优点和历史自身最优点有关。但从上面分析可以知，粒子越靠近最优点时，其速度越靠近 0，位变异率也就越高。当速度为 0 时，位变异概率高达到 0.5。

根据遗传算法变异算子性质，模式 H 保持概率为：

$$P_s=(1-P_m)^{o(H)}\approx 1-o(H)P_m$$

P_m 为位变异概率，o(H)为模式的阶。从上式中分析可，位变异概率 P_m 越小，模式 H 保持不变的概率越大。从二进制编码的进化算法理论来说，变异有利于种群的

多样性，增强算法全局探测能力。而算法早期应该具有种群多样性，即增强算法的全局探测能力。但在晚期算法的种群多样性应该减弱，有利于算法局部探索。因此，进化算法变异概率早期应该大于晚期，才有利于算法性能提高。

但根据上一节分析，当二进制 PSO 算法速度越靠近 0 时，其位变异的概率越高。PSO 算法思想是粒子在飞行中，粒子越来越靠近最优粒子，也就是速度慢慢收敛于 0。二进制 PSO 算法的变异概率越来越大，种群多样性越来越高，全局探测能力越来越增强，缺乏局部探索性。这也说明二进制 PSO 的缺乏局部探测性的缺点，所以在利用二进制 PSO 算法求解离散的组合优化问题时，需要混合一些局部搜索算法。

此外，从遗传算法理论原理分析来看，二进制 PSO 算法只是利用了变异算子，其缺少选择操作和交叉操作，并不能保持最优模式呈指数级增长。

6.5 改进二制 PSO 算法

分析速度公式(3-1)式，为了叙述方便，重新列出如下：

$$v_{id} = \omega \cdot v_{id} + c_1 \cdot rand() \cdot (p_{id} - x_{id}) + c_2 \cdot rand() \cdot (p_{gd} - x_{id}) \tag{3-1}$$

上面公式分为三部分，第一部分 $\omega \cdot v_{id}$ 是速度惯性部分，而第二部分 $c_1 \cdot rand() \cdot (p_{id} - x_{id})$ 和第三部分 $c_2 \cdot rand() \cdot (p_{gd} - x_{id})$ 分别为粒子的“自身认知”部分和“社会认知”部分。而 $(p_{id} - x_{id})$ 和 $(p_{gd} - x_{id})$ 取值分别为 $-1, 0, 1$。0 意味着 p_{id} 或 p_{gd} 与 x_{id} 位的值相等；−1 意味着 p_{id} 或 p_{gd} 为 0, 而 x_{id} 的值为 1；1 意味着 p_{id} 或 p_{gd} 为 1，而 x_{id} 的值为 0。推广上面的分析，速度 v_{id} 的值可以这样理解：速度为 0 意味着粒子当前的位值与最优粒子或历史最优位值可能相等，位应该不需要变化。而速度为负，最优粒子和历史最优位置最大可能是为 0，粒子当前的位值为 1 可能性大，位需要转变为 0；而速度为正，最优粒子和历史最优位置最大可能是为 1，粒子的位值为 0 可能性大，位需要转变为 1。这样的思想有利于粒子越来越趋向最优粒子，算法收敛于全局最优粒子。

6.5.1 公式的变换

为了让速度和位改变的关系与上小节的思想分析相符，需要对(6-1)式进行改进。当速度为 0，希望新的概率映射函数值为 0；当速度小于 0 与大于 0 时，概率映射函数关于 y 轴对称，即为偶函数；当速度趋向正负无穷时，概率映射函数值为 1。直接对 sigmond 函数改为如下：

$$s(v_{id}) = \begin{cases} 1 - \dfrac{2}{1 + \exp(-v_{id})} & \text{当} v_{id} \le 0 \\ \dfrac{2}{1 + \exp(-v_{id})} - 1 & \text{当} v_{id} > 0 \end{cases} \tag{6-11}$$

(6-11)式的函数图形如图 6-9。根据图 6-9，当速度为负时，(6-11)式概率映射函数递减；当速度为正时，(6-11)式概率映射函数递增；当速度为 0 时，函数为 0。

接着，对粒子的位值改变式(6-2)改为如下形式：

当$v_{id}<0$时

$$x_{id}=\begin{cases}0 & if\ rand()\le s(v_{id})\\ x_{id} & otherwise\end{cases} \tag{6-12}$$

当$v_{id}>0$时

$$x_{id}=\begin{cases}1 & if\ rand()\le s(v_{id})\\ x_{id} & otherwise\end{cases} \tag{6-13}$$

同样，这里 *rand*( )是一个随机数，从区间[0，1]的统一分布中随机产生。为了避免 $s(v_{id})$太靠近 1，一个参数 V_{max} 作为速度最大值，用于限制 v_{id} 的范围，即 $v_{id}\in[-V_{max}, V_{max}]$。

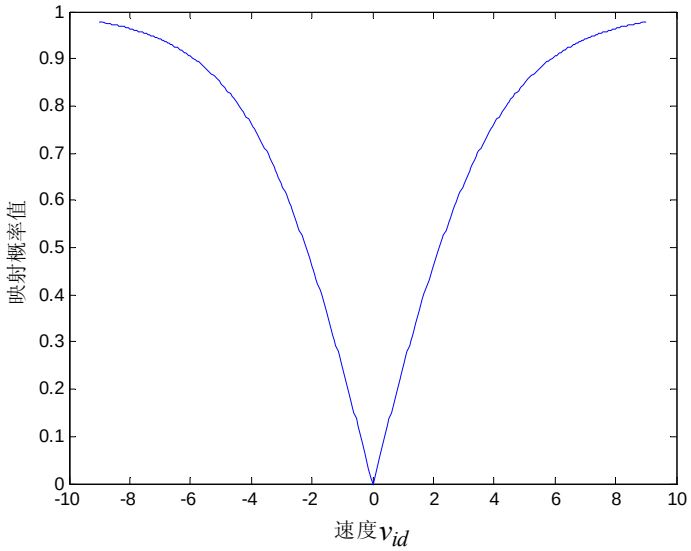

图 6-9　新概率映射函数

新方法与原方法区别在于两个方面。首先，新的概率映射函数与 sigmond 函数存在差别，图 6-1 与图 6-9 作一个比较，就很清楚。新方法的目的当让速度趋为 0 时，概率映射函数值为 0。其次，根据概率函数值让位取 0 和 1 形式为(6-13)式和(6-13)式，这种形式能保证速度为 0 时，位的值不变；当速度为负时，位只可能改变为 0；当速度为正时，位可能改变为 1 。这样做目的，可以让粒子群最终很容易靠近全局最优粒子，当速度为 0 时，粒子的位改变率靠近 0 的可能性增大。这种思想符合粒子群算法本质思想。

6.5.2 实验演示

为了说明上述改进的意义，用实验来检验上述改进的思想方法。同样，利用新的二进制粒子群算法来求解基准函数 *J.D.Schaffer* 函数，算法流程与原始二进制粒子群算法类似，速度映射函数与位改变公式变为新改进的形式(6-11)和(6-12)式。

在二维上，求基准函数 *J.D.Schaffer* 的最小值。首先对搜索空间每一维编码，每一维共 2^{10} 位二进制数字表示，迭代运行步数为 300，粒子数为 20，最大限制速度 V_{max}=9。每个粒子有二维，每一维区间[-5.12, 5.12]被编码成 2^{10} 位，实验工作与前面试验类似。某个粒子的位平均速度随迭代计算如图 6-10，其粒子各位取 1 的平均概率的迭代变化如图 6-11。图 6-12 是某粒子的位改变率的迭代变化。对一个粒子的每一位可能改变，也可能不改变，一个粒子共 2×2^{10} 二进制位，每次迭代计算其位改变数，再除以其 2×1024 二进制位，得到粒子的位改变率。

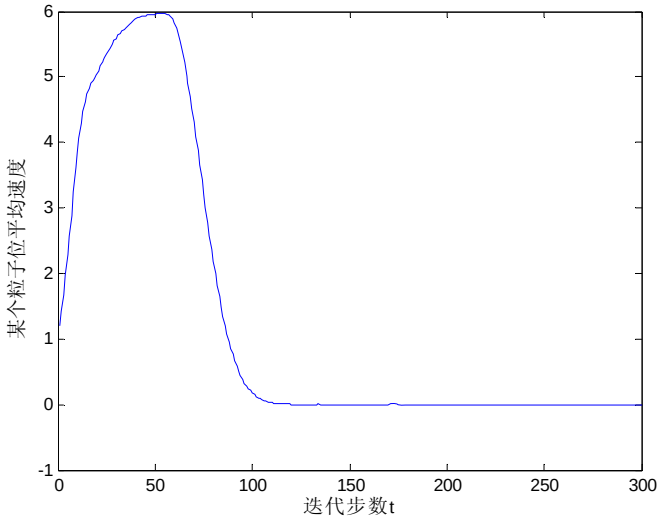

图 6-10　某粒的平均速度迭代变化

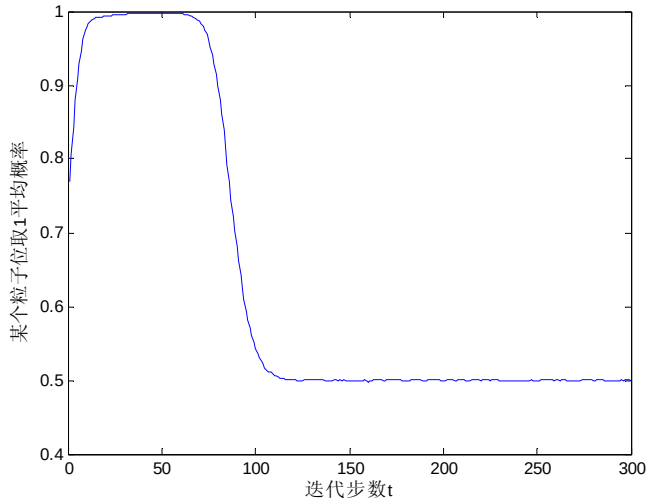

图 6-11　某粒子的取 1 平均概率迭代变化

图 6-10 表明粒子的速度在减小，并慢慢地趋向于 0。因此，粒子的位置很快收敛。 根据图 6-13，粒子到最优粒子的距离也慢慢趋向于 0，所以粒子的位置在靠向最优粒子。图 6-11 显示粒子取 1 的平均概率慢慢靠近 0.5，这意味粒子的位为 1 或 0 的概率各为 50%。图 6-12 粒子的位改变率慢慢趋向 0，这也意味着粒子在慢慢稳定，并且到了晚期，粒子的位置并没有多少变异发生。结合以上分析，本节改进方法的思想符合粒子群算法的思想，让粒子能够慢慢收敛于最优粒子的位置。从图 6-12 可以看出，位的改变是在减少，位改变概率非常低，最终不会高于 0.05。

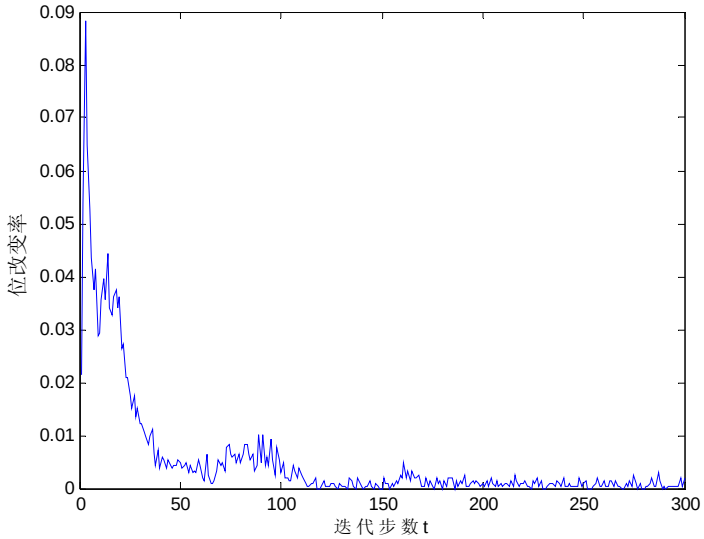

图 6-12　某粒子的位改变率的迭代变化

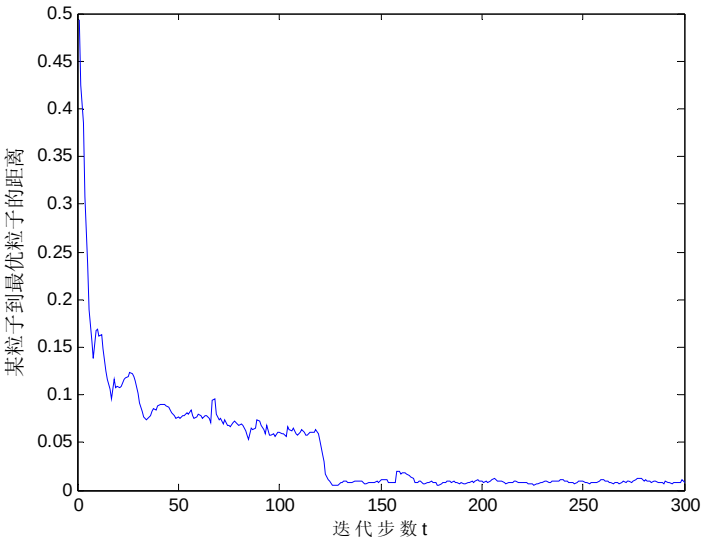

图 6-13　某粒子到最优点粒距离的迭代变化

用(6-12)式和(6-13)式代替原始 BPSO 算法的(6-1)和(6-2)式，粒子会很快收敛于全局点而不动，算法效果不理想。从上面分析可以知，原始 BPSO 算法运行到最后时，更具有随机性，这时说明它具有全局搜索能力，不具有局部搜索性。而新公式(6-12)

式和(6-13)式刚好与其相反，收敛很快，局部搜索能力很强，而全局搜索能力很弱。根据一般的启发式随机搜索算法原理，算法早期需要全局搜索能力，而晚期需要局部搜索能力。根据这个道理，构造如下混合算法：

If　$iter<\gamma\times MaxIter$

　　算法利用(6-1)式和(6-2)式

Else

　　算法采用新变公式(6-11)式和(6-12)式

End

这里γ是一个参数，其取值为[0，1]之间实数，*iter* 为当前运行的迭代步数，而 *ItMax* 为算法迭代运行的最大步数。当其为 1 时，算法就是原始 BPSO 算法；当为 0 时，其完全是新的变换公式。把这种算法叫做新的二进制粒子群算法(New Binary Particle Swarm Optimization，简称为 NBPSO)。

6.6 BPSO 求解背包问题

为了比较新的二进制粒子群算法与原始的二进制粒子群算法的效果，用这两种算法求解一个典型的组合优问题——0/1 背包问题。背包问题有着广泛的实际应用背景, 如预算控制、项目选择、材料切割、货物装载等问题，但是在计算理论中属于 NP 难问题。对该问题的求解，目前已有的求解方法分为动态规划等精确算法, 贪心法等近似算法和遗传算法、蚁群算法等仿生优化算法。背包问题能很好用 0/1 编码方式地表示，很适用于二进制粒子群算法来求解。

6.6.1 背包问题描述

已知 n 个物品的重量及其价值分别为 w_i 和 c_j ($j=1,2,...,n$)，如何将它们装入总容量为 M 的背包，使得所选物品的总价值最大，设变量

$$x_j=\begin{cases}0 & \text{不选择物品}j\\ 1 & \text{选择物品}j\end{cases}\qquad (j=1,2,...,n) \tag{6-14}$$

则该问题的模型可表示为：

$$\max f(x)=\sum_{i=1}^{n}c_i x_j \tag{6-15}$$

$$s.t.\begin{cases}g(x)=\sum_{j=1}^{n}w_j x_i-M\le 0\\ x_j\in\{0,1\}\qquad (j=1,2,...,n)\end{cases} \tag{6-16}$$

其中 $x=(x_1, x_2, \dots, x_n)$

6.6.2 求解背包问题的 BPSO 算法描述

应用二进制粒子群算法解决背包问题的关键是如何编码。这里用 x_i 表示第 i 个粒子的位置值，每一个粒子位置 x_i 表示成背包问题的一个解。x_i=[x_{i1}, x_{i2}, …, x_{in}], n 表示粒子的维数，在背包问题中代表物品数量。x_{ij} 的值表示第 i 粒子是否选择物品 j，其取值为 0 和 1。如里 x_{ij}=1 表法第 i 个粒子将物品 j 装入背包中，否则物品 j 没有被装入背包。粒子速度 v_i 将映射成概率，即物品将放入背包的概率，其中 v_i=[v_{i1}, v_{i2}, …,v_{in}]。

粒子的目标函数为：

$$f(x_i)=\sum_{j=1}^{n}c_j x_{ij}+Q\min\{0,V-\sum_{j=1}^{n}w_j x_{ij}\} \tag{6-17}$$

上式 Q 是一个惩罚因子，其取值为一个充分大的正数。这样，由(6-15)式和(6-16)组成的约束整数规则变成由(6-17)式组成的非约束整数规则。注意：这里是求解目标函数(6-17)的最大值，而不通常的求解最小值。

粒子的位置由下式初始化：

$$x_{ij}=\begin{cases}0 & rand()<0.5\\ 1 & \text{否则}\end{cases} \tag{6-18}$$

其中，$rand(\ )$是在[0, 1]之间随机产生的数。

粒子的初始速度 v_{ij} 按下式随机初始化：

$$v_{ij}=v_{\min}+rand()(v_{\max}-v_{\min}) \tag{6-19}$$

其中 v_{max} 和 v_{min} 表示速度的最大最小限制值。

NBPSO 求解背包问题的算法过程描述如下：

算法过程描述：

Step1：初始粒子群：采用二进制编码表示背包问题的候选解，按(6-18)式随机产生 n 个粒子；按(6-19)式随机产生速度；

Step2：计算每个粒子的适应值：按(6-17)式计算每一个粒子的目标函数值；

Step3：更新个体最优值及全群最优：与现有各粒子的目标函数作比较更新个体最优和全局最优；

Step4：计算速度：对每个粒子的每位计算其速度，按(3-1)式计算；

Step5：产生新的粒子群：若迭代步数 *iter* 小于最大步数的 *MaxIter* 的 90%，则算法利用(6-1)式和(6-2)式产生下一代粒子，否则算法采用(6-11)式和(6-12)式产生下一代粒子

Step6：若迭代条件满足，再输出全局最优粒子的目标值。否则转入 Step2

6.6.3 实验与分析

实验采用网络上提供的七个背包数据(见 http://forums.cweek.com.cn/thread-893676-1-1.html)。每一背包数据共有 100 个物品，每一物品提供了其重量及其价值，七个背包最大容量都为 1000。对每一背包数据分别采用原始二进制粒子群算法和新的二进制粒子群算法计算，两种算法各运算 100 次，分别计算最好值、平均最优值及其方差。算法的参数设置为：粒子数为 20，最大进化代数为 1000，权重 w都为 1.4 到 0.2 线性递减，参数c_1与c_2都为 1.8。实验的计算结果如表 6-1。

表 6-1　两种算法计算背包问题的对比

背包	BPSO			NBPSO		
数据	最好值	平均最优值	方差	最好值	平均最优值	方差
Data1	2116	1849.9	336.34	2415	2115.9	384.71
Data2	2137	1836.6	333.93	2380	2086.6	379.38
Data3	2130	1800.2	327.30	2382	2066.7	375.76
Data4	2000	1747	317.63	2262	1972.4	358.61
Data5	1917	1571.5	285.73	2097	1838.5	334.28
Data6	2057	1685.9	320.69	2253	1921	349.27
Data7	2416	2044.2	371.66	2713	2345.9	426.52

表 6-1 显示的数据说明用改进后的二进制粒子群算法计算的结果要好于原始的二进制粒子群算法。根据两种算法计算的平均最优值与最好值，NBPSO 计算七个背包数据的结果都要强于 BPSO 算法计算的结果，NBPSO 算法具有更好的性能。

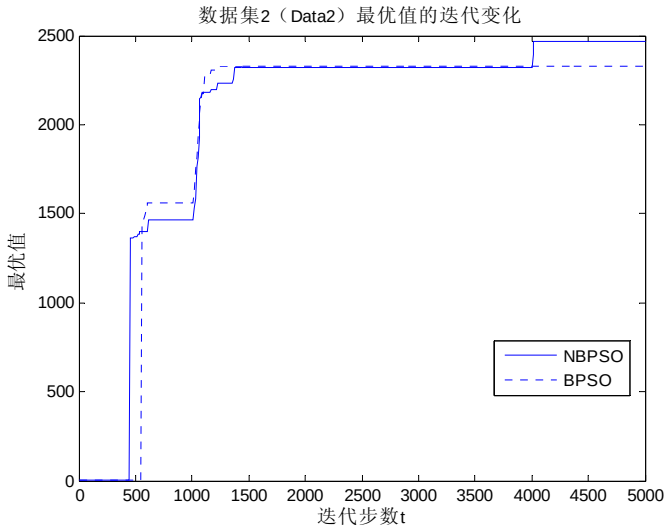

图 6-14　数据 Data1 最优值的迭代变化

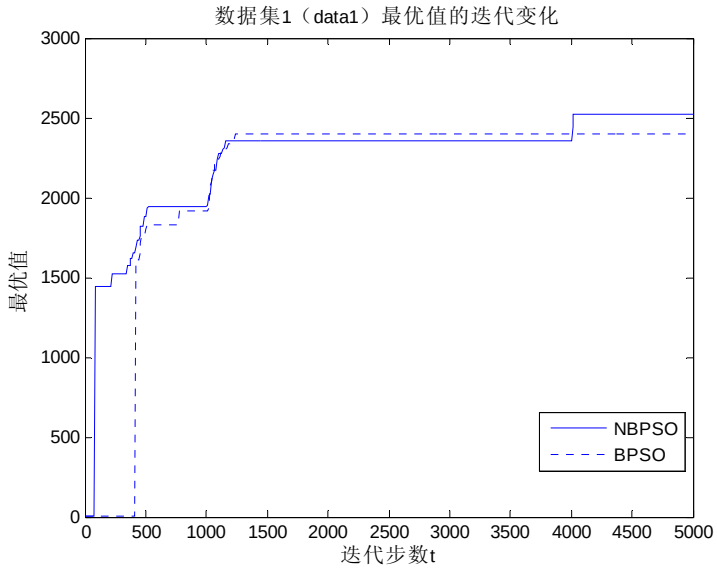

图 6-15　数据 Data2 最优值的迭代变化

为了更好地比较两种算法的性能，对两个背包数据 Data1 和 Data2 的计算过程用图形演示，即演示最优值随迭代步数递增而变化的情况，算法每次运行最大的迭代步数为 5000。结果得到图 6-14 和图 6-15。两个图形的结果说明，NBPSO 算法在迭代后期能再次寻找更优的值，而 BPSO 算法却过早停滞寻优。这是由于 NBPSO 算法在后期有局部寻优过程，而 BPSO 却没有。

6.7 本章小结

本章对二进制 PSO 算法从位改变率、速度期望值和遗传算法的模式概念等三方面作分析。三方面的分析结论基本一致，那就是原始二进制 PSO 算法随机性很强，具有较强的全局探索能力，但不会收敛到全局最优粒子。随着迭代运行，BPSO 算法的随机性反而会增强。因此，离散二进制 PSO 算法是一个缺乏局部探测性的算法。这个结论能指导对二进制 PSO 算法的改进。基于上述分析结论，本章提出一种改进原始离散二进制 PSO 算法方法，即改变其概率变换公式及位的取值公式。新的公式有助于粒子收敛于群体最优粒子，增强局部探测性。本章最后为了比较新的二进制 PSO 算法与原始的二进制 PSO 算法的性能，用两种算法求解背包问题。实验的结果表明新改进的算法能提高原算法的性能。

第七章 结论

粒子群算法在仿真生物群体社会活动的基础上，通过模拟群体生物的相互协同寻优能力，从而构造出一种基于群体智能的优化算法。但是，由于粒子群算法本身来源于生物群体现象，其理论基础尚不完备，且其标准算法形式也存在性能的缺陷，而标准PSO算法主要应用于连续搜索区域。因此，对标准PSO算法的理论分析、算法改进及离散问题分析具有重要意义的。本文在前人工作的基础上对标准PSO算法和离散二进制PSO算法进行了理论分析与改进。

1．结论

(1).粒子群算法在随机搜索过程中最终会收敛于群体最优粒子。本文在增加随机性和粒子最优点更新的条件下，理论上证明了粒子的轨迹是收敛于群体最优粒子，并且进一步说明算法权重选择的合理性。本文根据优化理论分析得出：当去掉随机性时，PSO 算法计算单峰函数时，粒子轨迹最终会收敛全局最优位置 p_g。而当计算多峰函数时，粒子最终未必收敛。但如果增加随机性，目标函数无论是单峰函数还是多峰函数，粒子都会收敛于最优位置 p_g。本文的分析结果说明了 PSO 算法的随机性及动态性给算法带来的效果。与已有的分析方法相比，本文的分析方法考虑了算法的随机量和粒子最优位置的动态性。

(2).本文定义了一个粒子间的相似度概念，通过计算群体粒子与其最优粒子的平均相似度，得到一个聚集度概念，用来度量群体粒子的多样性程度。每个粒子根据群体聚集度及其与群体最优粒子的相似度，随机产生变异，构造了一种改进的PSO算法。仿真结果表明本文改进的PSO算法在求多峰函数时，算法的收敛效果与收敛速度都有所提高。

(3).本文根据粒子与群体最优粒子的相似度，对不同粒子赋予不同权重，使每个粒子权重不同，并且每个粒子的权重随着不同的时间迭代动态变化，由此构造一种权重动态变化的粒子群算法。通过对基准测试函数的仿真实验，本文提出的权重动态变化粒子群算法相比标准粒子群算法的性能有所提高。

(4).本文利用PSO算法的数学模型代替标准PSO算法速度及位置迭代公式，并构造适当的参数，得到一种新的进化算法。仿真试验检验，新的进化算法效果不会差于标准PSO算法。新的进化算法形式更容易理解、更具有数学的直观性，而且参数构造少且容易分析，新算法形式能直接体现PSO算法的数学思想。本文利用新的算法求解单交叉口信号灯的时间分配。实验仿真结果表明，本文的算法在静态环境是有效的。

(5).本文从位改变率、速度的数学期望及遗传算法模式概念对离散二进制PSO算

法进行分析。分析结果发现，二进制PSO算法不收敛于群体最优粒子，其位值随着迭代运行而越来越随机，算法缺少局部搜索性。据此，本文提出改进原始PSO算法的方法，构造一种符合PSO算法思想（跟随群体最优粒子）新的二进制PSO算法。将两种算法运用于求解典型的二进制离散性问题——0/1背包问题。实验的结果比较表明，新改进的二进制PSO算法能提高原始二进制PSO算法的性能。

2．PSO算法存在问题及研究期望

PSO 算法的理论分析并不完善，本文只是说明了 $x(t)$收敛于 p_g，并没有说明 p_g 的轨迹是怎么变化，因此 p_g 轨迹变化是值得进一步分析。虽然 $x(t)$收敛于 p_g，但 p_g 最终是否收敛于目标最优点？（全局最优还是局部最优）。$x(t)$收敛于 p_g 时间性与目标问题的维数存在什么关系？这些问题是需要进一步分析研究。

正如 No Free Lunch 的理论所说[118] ，一种智能优化算法的性能好坏是与一定问题有关的。PSO 算法优化什么类型的问题才能体现其性能优势?回答这个问题应该更具有实际意义。

PSO 算法主要适用求解连续空间问题的优化，对于离散空间求解，目前也有不少应用研究。二进制离散 PSO 算法是求解离散空问题的一种方法，但还存在其他离散 PSO 算法来解决非二进制形式的离散问题，例如 TSP 问题。但各种方法目前只是处在算法的构造性及用实验验证的阶段，需要对其收敛性及时间性进行分析研究。

PSO 算法的理论形式为算法的理解提供数学模型。不同的智能优化算法应该都具有不同的数学模型，根据数学模型构造出新的算法具有更广的意义。这些研究结论可能为我们得到一些更优的随机优化算法。

PSO 算法本质是生物群体的协同能力，这协同能力为算法带来优化问题的性能，如同蚁群算法、免疫算法一样。但生物群体的协同特性不仅仅可以带来优化的性能，正如蚁群的社会合作特性可以构造聚类与分类的算法 [119] ，人体的免疫特性可能用来入侵检测一样 [120] ，PSO 算法的群体协同特性是否能带来另外的性能，这种性能也许能为智能学科带来新的新方法与思想。

参考文献

[1] 陈宝林．最优化理论与算法[M]．北京：清华大学出版社，2005. 1-466.

[2] 王凌．智能优化算法及其应用[M]．北京：清华大学出版社, 2001. 1-122.

[3] J H Holland. Adapatation in Nature and Artificial Systems[M] . USA:The University of Michigan Press, 1975.

[4] Farmer J D, Packed N H ,Perelson A S. The immune system , adapation and machine learning[J]. Physica. 1986, 22(2): 187-204.

[5] Dorigo M, Maniezzo V C A. The ant system: optimization by a colony of cooperating agents[J]. IEEE Transaction on System, Man and Cyternetics Part B, 1996, 26(1): 29-41.

[6] Kennedy J, Eberhart R. Particle Swarm Optimization[C]. In: Proceeding of IEEE International Conference on Neural Networks.Piscataway , NJ: IEEE CS, 1995: 1942-1948.

[7] Eberhart R, Kennedy J. A new optimizer using particle swarm theory[C]. In: Proceeding of the 6th International Symposium on Micro Machine and Human Science, 1995:39-43.

[8] Eberhart R, Shi Y. Comparision between Genetic Algorithm and Particle Swarm Optimization[M]. Lecture Notes in Computer Science, Springer Press, 1998: 611-616.

[9] Reynolds C W f. Herd and Schools: A Distributed Behavioral Model[J]. Computer Graphics, 1987, 21(4): 25-34.

[10] Heppner F, U Grenander. A stochastic nonlinear modle for coordinated bird flocks[M]. Washington D.C: AAAS Publications, 1990.

[11] 吴启迪，汪镭．智能微粒群算法研究及其应用[M]．江苏教育出版社，2005.

[12] Hu Xiaohui, Shi Yuhui, Eberhart R．Recent advances in particle swarm[C]． In: Congress on Evolutionary Computation (CEC2004)，2004：19-23.

[13] Shi Yuhui. Particle Swarm Optimization[J]. IEEE Connections, 2004, 2(1): 8-13.

[14] Jaca F Schutte, Albert A Groenwold. A Study of Global Optimization Using Particle Swarms[J]. Journal of Global Optimization, 2005, 31: 93-108.

[15] Ozcan E, Mohan C. Particle Swarm Optimization: Surfing the Waves[C]. In: Proceeding of 1999 Congress on Evolutionay Computation, IEEE CS, 1999: 1939 - 1944.

[16] Clerc M, Kennedy J. The Particle Swarm --Explosion, Stability and Convergence in Multidimesional Complex space[J]. IEEE Transaction on Evolutionary Computation, 2002, 6(1): 58-73.

[17] F.solis R W. Minimization by random search techniques[J]. Mathematics of Operations Research, 1981, 6(1): 937-971.

[18] Van de Bergh F. An Analysis of Particle Swarm Optimizer[D]. Ph.D, University of Pretoria, 2002.

[19] Van de Bergh F, A P Engelbrecht. A study of particle swarm optimization trajectories [J]. Information Sciences, 6, 17(6): 937-971.

[20] Trelea I C. The Particle Swarm Optimization Algorithms : Convergence Analysis and Parameter Selecion[J]. I nformation Processing Letters, 2003, 8(5): 317-325.

[21] Zheng Y, Ma L ,Zhang L.et.al. On the Convergence Analysis and Parameter Selection in Particle Swarm Optimization[C]. In: Proceeding of the Second International Conference on Machine Learning and Cybernetics, Xi'an, 2003: 1802-1807.

[22] Zhang L,Yu H, Hu S. Optimal Choice of Parameters for Particle Swarm Opimization[J]. Journal of Zhenjian University SCIENCE, 2005, 6A(6): 528-534.

[23] 潘峰，陈杰，甘明刚等 粒子群优化算法模型分析[J]. 自动化学报, 2006, 32(3): 368-375.

[24] 李宁. 粒子群优化算法的理论分析与应用研究[D]. [博士学位论文]. 华中科技大学, 2007.

[25] 李宁,孙德宝,邹彤,秦元庆等. 基于差分方程的PSO算法粒子运动轨迹分析[J]. 计算机学报, 2006,(11):2052-2061

[26] 金欣磊,马龙华,吴铁军等. 基于随机过程的PSO收敛性分析[J]. 自动化学报, 2007, 33(12): 1263-1268.

[27] 王俊伟,汪定伟. 粒子群算法中惯性权重的实验与分[J]. 系统工程学报, 2005, 20(2): 195-199.

[28] Shi Y, Eberhart R. Fuzzy Adaptive Particle Swarm Optimization[C]. In: Proceedings of the IEEE Conference on Evolutionary Computation.Koreal ,Soul: IEEE Press, 2001:103-106.

[29] 李宁,邹彤,孙德宝.带时间窗车车辆路径问题的粒子群算法[J].系统工程理论与实践, 2004, 24(4): 130-135.

[30] Chatterjee A, Siarry P. Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization[J]. Computers and Operations Research. 2006,

33(3): 859-871.

[31] Jiao B, Lian Z, Gu X. A dynamic inertia weight particle swarm optimization algorithm[J]. Chaos, Solitons and Fractals. 2008, 37(3): 698-705.

[32] R Eberhart, Y Shi. Comparing inertia weights and constriction factors in particle swarm optimizaiton[C]. In: Proceeding fo the IEEE Conference on Evolutionary Compuation, ICEC 2000,7:84-88.

[33] Blackwell T M. Particle Swarms and Populaiton diversity[J]. soft Computing. 2005, 9: 793-802.

[34] Riget J, Vesterstr J S. A diversity-guided particle swarm optimizer-the ARPSO[J]. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.2929. 2002, 2: 2002.

[35] Krink T, Vesterstorm J S, Riget J. Particle swarm optimization with spatial particle extension[C]. In: Proceeding fo the IEEE International 1th Conference on Evolutionary Computation. Honolulu: IEEE Inc., 2002:1474-1497.

[36] Kazemi BAL, Mohan C K. Multi-Phase generalization of the particle swarm optimization algorithm[C]. In: Proc. of the IEEE Int'l Conf. on Evolutionary Computation. Honolulu: IEEE Inc. 2002:489−494.

[37] Hu X, Eberhart R. Adaptive particle swarm optimization: detection and response todynamic systems[Z]. Honolulu, HI, USA: 2002: 2:1666-1670.

[38] Xie X F, Zhang W J, Yang Z L. A dissipative Particle Swarm Optimization [C]. In: Proceeding of the IEEE International Conference on Evolutionary Computation. Honolulu: IEEE Inc., 2002:1456-1461.

[39] Higashi N, Iba H. Particle Swarm Optimization with Gaussian Mutation[C]. In: Proceeding of the IEEE Swarm Intelligence Sympois.Indianapolis: IEEE Inc., 2003. 72-79.

[40] Kennedy J. Bare bones particle swarms[C]. In: Proceeding of the IEEE Swarm Intelligence Sympois.Indianapolis: IEEE Press, 2003. 53-57.

[41] Zhang W.J, Xie X.F. DEPSO: Hybrid particle swarm with differential evolution operator[C]. In: Proceeding of the IEEE International Conference on Systems, Man and Cybernetics.Washington: IEEE Press, 2003. 3816-1670.

[42] Lovbjerg M, Krink T. Extending Particle swarm optimizers with Self-organized critically[C]. In: Proceeding of the International Conference on Evolutionary Computionary.Honolulu: IEEE Inc., 2002. 1588-1593.

[43] Sun J, Feng B, Xu W. Particle swarm optimization with particles having quantum behavior[C]. In: Congress on Evolutionary Computation, CEC2004.2004.

325-331.

[44] Liu B, Wang L, Jin Y H, et al. Improved particle swarm optimization combined with chaos[J]. Chaos, Solitons and Fractals. 2005, 25(5): 1261-1271.

[45] Angeline P J, Inc N S, Vestal N Y. Using selection to improve particle swarm optimization[C]. In: The 1998 IEEE International Conference on Evolutionary Computation Proceedings.Anchorage, AK, USA: 1998. 84-89.

[46] Eberhart R, Shi Y. Particle Swarm Optimization: Developments, Applications and Resources[C]. Piscataway, NJ, USA: IEEE, 2001: 81-86.

[47] Clerc M. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization[C]. In: Proceedings of the 1999 Congress on Evolutionary Computation, CEC 99.Washington, DC, USA: 1999. 1951-1957.

[48] Van den Bergh F, engelbrecht A P. A Cooperative approach to particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 225-239.

[49] 高海兵, 周驰, 高亮. 广义粒子群优化模型[J]. 计算机学报, 2005, 28(12): 1980-1988.

[50] 王华秋, 曹长修. 基于模拟退火的并行粒子群优化研究[J]. 控制与决策, 2005, 20(5): 500-505.

[51] Zhang W J, Xie X F. DEPSO: hybrid particle swarm with differential evolution operator[C]. In: IEEE International Conference on Systems, Man and Cybernetics.2003:3816-3821.

[52] Lovbjerg M, Rasmussen T K, Krink T. Hybrid particle swarm optimiser with breeding and subpopulations[C]. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001).San Francisco, USA: San Francisco, USA, July, 2001:469-476.

[53] Kennedy J, Eberhart R. A Discrete Binary Version of the Particle Swarm Algorithm[C]. In: Proceeding of the World Multiconference on Systemics, Cybernetics and Informatics. N J :Piscataway: 1997: 4104-4109.

[54] Salman A, Ahmad I, Al-madani S. Particle swarm optimization for task assignment problem[J]. Microprocessors and Microsystems, 2002, 26(8): 363-371.

[55] Lian Z, Gu X, Jiao B. A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan[J]. Chaos, Solitons and Fractals, 2008, 35(5): 851-861.

[56] Liao C J, Tseng C T, Luarn P. A discrete version of particle swarm optimization for flowshop scheduling problems[J]. Computers and Operations Research, 2007, 34(10):

3099-3111.

[57] Correa E S, Freitas A A, Johnson C G. Particle swarm for attribute selection in Bayesian classification: an application to protein function prediction[J]. Journal of Artificial Evolution and Applications, 2008, 8(2): 1-12.

[58] Shen Q, Shi W M, Kong W, et al. A combination of modified particle swarm optimization algorithm and support vector machine for gene selection and tumor classification[J]. Talanta, 2007, 71(4): 1679-1683.

[59] Yin P Y. A discrete particle swarm algorithm for optimal polygonal approximation of digital curves [J]. Journal of Visual Communication and Image Representation, 2004, 15(2): 241-260.

[60] Sudholt D, Witt C. Runtime analysis of binary PSO[C]. In: Proceedings of the 10th annual conference on Genetic and evolutionary computation.ACM New York, NY, USA, 2008. 135-142.

[61] Clerc M. Discrete particle swarm optimization illustrated by the traveling salesman problem[M]. New optimization techniques in engineering, 2004: 219–239.

[62] 钟一文, 宁正元, 蔡荣英等. 一种改进的离散粒子群优化算法[J]. 小型微型计算机系统, 2006, 27(010): 1893-1896.

[63] 高尚，韩斌，吴小俊. 求解旅行商问题的混合粒子群优化算法[J]. 控制与决策, 2004, 19(11): 1286-1289.

[64] Tasgetiren M F, Sevkli M, Liang Y C, et. al. Particle Swarm Optimization Algorithm for Permutation Flow Shop Sequencing Problem[J]. Lecture Notes in Computer Science. 2004, 31(72): 382-390.

[65] Tasgetiren M F, Sevkli M, Liang Y C, et.al. Particle Swarm Optimization on Algorithm for single Machine total Weighted Tardiness Problem[C]. In: Proceeding of the 2004 Congress on Evolutionary Computiona. Portland , Oregon: 2004.:1412-14919.

[66] Riccardo Poli. An Analysis of Publications on Particle Swarm Optimisation Applications[M]. Technical Report CSM-469, Department of Computer Science ,University fo Essex, 2007.

[67] Riccardo Poli, James Kennedy, Tim Blackwell. Particle swarm optimization, An overview[J]. Swarm Intelligence. 2007, 1: 33-57.

[68] Alec Banks, jonathan Vincent, Chukwudi Anyakoha. A review of Particle swarm optimization. Part I: background and development[J]. Natural Computing, 2007, 6: 467-484.

[69] Alec Banks, jonathan Vincent, Chukwudi Anyakoha. A review of Particle

swarm optimization. Part II: hybridisation,combinatorial,multicriteria and constrained optimization, and indicative applications[J]. Natural Computing, 2008, 7: 109-124.

[70] F Van den Bergh, AP Engelbrecht. Cooperative learning in neural networks using particle swarm optimizers[J]. South African Computer Journal, 2000, 26: 84-90.

[71] 傅强, 胡上序, 赵胜颖．基于PSO 算法的神经网络集成构造方法[J]. 浙江大学学报(工学版)， 2004, 38(12): 1596-1600.

[72] Tiagl Sousa, Arlindo Silva, Ana Neves. Particle Swarm based Data Mining Algorithms for classification tasks[J]. Parallel Computing, 2004, 30(5-6): 767-783.

[73] 丁华, 王秀坤, 孙焘．基于PSO改进决策树算法的研究[J]. 小型微型计算机系统，2005， 26(7): 1026-1021.

[74] Christopher K M, Kevin D,seppij. Bayesian Optimization Models for Particle Swarms[C]. In: Proceedings of the 2005 conference on Genetic and evolutionary computation, New York, USA: ACM, 2005. 193-200.

[75] Tao Du, S S Zhang, Zongjiang Wang. Efficient Learning Bayesian Networks Using PSO[J]. Lecture notes in computer science, 2005: 151–156.

[76] Dw Van Der Merwe, Ap Engelbrecht. Data Clustering using Particle Swarm Optimization[C]. In: The 2003 Congress on Evolutionary Computation, CEC'03.2003. 215-220.

[77] Walter Cedeno , Dimitris K Agrafiotis. Using particle swarms for the development of QSAR models based on K-nearest neighbor and kernel regression[J]. Journal of Computer-Aided Molecular Design, 2003, 17: 255–263,.

[78] 孙波, 陈卫东, 席裕庚. 基于粒子群优化算法的移动机器人全局路径规划[J]. 控制与决策，2005， 20(9): 1052-1056.

[79] Bill C.h. Chang, Asanga Ratnawera S K H. Particle Swarm Optimisation for Protein Motif Discovery[J]. Genetic Programming and Evolvable Machines, 2004(5): 203–214.

[80] View T O. An approach to multimodal biomedical image registration utilizing particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation,. 2004, 8(3): 289-301.

[81] Xu R, Anagnostopoulos G C, Wunsch D C. Multiclass Cancer Classification Using Semisupervised Ellipsoid ARTMAP and Particle Swarm Optimization with Gene Expression Data[J]. IEEE ACM Transactions on Computational Bioinformatics. 2007, 4(1): 65-77.

[82] Horst, Pardalos, Thoai, et al. 全局优化引论[M]. 清华大学出版社, 2003.

120-207.

[83] Fonseca C M, Fleming P J. An Overview of Evolutionary Algorithms in Multiobjective Optimization[J]. Evolutionary Computation. 1995, 3(1): 1-16.

[84] Back T, Fogel D B, Michalewicz Z. Handbook of Evolutionary Computation [M]. IOP Publishing Ltd. Bristol, UK, 1997.

[85] WM Spears, KA De Jong, T Baeck, et al. An overview of evolutionary computation[C]. proceedings of the 1993 European Conference on Machine Learning, 1993: 442-442.

[86] Holland J H. Adaptation in Natural and Artficial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence[M]. Cambridge: MIT: 1992. 1-200.

[87] J R Koza. Genetic Programming : On the Programming of Computers by Means of Natural Selection[M]. MIT press, 1992.

[88] Koza J R, Rice J P. Genetic programming[M]. Springer, 1992.

[89] Randy L. Haupt, Sue Ellen Haupt. Practical Genetic Algorithms [M]. Pitman Publishing, 1999.

[90] L J Forgel. On the Organization of Intellect[M]. Los Angeles: The University of California Press, 1964.

[91] Forgel D B．An introduction to simulated evolutionary optimization[J]．IEEE Transaction on Neural Network, 1994,5(1): 3-4.

[92] Kennedy J，Eberhart RC，Shi Y.Swarm intelligence[M]．Morgan Kaufmann Publishers，2001

[93] Kennedy J F，Eberhart R，Shi Y,et al．Swarm intelligence[M]．Springer，2001．

[94] 彭喜元，彭宇，戴海丰．群智能理论及其应用[J]．电子学报．2003，31(12A)：1982-1988

[95] Dorigo M, Gambardella L M. Ant colonies for the travelling salesman problem[J]. BioSystems. 1997, 43(2): 73-81.

[96] Endoh S, Toma N, Yamada K. Immune algorithm for n-TSP[C]. In: IEEE International Conference on Systems, Man, and Cybernetics, 1998.San Diego, CA, USA: 1998:3844-3849.

[97] Martin T.h, Howard B.D, Mark H.b. Neural Network Design[M]. Beijin: China Machine Press, 2002.

[98] Reynolds R G. An introduction to cultural algorithms[C]. In: Proceedings of the Third Annual Conference on Evolutionary Programming.1994. 131–139.

[99] Paun G, Rozenberg G, Salomaa A. DNA Computing: New Computing Paradigms[M]. Springer, 1998.

[100] Kennedy J. The particle swarm: social adaptation of knowledge[C]. In: IEEE International Conference on Evolutionary Computation.Indianapolis, IN, USA: 1997. 303-308.

[101] Shi Y, Eberhart R. A Modified Particle Swarm Optimizer[C]. In: IEEE World Congress on Computational Intelligence.Anchorage, AK, USA: 1998. 69-73.

[102] Shi Y, Eberhart R. Empirical Study of Particle Swarm Optimization[C]. In: Piscataway, NJ: IEEE Service Center, 1999. 1948-1950.

[103] Li-ping Z, Huan-jun Y, Shang-xu H. Optimal choice of parameters for particle swarm optimization[J]. Journal of Zhejiang University-Science A. 2005, 6(6): 528-534.

[104] Elbeltagi E, Hegazy T G D. Comparision among five elolutionary-based optimization algorithms[J]. Advanced Engineering Informatics. 2005(19): 43-54.

[105] 雷开友. 粒子群算法及其应用研究[D]，[博士学位论文]. 西南大学, 2006.

[106] Hu X, Shi Y, Eberhart R. Recent advances in particle swarm[C]. In: Congress on Evolutionary Computation (CEC2004).2004. 19-23.

[107] Riccardo Poli, James Kennedy, Tim Blackwell. Particle Swarm Optimization -An overview[J]. Swarm Intelligence. 2007, 1: 33-57.

[108] Poli R. Analysis of the publication on the application of Particle swarm optimizaton[J]. Journal of Artificial Evolution and application. 2007, Technical Report May 2007: 1744-8050.

[109] Aler Banks, J.vincent, C.anyakoha. A review of Particle swarm optimization[J]. Natural Computing. 2008, 7(3): 109-124.

[110] Ozcan E, Mhoan C. Analysis of a Simple Particle Swarm Optimization System[M]. Intelligent Engineering Systems Through Artifical Neural Networks, 1998: 253-258.

[111] R.brits, A.p.engelhrecht, Fan Ven Den Bergh. Locating multiple optimal using partilce swarm optimization[J]．Applied Mathematics and Computation, 2007, 189: 1859-1883.

[112] Jaco F. S, Albert A.g. A Study of Global Optimization Using Particle Swarms[J]. Journal of Global Optimization,2005, 31:93-108.

[113] Van B F, Engelbrecht A P．A study of particle swarm optimization particle trajectories[J]. Information Sciences. 2006, 176(8): 937-971.

[114] Shi Y, Eberhart C．Fuzzy Adaptive Particle Swarm Optimization[C]．In:

Proceedings of the IEEE Conference on Evolutionary Computation.Koreal ,Soul: IEEE press, 2001:101-106.

[115] 李艳, 樊晓平．基于遗传算法的城市单交叉路口信号动态控制[J]．交通运输系统工程与信息，2002, 2(1): 49-53.

[116] 黄辉先, 史忠科．城市单交叉路口交通流实时遗传算法优化控制[J]．系统工程理论与实践，2001, 21(3): 102-106.

[117] Rudolph G．Convergence analysis of canonical genetic algorithm[J]．IEEE Transications on Neural Networks, 1994, 5: 96-101.

[118] Wolpert D H, Macready W G．No free lunch theorems for search[J]. IEEE Transactions on Evolutionary Computation，1997, 1(1): 67-82.

[119] Bin W, Yi Z, Shaohui L, et al．CSIM: a document clustering algorithm based on swarm intelligence[C]．In: Proceedings of the 2002 Congress on Evolutionary Computation, CEC'02.2002: 477-482.

[120] Aickelin U, Greensmith J, Twycross J．Immune system approaches to intrusion detection-a review[J]. Lecture notes in computer science，2004: 316-329

.

攻读学位期间发表的论文及参与的课题

公开发表的论文:

1. 刘建华, 樊晓平, 瞿志华. 一种惯性权重动态调整的新型粒子群算法[J]. 计算机工程与应用. 2007, 43(7): 68-70.

2. 刘建华, 樊晓平, 瞿志华. 一种基于相似度的新型粒子群算法[J]. 控制与决策. 2007, 22(10): 1155-1159. (EI: 074710936675)

3. Liu Jianhua, Fan Xiaoping, Qu Zhihua. An Improved Particle Swarm Optimization with Mutation Based on Similarity[C]. In: Third International Conference on Natural Computation, ICNC 2007.Haikou, China, 2007.9: 824-828. (EI: 080311026983, ISTP)

4. 刘建华, 刘建伟. 基于粒子群算法的城市单交叉口信号控制[J]. 系统工程. 2007, 25(7): 83-87.

5. Jianhua Liu, Xiaoping Fan, Zhihua Qu. The Progressive Analysis of Particle Swarm Optimization［J］. Journal of computational Information Systems. 2008.4(6):1829-1836.

6. Jianhua Liu, Xiaoping Fan, Zhihua Qu. A New Interestingness Measure of Association Rule[C]. Second International Conference on Genetic and Evolutionary Computing. JingZou, Hubei, China. 2008.9.25-26:393-397. (EI, ISTP)

参与的课题：

1.半督聚类算法及其应用研究，福建省自然科学基金，项目编号：2008J04004;

2.特征语义信息网络视频图像检索，福建省教育厅教育厅 A 类，项目编号：JA05213

致谢

首先，感谢导师樊晓平教授。几年的博士学习，离不开导师的热心支持与悉心关怀。论文的成文定稿，离不开导师的精心指导和辛勤教育。樊老师严谨治学的态度及为学生着想的为人，使我几年的博士学业得以顺利完成。在此，向樊老师致以崇高的敬意，并表示深深的谢意。

感谢瘳志芳老师为我博士学习提供的帮助。感谢多年来一起在实验室学习工作的刘少强博士、黄琛喜博士、黎燕博士、李勇周博士以及其他所有的师兄姐弟妹们，是他们的相伴让我在实验室度过一段段值得一生回忆的时光，感谢一同学习的陈冰梅博士、张剑博士一路提供的相助。感谢我的同事郭躬德夫妇、姚志强夫妇给予的帮助，感谢我的单位领导给予的支持。

感谢我的妻子杨发萍默默的支持与无私的奉献，谢谢她为我的学业作出的付出。感谢岳父岳母及我家人为我提供的支持与帮助，希望他们健康长寿。感谢儿子刘阳锐给我带来的快乐与精神动力，希望他健康成长。

最后感谢所有给予我诚恳意见的专家和学者们，敬请批评指正。并借此向各位致以诚挚的问候。

刘建华

2009 年 3 月