南京理工大学

博士学位论文

蚁群算法理论、应用及其与其它算法的混合

姓名：高尚

申请学位级别：博士

专业：模式识别与智能系统

指导教师：杨静宇

20051220

摘 要

意大利学者 Dorigo M.，Maniezzo V.和 Colorni A.于 1992 年通过模拟蚁群觅食行为提出了一种基于种群的模拟进化算法—蚁群优化(ACO)。该算法的出现引起了学者们的极大关注，在过去短短十多年的时间里，已在组合优化、网络路由、函数优化、数据挖掘、机器人路径规划等领域获得了广泛的应用，并取得了较好的效果。

本论文围绕蚁群算法的理论及其应用，就如何解决非线性整数规划问题、连续性优化问题、聚类问题、与其它算法混合以及收敛性进行了较为深入、系统的研究。本文的主要研究成果包括：

（1）提出了一种新的蚁群算法来求解无约束的整数规划问题。蚂蚁在整数空间内运动，同时在路径上留下激素，以此引导搜索方向。对蚁群算法参数的合理选取进行了实验分析，给出了算法参数选取的基本原则。数值试验计算结果表明该方法比较有效，并具有通用性。对几个典型的整数规划问题，如武器-目标分配、多处理机调度、可靠性优化等问题，根据各问题的特点，采用了不完全相同方法，并与其他方法作了比较，效果比较有效。提出了一种求解旅行商问题的多样信息素的蚁群算法。根据蚁群算法信息素更新的特性，把蚁群的三种不同的信息素更新方式混合在一起，既利用了局部信息，又考虑了整体信息，将局部搜索和全局搜索相结合，使收敛性得到提高。旅行商问题的仿真实验结果表明了该混合算法的有效性。提出一种解决连续优化问题的蚁群算法。把连续解空间分成若干空间网格点，采用蚁群算法找出信息量大的空间网格点，缩小变量范围，继续找出信息量大的空间网格点，直到网格的间距小于预先给定的精度。

（2）对模式识别中典型问题-聚类问题进行了研究，提出了两种求解聚类问题的蚁群算法。一种方法是模拟蚂蚁寻食过程，蚂蚁在模式样本到聚类中心的路径上留下外激素，外激素引导聚类过程；另一种是与 K-均值算法混合，利用 K-均值方法的结果作为初值，根据分类结果更新信息素。测试数据显示与 K-均值算法混合的算法效果相当好。

（3）提出了与模拟退火算法混合的两种算法。一种是在模拟退火算法中运用蚁群算法思想找邻域的解，称为蚁群模拟退火算法，并用该算法解决圆排列问题；另一种是由模拟退火算法生成初始信息素分布，然后由蚁群算法根据累计更新的信息素找出若干组解后，再经过模拟退火算法在邻域内找另外一个解的操作，得到更有效的解，称为模拟退火蚁群算法，并用该算法解决旅行商问题。

（4）根据蚁群算法与遗传算法的特性，提出了与遗传算法混合的蚁群算法。由遗传算法生成初始信息素分布，在蚂蚁算法寻优中，采用遗传算法的交叉和变异的策略，得到更有效的解。

（5）利用混沌运动的遍历性、随机性和规律性等特点，提出了一种求解旅行商问题的混沌蚁群(CACO)算法。该算法的思想是采用混沌初始化进行改善个体质量和利用混沌扰动避免搜索过程陷入局部极值。与模拟退火算法、标准遗传算法进行比较，仿真结果表明该方法是一种简单有效的算法。

（6）对群智能算法中另一种算法-粒子群算法进行了研究，提出了模拟退火思想的粒子群算法来解连续性优化问题，经过与基本粒子群算法比较测试，证实它是一种简单有效的算法。提出了混沌粒子群优化算法连续性优化问题，典型复杂函数优化仿真结果表明该方法是一种有效的算法。提出了一种结合遗传算法的思想的混合粒子群算法来解决背包问题，对于目前还没有好的解法的组合优化问题，很容易地修改此算法就可解决。根据蚁群算法与粒子群优化(PSO)算法的特性，提出了两种混合算法。一种是结合遗传算法、蚁群算法和模拟退火算法的思想提出混合粒子群算法。用该算法求解著名的旅行商问题，被证实是一种比较有效的方法。另一种是粒子群-蚁群算法，首先随机产生若干组比较好的解生成信息素分布，然后由蚁群算法根据累计更新的信息素找出若干组解后，再由粒子群算法进行交叉、变异操作，得到更有效的解。与模拟退火算法、标准遗传算法和标准蚁群算法进行比较，混合算法效果很好。

(7)对求解最短路问题的蚁群算法的几个收敛特性进行了证明，提出的定理给出了寻找最短路的蚁群算法收敛的充分条件，并通过一个数值例子验证了该结果。

最后，对全文的研究工作进行了总结，并展望了蚁群算法进一步还要研究的课题。

关键词：蚁群算法，整数规划，聚类问题，K-均值算法，模拟退火算法，遗传算法，混沌，粒子群优化算法，旅行商问题

Abstract

A population-based simulated evolutionary algorithm called ant colony optimization (ACO for short) was proposed in 1992 by Italian researchers Dorigo M., Maniezzo V. and Colorni A. Many scholars are attracted to study ACO and in the past ten years the algorithm has been widely applied to the fields of combinatorial optimization, network routing, functional optimization, data mining, and path planning of robot etc, and good effects of application are gained.

The dissertation focuses on the principles, theory, and applications of ACO, especially, an in-deep and systemic study on how to improve the basic ACO algorithm, solving nonlinear integer programming, continuous optimization problem, clustering problem, hybridizing other algorithms and convergence. The main achievements of this dissertation include:

1. A new ACO algorithm for unconstrained nonlinear integer optimization problem is present. Ants move around the set of integers space, and while walking the ants lays down pheromone on the ground. The pheromone is used to direct the search process. Experimental analyses are carried out on the reasonable selection on the parameters of this algorithm, and basic principles for the parameter selection are provided. Results of numerical tests show the effectiveness and generality of the method. By use of the properties of weapon-target assignment problem, multiprocessor scheduling problem and reliability optimization problem, incomplete same methods are presented to solve them. Comparing with other methods, and their effectiveness are illustrated through result. An ant colony algorithm based on multiplicate pheromone is proposed to solve the traveling salesman problems. By use of the properties of pheromone of ant colony algorithm, three modes of updating the pheromone are hybridized. The method uses not only local information but also global information and combines the local search with the global search to improve its convergence. The simulation results for TSP show the validity of this algorithm. An ACO algorithm for solving continuous optimization problem is presented. By dividing the space of solution into many grids, ants first find the grid in which the trail information is most. Ants squeeze the intervals and repeat the process until the interval is as small as desired.

2. The clustering problem is a typical problem in pattern recognition. Two kinds of clustering methods based on ant colony algorithm are proposed. One method is to act how ants look for food. While walking the ants lays down pheromone on the ground, which is

from the sample to center. The pheromone is used to direct the search process. The other method is a hybrid algorithm combining ant colony optimization algorithm with K-means. The algorithm is then extended to use K-means clustering to seed the initial solution and the information pheromone is adjusted according to them. The hybrid ant colony algorithms are proved effective by testing.

3.Two hybrid algorithms combining ACO algorithm with simulated annealing algorithm optimization are presented. One is called ant colony-simulated annealing algorithm. Combine the ideal of the ant colony algorithm, the simulated annealing algorithm search the neighborhood solution. Using ant colony-simulated annealing method, the circle permutation problem is solved. The other method is called simulated annealing-ant colony algorithm. First, it adopts simulated annealing algorithm to give information pheromone to distribute. Second, it makes use of the ant colony algorithm to get several solutions through information pheromone accumulation and renewal. Finally, by searching a solution of neighborhood of simulated annealing algorithm, the effective solutions are obtained. The traveling salesman problems are solved by it.

4. By use of the properties of ant colony algorithm and genetic algorithm, a hybrid algorithm is proposed to solve the traveling salesman problems. First, it adopts genetic algorithm to give information pheromone to distribute. Second, it makes use of the ant colony algorithm to get several solutions through information pheromone accumulation and renewal. Finally, by using across and mutation operation of genetic algorithm, the effective solutions are obtained.

5.By use of the properties of ergodicity, randomicity, and regularity of chaos, a chaos ant colony optimization (CACO) algorithm is proposed to solve traveling salesman problem. The basic principle of CACO algorithm is that chaos initialization is adopted to improve individual quality and Chaos perturbation is utilized to avoid the search being trapped in local optimum. Compared with the standard GA and simulated annealing algorithm , simulation results show that chaos ant colony optimization is a simple and effective algorithm.

6.The particle swarm optimization (PSO) algorithm that is another swarm intelligence method is researched in this paper. The particle swarm optimization algorithm combine the ideal of the simulated annealing algorithm are recommended. Compared with the standard PSO, it is proved that a new method is a simple and effective algorithm. A chaos particle swarm optimization (CPSO) algorithm is proposed to solve the optimization problems. Simulation results of typical complex function optimization show that chaos particle

swarm optimization is an effective algorithm. The particle swarm optimization algorithm combine the ideal of the genetic algorithm is recommended to solve knapsack problem. It can easily be modified for any combinatorial problem for which we have no good specialized algorithm. By use of the properties of ant colony algorithm and particle swarm optimization algorithm, two hybrid algorithms are proposed. One is a hybrid particle swarm optimization combine the ideal of the genetic algorithm, ant colony algorithm and the simulated annealing algorithm. The hybrid particle swarm optimization algorithms proved to be highly effective for solving to solve the traveling salesman problems. The other method is called particle swarm-ant colony algorithm. First, it adopts statistics method to get several initial better solutions and in accordance with them, gives information pheromone to distribute. Second, it makes use of the ant colony algorithm to get several solutions through information pheromone accumulation and renewal. Finally, by using across and mutation operation of particle swarm optimization algorithm, the effective solutions are obtained. Compare with the simulated annealing algorithm, the standard genetic algorithm and the standard ant colony algorithm, the hybrid algorithms are proved effective.

7. Some convergence properties of ant colony algorithm for solving shortest path problem are proved. The theorems establish sufficient condition for the convergence ant colony algorithm for solving shortest path problem .A numerical example is given to illustrate the efficiency of the results.

Finally, the work of this dissertation is summarized and the prospective of future research is discussed.

Keywords: Ant Colony Algorithm, Integer programming, Clustering problem, K-Means algorithm, Simulated annealing algorithm, Genetic algorithm, Chaos, Particle swarm optimization algorithm , Traveling salesman problem

声　明

本学位论文是我在导师的指导下取得的研究成果，尽我所知，在本学位论文中，除了加以标注和致谢的部分外，不包含其他人已经发表或公布过的研究成果，也不包含我为获得任何教育机构的学位或学历而使用过的材料。与我一同工作的同事对本学位论文做出的贡献均已在论文中作了明确的说明。

研究生签名：高尚　　05年12月20日

学位论文使用授权声明

南京理工大学有权保存本学位论文的电子和纸质文档，可以借阅或上网公布本学位论文的全部或部分内容，可以向有关部门或机构送交并授权其保存、借阅或上网公布本学位论文的全部或部分内容。对于保密论文，按保密的有关规定和程序处理。

研究生签名：高尚　　05年12月20日

1 绪论

1.1 引言

人工智能在经历了上个世纪80年代整整10年的繁荣后，由于方法论上始终没有突破经典计算思想的藩篱，再次面临着寒冬季节的考验。人工智能的研究前景又一次变得暗淡无光。与此同时，随着人们对生命本质的不断了解，生命科学却以前所未有的速度迅猛发展，使人工智能的研究开始摆脱经典逻辑计算的束缚，大胆探索起新的非经典计算途径。正如人工智能先驱Minsky所认为的“我们应该从生物学而不是物理学受到启示……”那样，对生物启发式计算的研究，成为人工智能迎接新曙光而开启的又一个春天。在这种背景下，社会性动物（如蚁群、蜂群、鸟群等）的自组织行为引起了人们的广泛关注，许多学者对这种行为进行数学建模并用计算机对其进行仿真，这就产生了所谓的“群集智能”(Swarm Intelligence,简称SI)。社会性动物的妙处在于：个体的行为都很简单，但当它们一起协同工作时，却能够“突现”出非常复杂（智能）的行为特征。例如，单只蚂蚁的能力极其有限，但当这些简单的蚂蚁组成蚁群时，却能完成像筑巢、觅食、迁徙、清扫蚁巢等复杂行为；一群行为显得盲目的蜂群能造出精美的蜂窝；鸟群在没有集中控制的情况下能够同步飞行等。在这些自组织行为中，又以蚁群在觅食过程中总能找到一条从蚁巢到食物源的最短路径最为引人注目。

上世纪50年代中期创立了仿生学，人们从生物进化的机理中受到启发，提出了许多用以解决复杂优化问题的新方法，如遗传算法、进化规划、进化策略等，蚁群优化(ACO: ant colony optimization)算法是最近几年才提出的一种新型的模拟进化算法，由意大利学者Colorni A、Dorigo M和Maniezzo V等人于1992年首先提出来[1]，用蚁群在搜索食物源的过程中所体现出来的寻优能力来解决一些离散系统优化中困难问题。已经用该方法求解了旅行商问题、指派问题、调度问题等，取得了一系列较好的实验结果[2,3,4]。蚁群算法是一种新型的模拟进化算法，鉴于目前国内尚缺乏这一方面的研究，其研究刚刚开始，远未像遗传算法，模拟退火等算法那样形成系统的分析方法和坚实的数学基础，有许多问题有待进一步研究，如算法的收敛性、理论依据等更多细致的工作还有待于进一步展开。本论文对蚁群算法的理论依据等作深入研究，并推广其应用面。

1.2 蚁群算法的基本原理

像蚂蚁这类群居昆虫，虽然没有视觉，却能找到由蚁穴到食物源的最短路径，原因是什么呢？虽然单个蚂蚁的行为极其简单，但由这样的单个简单的个体所组成的蚁群群体却表现出极其复杂的行为，能够完成复杂的任务，不仅如此，蚂蚁还能够适应环境的变化，如：在蚂蚁运动路线上突然出现障碍物时，蚂蚁能够很快重新找到最优

路径。蚂蚁是如何完成这些复杂任务的呢？人们经过大量的研究发现，蚂蚁个体间是通过一种称之为信息素（pheromone）的物质进行信息传递，从而能相互协作，完成复杂的任务。蚂蚁之所以表现出复杂有序的行为，个体之间的信息交流和相互协作起着重要的作用。

蚂蚁在运动过程中，能够在它所经过的路径上留下该物质，并以此指导自己的运动方向，蚂蚁倾向于朝着该物质强度高的方向移动。因此，由大量蚂蚁组成的蚁群的集体行为边表现出一种信息正反馈现象：某一路径上走过的蚂蚁越多，则后者选择该路径的概率越大。蚂蚁个体之间就是通过这种信息的交流达到搜索食物的目的。这里，用一个形象化的图示来说明蚂蚁群体的路径搜索原理和机制：

假定障碍物的周围有两条道路可从蚂蚁的巢穴到达食物源（如图 1.1）：Nest-ABD-Food 和 Nest-ACD-Food，分别具有长度 4 和 6。蚂蚁在单位时间内可移动一个单位长度的距离。开始时所有道路上都未留有任何信息素。

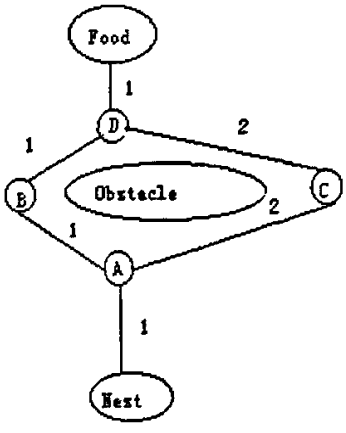

图 1.1 蚁群系统示意图

在 $t=0$ 时刻，20 只蚂蚁从巢穴出发移动到 A，它们以相同概率选择左侧或右侧道路，因此平均有 10 只蚂蚁走左侧，10 只走右侧；

在 $t=4$ 时刻，第一组到达食物源的蚂蚁将折回；

在 $t=5$ 时刻，两组蚂蚁将 D 点相遇。此时 BD 上的信息素数量和 CD 上的相同，因为各有 10 只蚂蚁选择了相应的道路，从而有 5 只返回的蚂蚁将选择 BD 而另 5 只将选择 CD；

在 $t=8$ 时刻，前 5 只蚂蚁将返回巢穴，而 AC、CD、BD 上各有 5 只蚂蚁；

在 $t=9$ 时刻，前 5 只蚂蚁又回到 A 并且再次面对往左还是往右的选择；

这时，AB 上的轨迹数是 20 而 AC 上是 15，因此将有较为多数的蚂蚁选择往左，

从而增强了该路线的信息素。随着该过程的继续，两条道路上的信息素数量的差距将越来越大，直至绝大多数蚂蚁都选择了最短的路线。正是由于一条道路要比另一条道路短，因此，在相同的时间区间内，短的路线会有更多的机会被选择。

蚁群算法是一种随机搜索算法，与其他模型进化算法一样，通过侯选解组成的群体的进化过程来寻求最优解，该过程包含两个阶段：适应阶段和协作阶段。在适应阶段，各侯选解根据积累的信息不断调整自身结构；在协作阶段，侯选解之间通过信息交流，以期望产生性能更好的解。

作为与遗传算法同属一类的通用型随机优化方法，蚁群算法不需要任何先验知识，最初只是随机地选择搜索路径，随着对解空间的“了解”，搜索变得有规律，并逐渐逼近直至最终达到全局最优解。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。一只蚂蚁搜索过的路径在下次搜索时就不会再被选择，由此在蚁群算法中建立禁忌列表来进行模拟；

(2)蚂蚁利用信息素(pheromone)进行相互通信。蚂蚁在所选择的路径上会释放一种叫做信息素的物质，当同伴进行路径选择时，会根据路径上的信息素进行选择，这样信息素就成为蚂蚁之间进行通讯的媒介。

(3)蚂蚁的集群活动。通过一只蚂蚁的运动很难到达食物源，但整个蚁群进行搜索就完全不同。当某些路径上通过的蚂蚁越来越多时，在路径上留下的信息素数量也越来越多，导致信息素强度增大，蚂蚁选择该路径的概率随之增加，从而进一步增加该路径的信息素强度，而某些路径上通过的蚂蚁较少时，路径上的信息素就会随时间的推移而蒸发。因此，模拟这种现象即可利用群体智能建立路径选择机制，使蚁群算法的搜索向最优解推进。蚁群算法所利用的搜索机制呈现出一种自催化或正反馈的特征，因此，可将蚁群算法模型理解成增强型学习系统。

蚁群算法首先成功应用于旅行商问题，下面简单介绍其基本算法。

设有 m 个蚂蚁，每个简单蚂蚁有以下特征：它根据以城市距离和连接边上外激素的数量为变量的概率函数选择下一个城（设 $\tau_{ij}(t)$ 为 t 时刻边 $e(i,j)$ 上外激素的强度）。规定蚂蚁走合法路线，除非周游完成，不允许转到已访问城市，有禁忌表控制（设 $tabu_k$ 表示第 k 个蚂蚁的禁忌表，$tabu_k(s)$ 表示禁忌表中第 s 个元素）。它完成周游后，蚂蚁在它每一条访问的边上留下外激素。

设 $B_i(t)\ (i=1,...,n)$ 是在 t 时刻城市 i 的蚂蚁数，设 $m=\sum_{i=1}^{n} b_i(t)$ 为全部蚂蚁数。每个简单蚂蚁有以下特征：

（1）它根据以城市距离和连接边上外激素的数量为变量的概率函数选择下一个

城（设$\tau_{ij}(t)$为t时刻边$e(i,j)$上外激素的强度）。

（2）规定蚂蚁走合法路线，除非周游完成，不允许转到已访问城市，有禁忌表控制（设$tabu_k$表示第k个蚂蚁的禁忌表，$tabu_k(s)$表示禁忌表中第s个元素）。

（3）它完成周游后，蚂蚁在它每一条访问的边上留下外激素。

初始时刻，各条路径上的信息量相等，设$\tau_{ij}(0)=C$（C为常数）。蚂蚁$k(k=1,2...,m)$在运动过程中，根据各条路径上信息量决定转移方向，$p_{ij}^k(t)$表示在t时刻蚂蚁k由位置i转移到位置j的概率，

$$p_{ij}^k=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)\cdot\eta_{ij}^{\beta}(t)}{\sum_{s\in allowed_k}\tau_{is}^{\alpha}(t)\cdot\eta_{is}^{\beta}(t)} & if\ j\in allowed_k\\ 0 & otherwise\end{cases} \tag{1.1}$$

其中，$allowed_k=\{0,1,\cdots,n-1\}-tabu_k$表示蚂蚁$k$下一步允许选择的城市，与实际蚁群不同，人工蚁群系统具有记忆功能，$tabu_k\ (k=1,2...,m)$用以记录蚂蚁k当前所走过的城市，集合$tabu_k$随着进化过程做动态调整。η_{ij}表示边弧(i,j)的能见度，用某种启发式算法算出，一般取$\eta_{ij}=1/d_{ij}$，d_{ij}表示城市i与城市j之间的距离。α表示轨迹的相对重要性，β表示能见度的相对重要性，ρ表示轨迹的持久性，$1-\rho$理解为轨迹衰减度。随着时间的推移，以前留下的信息逐渐消失，用参数$1-\rho$表示信息消逝程度，经过n个时刻，蚂蚁完成一次循环，各路径上信息量要根据以下式做调整：

$$\tau_{ij}(t+n)=\rho\cdot\tau_{ij}(t)+\Delta\tau_{ij} \tag{1.2}$$

$$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^k \tag{1.3}$$

$\Delta\tau_{ij}^k$表示第k只蚂蚁在本息循环中留在路径ij上的信息量，$\Delta\tau_{ij}$表示本次循环中路径ij上的信息量增量，L_k表示第k只蚂蚁环游一周的路径长度，Q为常数。

$$\Delta\tau_{ij}^k=\begin{cases}\dfrac{Q}{L_K} & \text{若第}k\text{只蚂蚁在本次循环经过}ij\\ 0 & \text{否则}\end{cases} \tag{1.4}$$

$\tau_{ij}(t),\Delta\tau_{ij}(t),p_{ij}^{k}(t)$ 的表达形式可以不同，要根据具体问题而定。Dorigo M 曾给出三种不同模型，分别称为 ant-cycle system、ant-quantity system、ant-density system，它们的差别在于表达式（1.4）的不同。在 ant-quantity system 模型中：

$$\Delta\tau_{ij}^{k}=\begin{cases}\dfrac{Q}{d_{ij}} & \text{若第}k\text{只蚂蚁在时刻}t\text{和}t+1\text{之间经过}ij\\ 0 & \text{否则}\end{cases} \tag{1.5}$$

在 ant-density system 模型中：

$$\Delta\tau_{ij}^{k}=\begin{cases}Q & \text{若第}k\text{只蚂蚁在时刻}t\text{和}t+1\text{之间经过}ij\\ 0 & \text{否则}\end{cases} \tag{1.6}$$

它们的区别在于：后两种模型中，利用的是局部信息，而前者利用的是整体信息，在求解 TSP 问题时，性能较好，因而通常采用它为基本模型。解旅行商问题的蚁群算法的基本步骤：

步骤 1　$nc\leftarrow 0$；（nc 为迭代步数或搜索次数）各 τ_{ij} 和 $\Delta\tau_{ij}$ 的初始化；将 m 个蚂蚁置于 n 个顶点上；

步骤 2　将各蚂蚁的初始出发点置于当前解集中；对每个蚂蚁 k（$k=1,2,\cdots,m$），按概率 p_{ij}^{k} 移至下一顶点 j；将顶点 j 置于当前解集；

步骤 3　计算各蚂蚁的路径长度 L_k（$k=1,2,\cdots,m$）；记录当前的最好解；

步骤 4　按更新方程修改轨迹强度；

步骤 5　对各边弧 (i,j)，置 $\Delta\tau_{ij}\leftarrow 0$，$nc\leftarrow nc+1$；

步骤 6　若 $nc<$预定的迭代次数且无退化行为（即找到的都是相同解）则转步骤 2；

步骤 7　输出目前最好解。

由算法复杂性理论分析可知，该算法复杂度为 $O(nc\cdot n^3)$。

1.3 蚁群算法理论研究现状

20 世纪 50 年代中期出现了仿生学，人们从生物进化机理中受到启发，提出许多用以解决复杂优化问题的新方法，如遗传算法、进化规划、进经策略等。这些以生物特性为基础演化算法的发展，及对生物群落行为的发现引导研究人员进一步开展了对生物社会性的研究，从而出现了基于群智能理论的蚁群算法[4](1992，意大利 Dorigo M，Maniezzo V，Colorni A 等)和粒子群算法(1995，James Kenney 和 Russell

Eberhart)。虽然二者的研究基础和基本思想是一致的，但是它们的产生和发展却是相对独立的。

90 年代 Dorigo M 最早提出了蚁群优化算法一蚂蚁系统(Ant system, AS)并将其应用于解决计算机算法学中经典的旅行商问题(TSP)。从蚂蚁系统开始，基本的蚁群算法得到了不断的发展和完善，并在 TSP 以及许多实际优化问题求解中进一步得到了验证。这些 AS 改进版本的一个共同点就是增强了蚂蚁搜索过程中对最优解的探索能力，它们之间的差异仅在于搜索控制策略方面。而且，取得了最佳结果的 ACO 是通过引入局部搜索算法实现的，这实际上是一些结合了标准局域搜索算法的混合型概率搜索算法，有利于提高蚁群各级系统在优化问题中的求解质量。

最初提出的 AS 有三种版本:Ant density、Ant quantity 和 Ant cycle[5]。在 Ant density 和 Ant quantity 中蚂蚁在两个位置节点间每移动一次后即更新信息素，而在 Ant cycle 中当所有的蚂蚁都完成了自己的行程后才对信息素进行更新，而且每个蚂蚁所释放的信息素被表达为反映相应行程质量的函数。通过与其他各种通用的启发式算法相比，在不大于 75 城市的 TSP 中，这三种基本算法的求解能力还是比较理想的，但是当问题规模扩展时，AS 的解题能力大幅度下降，因此，其后的 ACO 研究工作主要都集中于 AS 性能的改进方面。较早的一种改进方法是精英策略(Elitist Strategy)[6]，其思想是在算法开始后即对所有已发现的最好路径给予额外的增强，并将随后与之对应的行程记为 Tgb(全局最优行程)，当进行信息素更新时，对这些行程予以加权，同时将经过这些行程的蚂蚁记为“精英”，从而增大较好行程的选择机会。这种改进型算法能够以更快的速度获得更好的解，但是若选择的精英过多则算法会由于较早的收敛于局部次优解而导致搜索的过早停滞。

为了进一步克服 AS 中暴露出的问题，文献[7]中提出了蚁群系统(Ant colony system, ACS)。该系统的提出是以该文作者较早提出的 Ant Q 算法[8]为基础的。Ant Q 将蚂蚁算法和一种增强型学习算法 Q learning 有机的结合了起来，ACS 与 AS 之间存在三方面的主要差异:首先，ACS 采用了更为大胆的行为选择规则；其次，只增强属于全局最优解的路径上的信息素，即:

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\rho\Delta\tau_{ij}^{gb}(t) \tag{1.7}$$

其中 $\Delta\tau_{ij}^{gb}(t)=1/L^{gb}$，$0<\rho<1$ 是信息素挥发参数，L^{gb} 是从寻路开始到当前为止全局最优路径长度。另外，还引入了负反馈机制，每当一只蚂蚁由一个节点移动到另一个节点时，该路径上信息素都按照式(1-5)被相应的消除一部分，从而实现一种信息素的局部调整以减小已选择过的路径再次被选择的概率。

$$\tau_{ij}=(1-\xi)\tau_{ij}+\xi\Delta\tau_0, \qquad 0<\xi<1 \tag{1.8}$$

在对 AS 进行直接完善的方法中，MAX-MIN Ant System 是一个典型代表[9]。该算

法修改了 AS 的信息素更新方式，每次迭代之后只有一只蚂蚁能够进行信息素的更新以获取更好的解。为了避免搜索停滞，路径上的信息素浓度被限制在$[\tau_{min},\tau_{max}]$范围内，另外，信息素的初始值被设为其取值上限，这样有助于增加算法初始阶段的搜索能力[9]。

另一种对 AS 改进的算法是 Rank based Version AS[10]与“精英策略”相似，在此算法中总是更新更好进程上的信息素，选择的标准是其行程长度$(L^1(t)\le L^2(t)\le\cdots\le L^m(t))$决定的排序，且每个蚂蚁放置信息素的强度通过式(1.6)中的排序加权处理确定，其中$\Delta\tau_{ij}^r=1/L^r$，$\Delta\tau_{ij}^{gb}(t)=1/L^{gb}$，$m$为每次迭代后放置信息素的蚂蚁总数。

$$\tau_{ij}(t+1)=(1-\rho)\tau_{ij}(t)+\sum_{r=1}^{m}(\varpi-r)\Delta\tau_{ij}^r+\varpi\Delta\tau_{ij}^{gb}(t) \tag{1.9}$$

这种算法求解 TSP 的能力与 AS 精英策略 AS 遗传算法和模拟退火算法进行了比较，在大型 TSP 问题中(最多包含 132 座城市)，基于 AS 的算法都显示出了优于 GA 和 SA 的特性，而且在 Rank based AS 和精英策略 AS 均优于基本 AS 的同时，前者还获得了比精英策略 AS 更好的解。

1.4 蚁群算法应用研究现状

随着群智能理论和应用算法研究的不断发展，研究者已尝试着将其用于各种工程优化问题，并取得了意想不到的收获。多种研究表明，群智能在离散求解空间和连续求解空间中均表现出良好的搜索效果，并在组合优化问题中表现突出。

蚁群优化算法并不是旅行商问题的最佳解决方法，但是它却为解决组合优化问题提供了新思路，并很快被应用到其他组合优化问题中，比较典型的应用研究包括:网络路由优化、数据挖掘以及一些经典的组合优化问题。蚁群算法在电信路由优化中已取得了一定的应用成果[11-14]。HP 公司和英国电信公司在 90 年代中后期都开展了这方面的研究，设计了蚁群路由算法(Ant colony routing，ACR)。每只蚂蚁就像蚁群优化算法中一样根据它在网络上的经验与性能，动态更新路由表项。如果一只蚂蚁因为经过了网络中堵塞的路由而导致了比较大的延迟，那么就对该表项做较大的增强，同时根据信息素挥发机制实现系统的信息更新，从而抛弃过期的路由信息，这样，在当前最优路由出现拥堵现象时，ACR 算法就能迅速的搜寻另一条可替代的最优路径，从而提高网络的均衡性、负荷量和利用率。目前这方面的应用研究仍在升温，因为通信网络的分布式信息结构、非稳定随机动态特性以及网络状态的异步演化与 ACO 的算法本质和特性非常相似。

基于群智能聚类算法起源于对蚁群蚁卵的分类研究，Lumer 和 Faieta 将 Deneubourg 提出将蚁巢分类模型应用于数据聚类分析[15]。其基本思想是将待聚类数

据随机地散布到一个二维平面内，然后将虚拟蚂蚁分布到这个空间内，并以随机方式移动，当一只蚂蚁遇到一个待聚类数据时即将之拾起并继续随机运动，若运动路径附近的数据与背负的数据相似性高于设置标准则将其放置在该位置，然后继续移动，重复上述数据搬运过程，按照这样的方法可实现对相似数据的聚类。

文献[16]中提出了一种利用蚁群算法设计的数据分类规则提取算法。利用蚂蚁的运动，根据数据属性的划分通过随机搜索逐步形成相对应规则的前件，在蚁群算法的搜索方法中定义了适合分类问题的规则构造方法、剪枝方法和信息素更新方法。该算法采用与 C4.5 相似的熵值度量方法，并将其与信息素更新结合起来进行以消除熵值度量的局限(局部启发性度量)引起的误差。与典型的分类算法 CN2 相比，这种方法的准确性与 CN2 相当，而且发现的规则列表比 CN2 获得规则列表简单。

另外，ACO 还在许多经典组合优化问题中获得了成功的应用，如二次规划问题(QAP)[17, 18]、机器人路径规划、作业流程规划、图着色(Graph coloring)等问题。可以说经过多年的发展，ACO 已成为能够有效解决实际二次规划问题的几种重要算法之一。AS 在作业流程计划(Job shops cheduling)问题中的首个应用实例出现在文献[19]中，该论文说明了 AS 在此领域的应用潜力。其后的文献[20]利用 MAX-MIN AS 解决 PAQ 取得了比较理想的效果，并通过实验中的计算数据证明采用该方法处理 PAQ 比较早的 SA 算法更好，且与禁忌搜索算法性能相当。文献[21]中利用 ACO 实现了对生产流程和特料管理的综合优化，并通过与遗传、模拟退火和禁忌搜索算法的比较证明了 ACO 的工程应用价值。

还有许多研究者将 ACO 用于了武器攻击目标分配和优化问题[22]、车辆运行路径规划[23]、区域性无线电频率自动分配[24]、Bayesian networks 的训练[25]和集合覆盖等应用优化问题[26]。Costa 和 Herz 还提出了一种 AS 在规划问题方面的扩展应用—图着色问题，并取得了可与其他启发式算法相比的效果。

蚁群算法理论的应用方法研究证明，虽然相对于各种比较成熟的计算智能方法来说蚁群算法的研究还处于初级阶段，并存在种种有待深入研究和解决的问题，但是可以预言群智能的研究代表了以后计算机研究发展的一个重要方向。

1.5 本文的主要工作

本课题研究了蚁群算法的理论，对其作了改进，将其应用在很多领域中，重点研究与其它智能算法的混合。主要工作和结论如下：

（1）首先采用蚁群算法解决一般性非线性整数规划问题，将整数规划变换成蚂蚁寻食过程。接着讨论了几个典型的整数规划问题，如武器-目标分配问题、多处理机调度问题、可靠性优化问题，根据各问题的特点，采用了不完全相同方法，并与其他方法作了比较，效果比较有效。提出了一种求解旅行商问题的多样信息素的蚁群算

法。根据蚁群算法信息素更新的特性，把蚁群的三种不同的信息素更新方式混合在一起，既利用了局部信息，又考虑了整体信息，将局部搜索和全局搜索相结合，使收敛性得到提高。旅行商问题的仿真实验结果表明了该混合算法的有效性。提出一种解决连续优化问题的蚁群算法。思路把连续空间进行离散化，分成若干空间网格点，采用蚁群算法找出信息量大的空间网格点，缩小变量范围，在此点附近进行人工蚁群移动，直到网格的间距小于预先给定的精度。

(2) 对模式识别中典型问题-聚类问题进行了研究，提出了两种求解聚类问题的蚁群算法。一种方法是把聚类问题变换成蚂蚁寻食过程，蚂蚁在模式样本到聚类中心的路径上留下外激素，引导其他蚂蚁选择；另一种是与K-均值算法混合，用K-均值算法作快速分类，根据分类结果更新信息素，指导其它蚂蚁选择。测试数据显示与K-均值算法混合的算法效果相当好。

(3) 提出了与模拟退火算法混合的两种算法，一种是在模拟退火算法中运用蚁群算法思想找邻域的解，称为蚁群模拟退火算法，并用该算法解决圆排列问题；另一种是采用模拟退火算法生成信息素分布，蚂蚁算法寻优中采用模拟退火的在邻域内找另外一个解的策略，称为模拟退火蚁群算法，并用该算法解决旅行商问题。

(4) 提出了与遗传算法混合的蚁群算法，由遗传算法生成初始信息素分布，在蚂蚁算法寻优中，采用遗传算法的交叉和变异的策略。以旅行商问题的典型的测试问题为测试对象，与其他算法进行了比较，该算法有其优越性。

(5) 提出了与混沌理论混合的蚁群算法，采用混沌初始化进行改善个体质量和利用混沌扰动避免搜索过程陷入局部极值。与其他算法进行了比较，该算法比较有效。

(6) 对群智能算法中另一种算法-粒子群算法进行了研究，提出了模拟退火思想的粒子群算法、混沌粒子群优化算法来解连续性优化问题；提出了混合其他算法的粒子群优化算法来求解背包问题。根据蚁群算法与粒子群优化算法的特性，提出了两种混合算法。一种是结合遗传算法、蚁群算法和模拟退火算法的思想提出混合粒子群算法。用该算法求解著名的旅行商问题，被证实是一种比较有效的方法。另一种是粒子群-蚁群算法，首先随机产生若干组比较好的解生成信息素分布，然后由蚁群算法根据累计更新的信息素找出若干组解后，再由粒子群算法进行交叉、变异操作，得到更有效的解。与模拟退火算法、标准遗传算法和标准蚁群算法进行比较，效率很高。

(7) 对求解最短路问题的蚁群算法的收敛性进行了探索性分析，提出的定理给出了寻找最短路的蚁群算法收敛的充分条件，并通过一个数值例子验证了该结果。

1.6 本文的内容安排

本文分 9 章。

第 1 章 先介绍了蚁群算法的基本原理，综述了蚁群算法理论研究现状和蚁群算

法应用研究现状，最后介绍了本文的主要工作。

第 2 章 先研究了求解一般非线性整数规划的蚁群算法，然后提出了求解武器-目标分配问题、多处理机调度问题、可靠性优化的蚁群算法，最后提出了多样信息素的蚁群算法。

第 3 章 提出了一种求解连续优化问题的简单的蚁群算法。

第 4 章 提出了求解聚类问题的基本蚁群算法和与 K-均值算法混合的蚁群算法。

第 5 章 提出了与模拟退火算法混合的蚁群模拟退火算法求解圆排列问题以及模拟退火蚁群算法求解旅行商问题。

第 6 章 首先介绍了基本遗传算法，然后提出了与遗传算法混合的遗传蚁群算法，最后与其他算法进行了测试比较。

第 7 章 首先介绍了混沌及运动特性，接着提出与混沌理论混合的混沌蚁群算法，最后与其他算法进行了测试比较。

第 8 章 先介绍了粒子群算法基本理论，然后提出模拟退火思想的粒子群算法、混沌粒子群优化算法来解连续性优化问题，提出了混合其他算法的粒子群优化来求解背包问题和其它组合优化问题的方法，最后提出了与粒子群优化算法两种混合算法，并用来求解旅行商问题。

第 9 章 针对最短路问题的蚁群算法的收敛性进行了探索性分析，提出的三个定理给出了寻找最短路的蚁群算法收敛的充分条件。

2 求解整数规划的蚁群算法

2.1 求解一般非线性整数规划的蚁群算法

2.1.1 引言

整数规划问题是运筹学中一门重要内容，其广泛应用于许多工程领域，如资源管理、生产调度、可靠性优化、目标分配、超大规模集成电路设计等。对于变量规模较小的整数规划，传统的求解方法有分支定界法[27]、割平面法[27]和隐枚举法[27]等。但对于较大规模的问题，传统的方法比较耗时，近年来随着进化计算的发展，许多学者运用遗传算法[28,29]、模拟退火算法[30]、粒子群优化算法[31]等方法来求解整数规划问题，文[32][33]采用混合蚁群算法来解整数规划问题，效果较好。本节提出一种新的蚁群算法来解一般非线性整数规划问题。特殊的问题可以采用特殊的方法，第二节、第三节和第四节将研究一些特殊的非线性整数规划，用蚁群算法来求解。

2.1.2 求解非线性整数规划的蚁群算法

无约束的整数规划问题其可描述为：

$$\begin{aligned} &\min f(x_1,x_2,\cdots,x_n) \\ &s.t.\ \ a_i \le x_i \le b_i \quad (i=1,2,\cdots,n) \\ &\quad x_i \in Z \quad (i=1,2,\cdots,n) \end{aligned} \tag{2.1.1}$$

其中 Z 为整数空间，$a_i,b_i(i=1,2,\cdots,n)$ 为整数，$l_i=b_i-a_i+1$ 为 x_i 的可能取的个数。可行解的空间如图 2.1.1 所示，x_i 有 l_i 个节点，每个变量取一个值就构成空间一个解。如 x_i 取第 m_i 个节点，则对应的解为

$$(x_1,x_2,\cdots,x_n)=(a_1+m_1-1,a_2+m_2-1,a_3+m_3-1,\cdots,a_n+m_n-1)$$

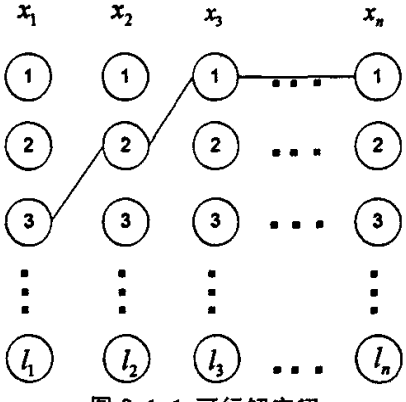

图 2.1.1 可行解空间

选取n个变量变成n级决策问题，第i级有l_i个节点，开始m个蚂蚁在第一级，第j级中选择第i个节点的概率p_{ij}为：

$$p_{ij}=\frac{\tau_{ij}}{\sum_{i=1}^{l_j}\tau_{ij}} \tag{2.1.2}$$

τ_{ij}理解为第j级中第i个节点的吸引强度。

更新方程为：

$$\tau_{ij}^{new}=\rho\tau_{ij}^{old}+\frac{Q}{f} \tag{2.1.3}$$

ρ表示强度的衰减系数，一般取0.5～0.9左右，Q为一正常数，f为目标函数值。

一条路径对应一个解，蚁群算法初始化时，各路径的信息素取相同值，让蚂蚁以等概率选择路径，这样使蚂蚁很难在短时间内从大量的杂乱无章的路径中，找出一条较好的路径，所以收敛速度较慢。假如初始化时就给出启发性的信息量，可以加快收敛速度。改进的方法是，产生大量的路径（如100条），从中选择比较优的（如30条），使这些路径留下信息素（与路径长度和成反比），各路径的信息量就不同，以此引导蚂蚁进行选择路径。

蚂蚁每次周游结束后，不论蚂蚁搜索到的解如何，都将赋予相应的信息增量，比较差的解也将留下信息素，这样就干扰后续的蚂蚁进行寻优，造成大量的无效的搜索。改进的方法是，只有比较好的解才留下信息素，即只有当路径长度小于给定的值才留下信息素。

解非线性整数规划的蚁群算法如下：

(1) $nc\leftarrow 0$（nc为循环次数），给τ_{ij}赋相同的数值，产生大量的路径（如100条），从中选择比较优的（如30条），使这些路径留下信息素，给出Q、ρ的值；

(2) 将m个蚂蚁置于第一级；

(3) 对每个蚂蚁按转移概率p_{ij}选择该级中一个节点，每个蚂蚁走遍n个节点；

(4) 计算目标函数值f，函数值f小于给定值的路径按更新方程修改吸引强度，$nc\leftarrow nc+1$；

(5) 若$nc>$规定的循环次数，停止运行，根据τ_{ij}选择接点（若$\tau_{i_0j}=\max_i\tau_{ij}$，

则第 j 级中选择第 i_0 个节点)；否则转(2)。

2.1.3 算例分析

为验证上述蚁群算法求解整数规划的有效性，用蚁群算法解下列整数规划[32]：

$$\min F_1 = |x_1| + |x_2| + \cdots + |x_{10}|, -10 \le x_i \le 10, x_i \in Z(i = 1,2,\cdots,10)$$

$$\min F_2 = x_1^2 + x_2^2 + \cdots + x_{10}^2, -10 \le x_i \le 10, x_i \in Z(i = 1,2,\cdots,10)$$

$$\min F_3 = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - 10x_4)^4,$$
$$-10 \le x_i \le 10, x_i \in Z(i = 1,2,\cdots,4)$$

用蚁群算法仿真计算时，给 τ_{ij} 赋相同的数值 100，$Q = 2000$，$nc = 200$，蚂蚁数目 $m = 30$，ρ 取不同值进行测试，各算法运行 100 次，结果如表 2.1.1 所示。$nc = 200$，$m = 30$，$\rho = 0.9$，Q 取不同值进行测试，结果如表 2.1.2 所示。$nc = 200$，$\rho = 0.9$，$Q = 1500$，m 取不同参数下进行测试，结果如表 2.1.3 所示。

表 2.1.1 衰减系数 ρ 对算法性能的影响

函数	ρ	最坏解	平均解	最好解	最好解的次数
F_1	0.7	17	4.29	0	5
	0.8	12	2.52	0	10
	0.9	6	1.52	0	41
	0.95	23	6.87	2	0
	0.98	57	23.69	8	0
F_2	0.7	19	5.29	0	2
	0.8	8	2.81	0	9
	0.9	6	1.51	0	37
	0.95	25	7.15	1	0
	0.98	68	25.83	10	0
F_3	0.7	12	5.21	0	23
	0.8	4	1.97	0	29
	0.9	2	1.21	0	85
	0.95	19	5.12	0	19
	0.98	44	19.88	0	9

表 2.1.2 信息量 Q 对算法性能的影响

函数	Q	最坏解	平均解	最好解	最好解的次数
F_1	500	5	1.76	0	41
	1000	6	1.53	0	43
	1500	5	1.54	0	45
	2000	6	1.52	0	41
	2500	8	1.78	0	34
F_2	500	6	1.86	0	41
	1000	7	1.51	0	47
	1500	6	1.47	0	45
	2000	6	1.51	0	37
	2500	9	1.82	0	34
F_3	500	4	1.71	0	47
	1000	5	1.53	0	48
	1500	4	1.48	0	51
	2000	2	1.21	0	85
	2500	7	1.64	0	39

表 2.1.3 蚂蚁数 m 对算法性能的影响

函数	m	最坏解	平均解	最好解	最好解的次数
F_1	20	10	3.21	0	18
	30	5	1.54	0	45
	40	3	0.74	0	60
	50	3	0.19	0	89
	60	3	0.17	0	92
F_2	20	11	3.15	0	15
	30	6	1.47	0	45
	40	4	0.8	0	58
	50	3	0.22	0	87
	60	4	0.18	0	91
F_3	20	9	3.23	0	20
	30	4	1.48	0	51
	40	3	0.65	0	59
	50	2	0.18	0	89
	60	2	0.15	0	93

从表 2.1.1 可知，$\rho=0.9$ 时效果最好，特别要求 ρ 不能取太小。从表 2.1.2 可知，$Q=1000$ 时效果最好。这里若 Q 固定，ρ 太小，$\rho\tau_{ij}^{old}$ 较小，表示以前的激素影响力较小，体现不出蚁群算法的优点；ρ 太大，$\rho\tau_{ij}^{old}$ 较大，表示以前的激素影响力较大，体现不出每次蚂蚁运动后的影响，因此 ρ 取适中。对 Q 取值，情况

与ρ取值分析类似，其也需取适中。从表2.1.3可以看出蚂蚁数目越大效果越好，不过收敛速度慢，运行时间增加了。

用蚁群算法解决了典型的无约束的整数规划问题，对于有约束的整数规划可以把原约束方程作为罚函数项加入到原目标中，变成无约束的优化问题，同样可以解决。对该文的蚁群算法稍加修改，可解决类似的一些特殊的非线性整数混合规划问题。

2.2 武器-目标分配问题的蚁群算法

2.2.1 引言

武器-目标分配（Weapon Target Assignment）问题是现代战争中十分重要的问题，为解决这个问题，人们提出了许多算法[34,35]，Kuttar 提出的序列算法，把分配问题假定为按顺序逐个地进行，用迎击失败概率最小的方式去选择目标与迎击武器组合，收敛速度很慢；Castanon 提出用非线性网络流程求准最优解的算法，结果会产生较大的误差；E.Wacholker[34]提出了一种神经网络的解法，其依据 Hopfield 和 Tank 的神经网络模型，用此网络解 WTA 问题，此方法有时得不到稳定解。文献[36][37]对神经网络模型提出了改进算法。本节用蚁群算法来解决 WTA 问题。

2.2.2 WTA 问题

给定战况如下：有n个目标$T_1,T_2,\cdots,T_n$，迎击武器分布于m个武器平台$W_1,W_2,\cdots,W_m$，第i $(i=1,2,\cdots,m)$个武器平台最多可使用r_i个武器，对目标T_j最多可使用s_j个武器，武器平台W_i迎击目标T_j的概率为p_{ij} $(i=1,2,\cdots,m;j=1,2,\cdots,n)$，武器最佳分配以分配迎击武器迎击全部目标的失败概率最小为目标。

若分配了武器平台W_i迎击目标T_j，则$x_{ij}=1$，否则$x_{ij}=0$。WTA 问题的数学模型为：

$$\begin{aligned} &\min E=\sum_{j=1}^{n}\prod_{i=1}^{m}(1-p_{ij}x_{ij}) \\ &s.t.\quad \sum_{j=1}^{n}x_{ij}\le r_i \quad (i=1,2,\cdots,m) \\ &\qquad \sum_{i=1}^{m}x_{ij}\le s_j \quad (j=1,2,\cdots,n) \\ &\qquad x_{ij}=0,1 \end{aligned} \tag{2.2.1}$$

只有当$r_i=1(i=1,2,\cdots,m)$，$s_j=1(j=1,2,\cdots,n)$时，上述问题为可转化为指派问

题，可以用匈牙利法解[38]。对于一般优化问题实质是非线性 0-1 整数规划问题，属于 $NP-$难题，目前没有有效的算法解此问题，这里提出用蚁群算法来解决此问题。

2.2.3 武器-目标分配问题的蚁群算法

蚁群算法的关键是把实际问题转化为蚁群网络。这里把武器分配过程看作 m 个阶段，每个阶段分配一个武器迎击目标，组成网络图如图 2.2.1 所示。

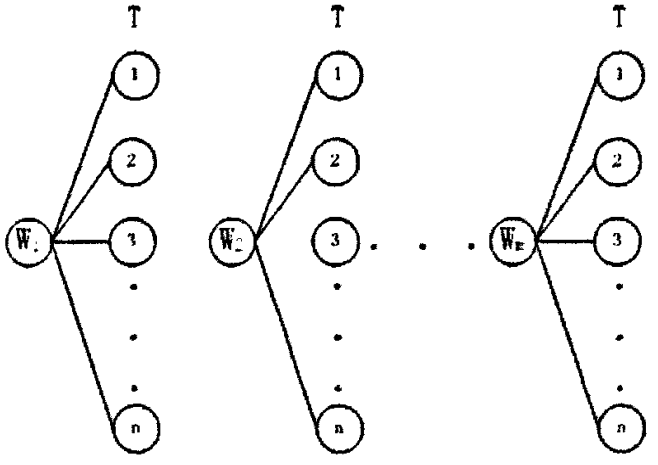

图 2.2.1 武器分配问题网络图

在 $W_i\ (i=1,2,\cdots,m)$ 处分别设置 r_i 个蚂蚁，每个蚂蚁从 W_i 处转移到 $T_1 \sim T_n$ 处，转移概率为：

$$q_{ij}=\frac{\tau_{ij}}{\sum_{i=1}^{n}\tau_{ij}} \tag{2.2.2}$$

更新方程为：

$$\tau_{ij}^{new}=\rho\tau_{ij}^{old}+\frac{p_{ij}Q}{E} \tag{2.2.3}$$

E 为此次分配全部目标的失败概率，ρ 表示强度的持久性系数，一般取 0.5～0.9 左右，Q 为一正常数。

解武器-目标分配的蚁群算法如下：

步骤 1　$nc \leftarrow 0$（nc 为循环次数），给 $\tau_{ij}\ (i=1,2,\cdots,m;j=1,2,\cdots,n)$ 矩阵赋相同的数值，给出 Q、ρ 的值；

步骤 2 对每个蚂蚁按转移概率 q_{ij} 选择下一个节点，要求：同一武器平台的蚂蚁不能转移到一起，且该节点的蚂蚁数量不能超过该目标的最多可使用的武器数量；

步骤 3 计算本次分配的全部目标的失败概率 E，按更新方程修信息强度；

步骤 4 比较出最好的结果（E 最小），$nc \leftarrow nc+1$；

步骤 5 若 $nc >$ 规定的循环次数，记录当前蚂蚁的位置（当前的解），停止运行，输出最好的解；否则转步骤 2；

2.2.4 仿真结果

2.2.4.1 与匈牙利法比较

有 4 个目标 T_1,T_2,T_3,T_4，迎击武器分布于 4 个武器平台 W_1,W_2,W_3,W_m，每个武器平台最多可使用 1 个武器，对每个目标最多可使用 1 个武器，武器平台 W_i 迎击目标 T_j 的概率为 p_{ij} $(i=1,2,\cdots,4; j=1,2,\cdots,4)$ 如表 2.2.1 所示。

表 2.2.1 $m=4,n=4$ 情况迎击概率表

目标 / 武器	1	2	3	4
1	0.88	0.75	0.77	0.86
2	0.8	0.86	0.76	0.75
3	0.81	0.76	0.74	0.77
4	0.73	0.72	0.69	0.71

此问题实质是指派问题，分别用匈牙利法和本文的蚁群算法解（$\rho=0.8$，$Q=100$），都可得到最优解："武器 1 迎击目标 4，武器 2 迎击目标 2，武器 3 迎击目标 1，武器 4 迎击目标 3"。由此说明此蚁群算法是合理的，对于 $m=n$ 比较小时，用匈牙利法解比较方便，当 $m=n$ 比较大时用蚁群算法效率较高。

2.2.4.2 算例分析

有 6 个目标 $T_1,T_2,\cdots,T_6$，迎击武器分布于 4 个武器平台 W_1,W_2,W_3,W_m，每个武器平台最多可使用武器为 (2,1,2,1)，对每个目标最多可使用 1 个武器，武器平台 W_i 迎击目标 T_j 的概率为 p_{ij} $(i=1,2,\cdots,4; j=1,2,\cdots,6)$ 如表 2.2.2 所示。

表 2.2.2 $m=6,n=4$情况迎击概率表

目标 / 武器	1	2	3	4	5	6
1	0.5	0.7	0.7	0.8	0.4	0.8
2	0.4	0.8	0.5	0.7	0.6	0.9
3	0.8	0.7	0.7	0.7	0.6	0.5
4	0.5	0.7	0.6	0.8	0.7	0.9

采用蚁群算法（$\rho=0.8$，$Q=100$）得到最优解“武器 1 迎击目标 4 和目标 6，武器 2 迎击目标 2，武器 3 迎击目标 1 和目标 3，武器 4 迎击目标 5”，循环了 50 次，最优解的迭代过程如图 2.2.2 所示。武器目标分配问题是一个非线性整数混合规划问题，特别当系统级数n很大时，用蚁群算法解决武器目标分配问题确实有效。

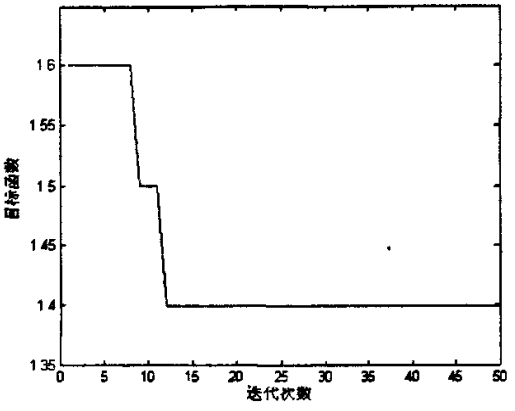

图 2.2.2 最优解的迭代过程

2.3 多处理机调度问题的蚁群算法

2.3.1 引言

所谓多处理机调度问题[39,40]（multiprocessor scheduling problem）是指有n台相同的处理机$P_1,P_2,\cdots,P_n$，处理m个独立的作业$J_1,J_2,\cdots,J_m$，以互不相关的方式工作，任何作业可以在任何处理机上运行，未完工的作业不允许中断。作业也不能拆分成更小的子作业。调度的任务是给出一种作业调度方案，使m个作业尽可能短的时间内由这n台相同的处理机完成。本节研究了用蚁群算法来解决此问题，并与贪心法和模拟退火算法作了比较。

2.3.2 多处理机调度问题数学模型

已知作业J_i需要的处理机时间为$t_i,(i=1,2,\cdots,m)$，若作业i分配到处理机j上

处理，则令 $x_{ij}=1$，否则令 $x_{ij}=0$。$\sum_{i=1}^{m}x_{ij}t_i$ 表示处理机 j 完工时间，$\sum_{j=1}^{n}x_{ij}=1$ 表示作业 i 只能分配到一个处理机上，因此多处理机调度问题的数学模型为：

$$\begin{aligned}&\min\max_{1\le j\le n}\sum_{i=1}^{m}x_{ij}t_i\\&s.t.\quad\sum_{j=1}^{n}x_{ij}=1\quad(i=1,2,\cdots,m)\\&\quad x_{ij}=0,1\end{aligned}\tag{2.3.1}$$

把（2.3.1）变换如下：

$$\begin{aligned}&\min v\\&s.t.\quad\sum_{j=1}^{n}x_{ij}=1\quad(i=1,2,\cdots,m)\\&\quad\sum_{i=1}^{m}x_{ij}t_i\le v\quad(j=1,2,\cdots,n)\\&\quad x_{ij}=0,1\end{aligned}\tag{2.3.2}$$

上式实质是线性 0-1 整数规划问题，属于 NP－难题，目前没有有效的算法解此问题。

2.3.3 解多处理机调度问题模拟退火算法

模拟退火算法[41,42,43]用于优化问题的出发点是基于物理中固体物质的退火过程与一般优化问题的相似性。算法的基本思想是从一给定解开始的，从邻域中随机产生另一个解，接受准则允许目标函数在有限范围内变坏，它由一控制参数 t 决定，其作用类似于物理过程中的温度 T，对于控制参数 t 的每一取值，算法持续进行“产生新解-判断-接受或舍弃”的迭代过程，对应着固体在某一恒定温度下趋于热平衡的过程。经过大量的解变换后，可以求得给定控制参数 t 值时优化问题的相对最优解。然后减小控制参数 t 的值，重复执行上述迭代过程。当控制参数逐渐减小并趋于零时，系统亦越来越趋于平衡状态，最后系统状态对应于优化问题的整体最优解，该过程也称冷却过程。

解多处理机调度问题模拟退火算法的步骤为：

（1）给定起、止“温度” T、T_0 和退火速度 α，处理机数目 n，作业数目 m，随机给出一个调度方案 $X_0=(x_{ij})_{n\times m}$，计算完工时间 f_0；

（2）若 $T>T_0$，转（3），否则算法停止，输出 X_0；

（3）随机产生作业 j 和处理机 i，令 $x_{kj}=0(k=1,2,\cdots,n,k\neq i)$，$x_{ij}=1$，此时

变量记为 X_1；

（4）计算完工时间 f_1，$\Delta E = f_1 - f_0$，若 $\Delta E \le 0$，接受新值，$X_0 \leftarrow X_1$，$T \leftarrow \alpha T$，转（2）；否则若 $\exp(-\Delta E/T) > rand(0,1)$，也接受新值，$X_0 \leftarrow X_1$，$T \leftarrow \alpha T$，转（2）；否则转（3）；

2.3.4 解多处理机调度问题蚁群算法

在作业 $J_i\ (i = 1,2,\cdots,m)$ 处分别设置 1 个蚂蚁，作业分配给处理机 j，蚂蚁就在作业 i 和处理机 j 的路径上留下外激素 τ_{ij}，第 i 个蚂蚁选择处理机 j 概率为：

$$p_{ij} = \frac{\tau_{ij}}{\sum_{j=1}^{n} \tau_{ij}} \qquad (2.3.3)$$

更新方程为：

$$\tau_{ij}^{new} = \rho \tau_{ij}^{old} + \frac{Q}{F} \qquad (2.3.4)$$

F 为此次分配后完工时间，ρ 表示强度的持久性系数，一般取 0.5～0.9 左右，Q 为一正常数。

解多处理机调度的蚁群算法如下：

(1) $nc \leftarrow 0$（nc 为循环次数），给 $\tau_{ij}\ (j = 1,2,\cdots,n)$ 赋相同的数值，给出 Q、ρ 的值，随机给出一个调度方案；

(2) 对每个蚂蚁按转移概率 p_{ij} 选择下一个节点，计算本次分配完工时间 F，按更新方程修信息强度；

(3) 比较这次循环的结果，若目标函数 F 有改进，保留当前解为最好解，否则，外激素量采用上次最好解时的外激素量，$nc \leftarrow nc + 1$；

(4) 若 $nc >$ 规定的循环次数，记录当前蚂蚁的位置（当前的解），停止运行，输出最好的解；否则转(2)；

2.3.5 算法比较

贪心法[39,40]的思路为先将作业按运行时间的长短从大到小排成非递增序，然后给空闲的处理机依次分配作业。

例如设有 3 台处理机和 9 个作业，作业需要的运行时间分别为 81，40，26，4，65，98，53，71，15。按贪心法，调度结果为 P_1（98，40，4），P_2（81，53，26）

和P_3（71，65，15），完工时间为160。

仍以上面数值为例，采用模拟退火算法，起始温度$T=20000$，终止温度$T_0=1$，退火速度$\alpha=0.95$，测试50次，得到的解如表2.3.1。图2.3.1是模拟退火算法的过程，纵坐标为完工时间，横坐标为迭代算迭代次数。采用蚁群算法（$\rho=0.8$，$Q=100$），循环了50次，解的迭代过程如图2.3.2所示。测试50次，得到的解如表2.3.1。最好解为：P_1（98，53），P_2（81，40，26，4）和P_3（71，65，15），完工时间为151，它是最佳调度方案。

表2.3.1 模拟退火算法与蚁群算法结果比较

算法	结果比较			
	平均时间（S）	平均值	最好解	最差解
模拟退火算法	0.58	165.2	151	177
蚁群算法	0.14	163.6	151	167

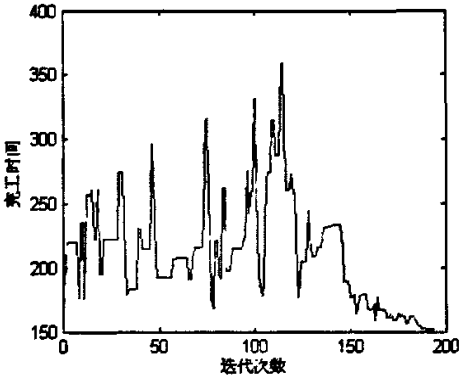

图2.3.1 模拟退火算法的计算过程

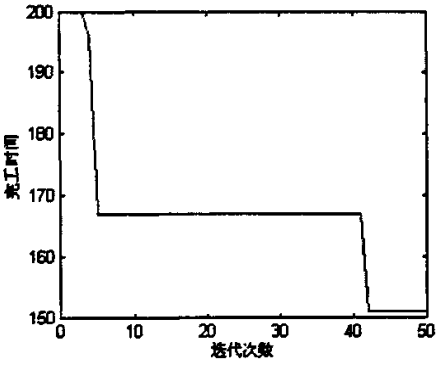

图2.3.2 蚁群算法的计算过程

对于多处理机调度问题，用贪心法[36,37]，但此方法虽然简单，但往往得不到最优解。对于模拟退火算法，由于固体退火必须缓慢降温，才能使固体在每一温度

下都达到热平衡，最终趋于平衡状态，因此，控制参数的值必须缓慢衰减，才能确保模拟退火算法最终趋于优化问题的整体最优解，因此迭代次数一般较长。蚁群算法也采用概率改变变量，模拟退火算法在解的附近随机地找下一个解，以概率方式选取下一个解，而蚁群算法考虑到了附近解“外激素”，好的解附近的“外激素”较多，它被选取的概率就大，蚁群算法的迭代次数一般比较少，从表 2.3.1 也可以得到证实，并且其方法相对比较有效。多处理机调度问题是一个线性整数规划问题，特别当作业数目和处理机数很大时，用蚁群算法解决多处理机调度问题确实有效。

2.4 可靠性优化的蚁群算法

2.4.1 引言

在工业、军事和日常生活的许多方面，系统可靠性的性能对于各种条件下的任务来说，都极其重要，因此可靠性问题是系统设计、研究和运行过程中必须考虑的关键因素之一。由于设计时受到资源限制，包括费用、重量、体积、功耗等受到限制，系统可靠性最优问题引起了广泛的重视和研究，可靠性优化是可靠性工程中的一项重要工作。有许多改进系统可靠性的方法，但实践认为比较好的要算最优冗余这一方法。

2.4.2 最优冗余优化模型及解法

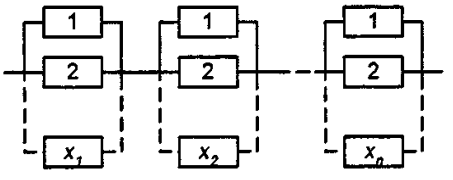

图 2.4.1 串-并联系统

假设系统由 n 个独立子系统组成，在每个子系统中使用同一种部件（如图 2.4.1），冗余优化模型为[44,45]：

$$\begin{aligned} &\min C_s = \sum_{i=1}^{n} c_i x_i \\ &s.t. \quad \prod_{i=1}^{n} R_i(x_i) \geq R_0 \\ &\quad x_i = 1,2,\cdots,(i=1,2,\cdots,n) \end{aligned} \tag{2.4.1}$$

这里：C_s—系统费用；R_S—系统可靠度；c_i—第 i 种部件的单价；x_i—第 i 子系统冗余第 i 种部件的个数，$x_i \geq 1$；p_i—第 i 种部件的可靠度；$R_i(x_i)$—第 i 子系统的可靠度，$R_i(x_i)=1-(1-p_i)^{x_i}$；R_0—系统要达到预定的可靠度。

上述可靠性优化模型属于 NP－难题。系统冗余可靠性优化方法较多，文献[45]

中作了综述，推荐了十几种方法，但在用于大规模非线性规划问题时，仅有少数算法被证明是有效的，没有哪一种算法被证明比其它的算法更优越。例如启发式算法虽简单直观，但启发式信息不易找，而且有点盲目性。整数规划得到的是整数解，但为了应用整数规划，把非线性目标函数和约束条件转化成线性形式，则是一项困难的任务；动态规划存在着维数“灾难”问题，并且对三个以上约束问题，求解相当困难。极大值原理求解三个以上约束问题也是困难的；几何规划只限于求解可用泊松函数形式表述的问题；序列无约束极小化方法、修正的单纯形序列搜索、广义的拉格朗日函数法是用于大型非线性规划问题中被证明是有效的方法，虽然解是非整数，但解决非冗余问题很有效。对于同时确定最优部件数和可靠度的问题，是一类混合整数非线性问题，解决这类问题很困难。文献[45]推荐了将 Hooke 和 Jeeves 等人提出的模式搜索法与 Aggarwal 等人提出的启发式算法组合起来的方法。这种方法首先假设部件的可靠度，然后用启发式算法确定最优冗余数，最后用 Hooke 和 Jeeves 的模式搜索法来进行序列搜索。20 世纪 80 年代以来，一些新颖的优化算法，如人工神经网络、遗传算法、模拟退火、蚁群算法以及混合优化策略等，通过模拟或揭示某些自然现象或过程而得到发展，为解决复杂问题提供了新的思路和手段。本节分别采用模拟退火算法、遗传算法和蚁群算法分别来解，最后作一比较。

2.4.3 可靠性优化的模拟退火算法

解可靠性优化的模拟退火算法的算法如下：

(1)给定起、止“温度” $T=100000$、$T_0=1$和退火速度$\alpha=0.9$，模拟参数初始化 X_0， p， C；

(2)若$T>T_0$，转（3），否则算法停止，输出 X_0；

(3)计算目标函数值$CS_0=CX_0^T$；

(4)随机产生变量 x_j，若 $rand(0,1)\geq 0.5$，则 $x_j \leftarrow x_j+1$，否则 $x_j \leftarrow x_j-1$，此时变量记为 X_1；

(5)判断是否满足约束条件，若满足转（6），否则转（4）；

(6)计算目标函数 $CS_1=CX_1^T$， $\Delta E=CS_1-CS_0$，若 $\Delta E\leq 0$，接受新值，$X_0 \leftarrow X_1$，$T \leftarrow \alpha T$，转(2)；否则若 $\exp(-\Delta E/T)>rand(0,1)$，也接受新值，$X_0 \leftarrow X_1$，$T \leftarrow \alpha T$，转（2）；否则转（4）；

2.4.4 可靠性优化的遗传算法

遗传算法[41,43]是基于自然遗传和自然优选机理的寻优方法。所谓自然遗传和自然优选来自于达尔文的进化论学说，该学说认为在生物进化过程中，任一动植物经过若干代的遗传和变异，使之能够适应新的环境，是优胜劣汰的结果，这种自然遗传思想也适用于求解优化问题。GA采用选择(selection)、交叉(crossover)和变异(mutation)运算来实现“物竞天择,适者生存”这一自然法则的模拟。

为了用遗传算法，把目标函数取最小值改为求最大值，数学优化模型为：

$$\begin{aligned} &\max M-\sum_{i=1}^{n} c_i x_i \\ &s.t. \quad \prod_{i=1}^{n} R_i(x_i) \geq R_0 \\ &\quad x_i = 1,2,\cdots,(i=1,2,\cdots,n) \end{aligned} \tag{2.4.2}$$

M为一个大的正数，如$M=250$。

这里不采用二进制编码，用染色体$X=(x_1,x_2,\cdots,x_n)$作为解的代码。对于染色体个数N、交叉概率P_c、变异概率P_m和遗传代数tt的选取值得研究，染色体个数越大，其代表性越大，最终进化到最优解的可能性越大，但会造成计算量的增加，一般取适度如$N=30$[41]。交叉概率或变异概率过小使解有一定的局限性，便历性较差，交叉概率或变异概率较大使得进化的随机性增大，而不容易得到稳定的解，因此也取适度。遗传代数越大，进化最优解的可能性大，但计算量大，也要取适度。其遗传算法的为：

(1)给出P，C，输入染色体个数$N=30$，交叉概率$P_c=0.2$，变异概率$P_m=0.5$，下一代的代数$tt=100$，$t=1$；

(2)产生$N=30$个并满足约束条件的染色体；

(3)若$t>tt$，则停止输出结果；否则传（4）

(4)计算目标函数$f_i(i=1,2,\cdots,N)$，并按f_i递减顺序对染色体排序，计算累积概率$ps_0=0$，$ps_i=\dfrac{\sum_{j=1}^{i} f_j}{\sum_{j=1}^{N} f_j}(i=1,2,\cdots,N)$；

(5)选择操作：产生[0，1]的随机数rnd，若$ps_{j-1}<rnd\leq ps_j$，选择染色体j，这样选择$N=30$个；

(6)交叉操作：产生$N=30$个[0，1]的随机数$rnd_i\ (i=1,2,\cdots,N)$，对每个染色体进行判断，若$rnd_i\leq p_c$，对染色体i进行交叉，随机产生染色体j和随机数α，

交叉后的染色体为$\alpha X_i+(1-\alpha)X_j$；否则不进行交叉；

(7)变异操作：产生$N=30$个[0，1]的随机数$rnd_i\ (i=1,2,\cdots,N)$，对每个染色体进行判断，若$rnd_i \le p_m$，对染色体i进行变异，随机产生一个方向D，变异后的染色体为X_i+mD（m为系数，如为5），若变异后的染色体不可行，可减小m值，直到可行；否则不进行变异；

(8)保留所有代中的最好的染色体，$t \leftarrow t+1$，转（3）；

2.4.5 可靠性优化的蚁群算法

首先把原约束方程作为罚函数项加入到原目标中，变成无约束的优化问题，即

$$\min\sum_{i=1}^{n}c_i x_i+M\left\{\min\left\{0,\left[\prod_{i=1}^{n}(1-R_i)^{x_i}-R_0\right]\right\}\right\}^2 \qquad (2.4.3)$$

其中M为一充分大的正数。

由于受到费用、体积、重量的限制，冗余数量x_i（$i=1,2,\cdots,n$）不可能很大，设最大为$x_{\max}$，可靠性框图可转化成如下的网络图（图 2.4.2）。每一级有$x_{\max}$个节点，表示$1\sim x_{\max}$个冗余数目共有$x_{\max}\times n$个节点。从第 1 级到第n级之间的连接在一起，组成空间一个解，如图 2.4.2 表示解(3,2,1,⋯,1)。

对每一级设置 1 个蚂蚁（也可多于 1 个），每个蚂蚁只在本级$1\sim x_{\max}$节点之间转移，蚂蚁在j级中转移到本级中第i个节点的概率p_{ij}为：

$$p_{ij}=\frac{\tau_{ij}}{\sum_{i=1}^{x_{\max}}\tau_{ij}} \qquad (2.4.4)$$

更新方程为：

$$\tau_{ij}^{new}=\rho\tau_{ij}^{old}+Q \qquad (2.4.5)$$

τ_{ij}理解为第j级第i个节点的吸引强度。ρ表示强度的持久性系数，一般取0.5～0.9左右，Q为一正常数。η_{ij}表示目标函数差，$\eta_{ij}>0$，目标函数值降低，蚂蚁从第i个节点转移到个第j个节点；$\eta_{ij}\le 0$，蚂蚁维持原状。

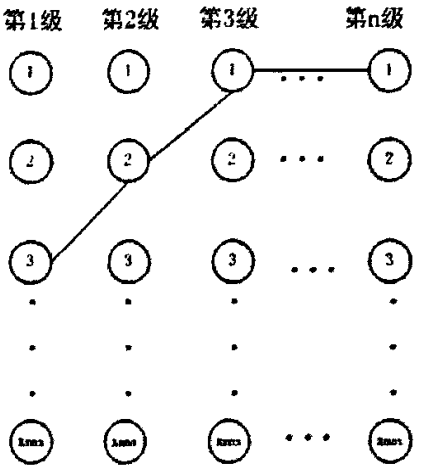

图 2.4.2 可靠性网络图

解可靠性优化的蚁群算法如下：

(1) $nc \leftarrow 0$（nc为循环次数），给τ_{ij}矩阵赋相同的数值，给出Q、ρ的值；

(2) 将n个蚂蚁置于每一级的第一个节点（相当于给出初解(1,1,…,1)；

(3) 对每个蚂蚁按转移概率p_{ij}选择下一个节点，如目标函数差$\eta_{ij} > 0$就转移到该节点；

(4) 按更新方程修改吸引强度，$nc \leftarrow nc+1$；

(5) 若$nc >$规定的循环次数，记录当前蚂蚁的位置（当前的解），停止运行；否则转(3)；

2.4.6 算例分析

以一个实例来比较 3 个算法，已知 5 种部件的可靠度和费用分别为$p_1 = 0.96$，$p_2 = 0.93$，$p_3 = 0.85$，$p_4 = 0.80$，$p_5 = 0.75$，$c_1 = 3$元，$c_2 = 12$元，$c_3 = 8$元，$c_4 = 5$元，$c_5 = 10$元，要求$R_0 = 0.9$。

模拟退火算法参数初始化$X_0 = (4,4,4,4,4)$，对大多数优化问题而言，模拟退火算法要优于局部搜索算法，所得近似最优解的质量也比局部搜索算法好，图 2.4.3 是模拟退火算法的过程。

遗传算法擅长全局搜索，以其简单通用、鲁棒性强、适于并行处理等为特点。尤其适用于处理传统搜索方法难以解决的复杂问题和非线性问题，图 2.4.4 是遗传算法的计算过程。

蚁群算法的初始值：$x_{max} = 4$，$\rho = 0.9$，$[\tau_{ij}]_{4\times5} = [10]_{4\times5}$，$M = 10^6$，$Q = 10$，

图 2.4.5 是蚁群算法的过程。图 2.4.3、图 2.4.4 和图 2.4.5 的纵坐标为费用，横坐标为解的变化次数。

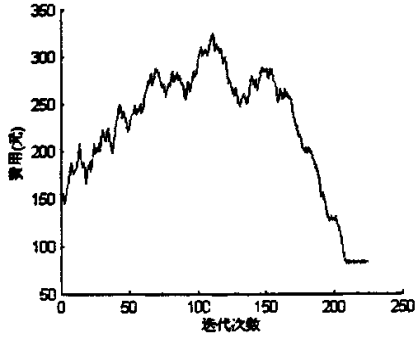

图 2.4.3 模拟退火算法的计算过程

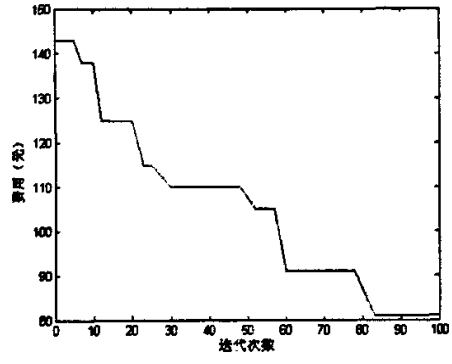

图 2.4.4 遗传算法的计算过程

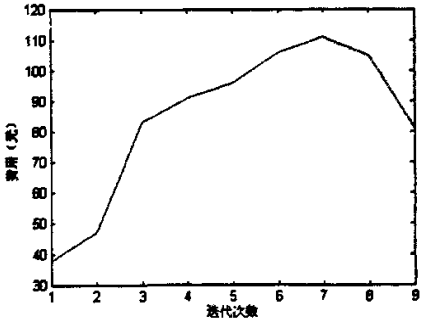

图 2.4.5 蚁群算法的计算过程

各种算法各随机测试 50 次，结果如表 2.4.1 所示。实际最优解(2,2,2,3,2)，此时总费用为 81 元。从表 2.4.1 可知，用蚁群算法比模拟退火算法和遗传算法的效率更高，推荐采用蚁群算法。可靠性优化问题是一个非线性整数混合规划问题，特别当系统级数 n 很大时，模拟退火算法和遗传算法均可以运用，经过比较蚁群算法解决可靠性优化问题比较有效。

表2.4.1 各种策算法比较

各种算法	结果比较				
	平均时间（S）	平均值（元）	最差解（元）	最好解（元）	最好解次数
退火算法	27.6	87.6	109	81	8
遗传算法	15.2	87.3	102	81	12
蚁群算法	0.2	86.2	98	81	22

2.5 求解旅行商问题的多样信息素的蚁群算法

2.5.1 信息素更新的三个模型

绪论中已介绍过求解旅行商问题的基本蚁群算法，蚂蚁完成一次循环，各路径上信息量要做调整，要根据具体问题而定。M.Dorigo 曾给出三种不同模型[7]，分别称为 ant-cycle system、ant-quantity system、ant-density system，它们的差别在于$\Delta\tau_{ij}^{k}$表达式的不同。它们的区别在于：后两种模型中，利用的是局部信息，而前者利用的是整体信息，在求解 TSP 问题时，性能较好，因而通常采用它，其被称为基本模型。

2.5.2 多样信息素更新规则

大多数文献采用 ant-cycle system 和 ant-quantity system 这两个系统，文献[7]还对这三个模型作了比较，认为 ant-cycle system 模型较好。实际上这三个模型在某种意义上有其合理的一面。真正蚁群社会中的蚁群是有组织的、有分工的，不同的蚁群有不同的信息素调控机制。采用单样信息素更新规则，并不全面，这里提出把蚁群分成三种，分别采用这三种模型，混合更新信息素。这种混合更新规则既利用了局部信息，又考虑了整体信息。将局部搜索和全局搜索相结合，可以使收敛性得到提高。为了避免搜索停滞，路径上的信息素浓度被限制在$[\tau_{\min},\tau_{\max}]$范围内，另外，信息素的初始值被设为其取值上限，这样有助于增加算法初始阶段的搜索能力[8]。该方法称为 MAX-MIN Ant System。这里也采用 MAX-MIN Ant System 技术。

多数文献[7]对α、β、ρ选取作了研究，对初始信息数C和Q的选取未考虑，实际上C与$\Delta\tau_{ij}$是有联系的，$\Delta\tau_{ij}$不能比C小很多，否则影响力很小；$\Delta\tau_{ij}$比C大很多也不好，否则前几只蚂蚁的影响力过大，容易陷入局部最优。因此要$\Delta\tau_{ij}$取合适一点，经过实验发现$\Delta\tau_{ij}$取$\frac{1}{5}C$左右时，效果较好。因此公式（1.4）、（1.5）和（1.6）中的Q是不一样的，要根据实际问题进行估算。

2.5.3 算法测试

选用 Oliver30(最好解为 423.7406)和 att48(TSPLIB 提供的最好解为 33522)作为实验例子，来研究三种蚁群的蚂蚁数的比例 $m_1:m_2:m_3$ 的选取。算法参数如下：$\alpha=1$，$\beta=5$，$m=60$，$C=10$，$\rho=0.99$，Q 的设置如表 2.5.1 所示，对各种算法测试 20 次，结果如表 2.5.2 所示。从表 2.5.2 可以看出，采用多样信息素更新规则的混合效率较高，特别蚂蚁数的比例 $m_1:m_2:m_3$ 取 2：1：1 的效果更好。图 2.5.1 是解 Oliver30 混合算法最好的解，总路程为 423.7406。图 2.5.2 是解 att48 混合算法最好的解，总路程为 33522。图 2.5.3 是 Oliver30 混合算法次好解，总路程为 424.5730。利用蚁群算法信息更新特性，提出的多样信息素的蚁群算法可以显著提高计算效率，具有较大的实用价值。蚁群算法研究处于初期，还有许多问题值得研究，如算法的参数选择只能通过仿真实验，无法给出理论指导。

表 2.5.1 Q 的取值

Q	Oliver30	att48
公式（4）	1000	100000
公式（5）	100	10000
公式（6）	2	2

表 2.5.2 几种算法测试结果 424.5730

算法		Oliver30			att48		
		最好解	平均值	最差解	最好解	平均值	最差解
ant-density system		425.6490	429.1071	433.1215	33786	35688	36559
ant-quantity system		424.5730	429.7738	435.2420	33902	35714	36111
ant-cycle system		423.7406	429.7032	432.4568	33780	35595	36534
$m_1:m_2:m_3$	1:1:1	423.7406	429.2475	432.4168	33896	35792	36357
	2:1:1	423.7406	427.9214	431.4568	33522	35561	36100
	3:2:1	423.7406	429.3649	432.7944	33657	35599	36297

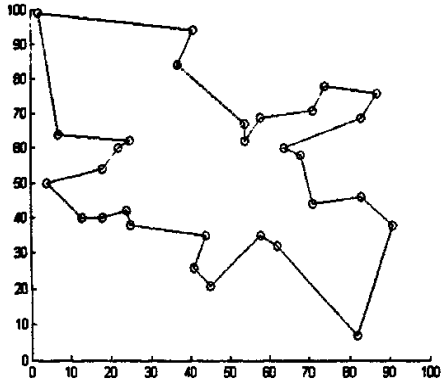

图 2.5.1 用混合算法解 Oliver 的最好的解(423.7406)

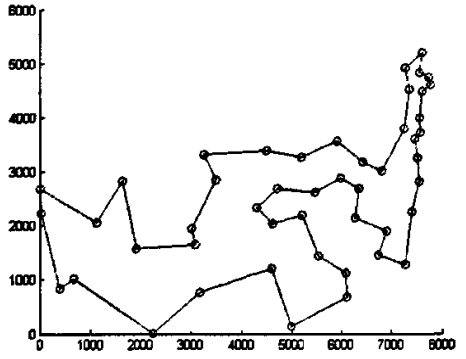

图 2.5.2 用混合算法解 att48 的最好的解(33522)

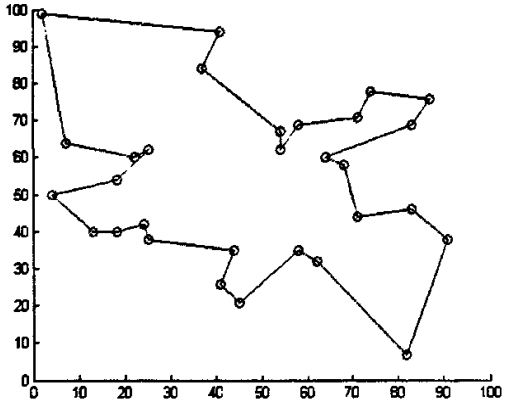

图 2.5.1 用混合算法解 Oliver 的次好的解（424.5730）

2.6 本章小结

提出了一种新的蚁群算法来解一般非线性整数规划问题。特殊的问题可以采用特殊的方法，对几个典型的整数规划问题，如武器-目标分配、多处理机调度、可靠性优化等问题，根据各问题的特点，采用了不完全相同方法，并与其他方法作了比较，效果比有效。

提出了一种求解旅行商问题的多样信息素的蚁群算法。根据蚁群算法信息素更新的特性，把蚁群的三种不同的信息素更新方式混合在一起，既利用了局部信息，又考虑了整体信息，将局部搜索和全局搜索相结合，使收敛性得到提高。针对旅行商问题的仿真实验结果表明了该混合算法的有效性。

3 连续优化问题的蚁群算法研究

3.1 无约束非线性最优化问题

对于连续有约束非线性最优化问题，可以通过惩罚函数法转化为连续无约束非线性规划问题[38, 46]，因此这里只研究连续无约束非线性最优化问题。

假设求解：$\min f(x_1,x_2,\cdots,x_n)$，对于非线性规划，目前还没有适合各种问题的一种解法，各个方法都有自己特定的适用范围。对于解析法，要求目标函数与约束函数具有连续性并且其导数存在。但在某些实际问题中，由于目标函数很复杂，有时甚至无法写出其表达式，当然更无法求的其导数，这样解析法就不再适用了，一般采用直接法，如网格法，随机优化法等。文献[47]将解空间划分成若干个子域，根据信息量求出解所在的子域，在该子域内已有的解中确定解的具体值。文献[48]将传统蚁群算法中的“信息量留存”过程拓展为连续空间中的“信息量分布函数”，根据信息量分布函数决定移动方向。本章提出一种简单的蚁群算法来解连续空间优化问题。本章的蚁群算法也属于直接法一种，不需要求导数等信息。

3.2 连续优化问题的蚁群算法

求解连续优化问题的蚁群算法思路为：首先可根据问题的性质估计一下最优解的范围，估计出各变量的取值范围：$x_{jl} \le x_j \le x_{ju}$ $(j=1,2,\cdots,n)$。在变量区域内打网格，空间的网格点上对应于一个状态，人工蚂蚁在各个空间网格点之间移动，根据各网格点的目标函数值，留下不同的信息量，以此影响下一批人工蚂蚁的移动方向。循环一段时间后，目标函数值小的网格点信息量比较大。根据信息量，找出信息量大的空间网格点，缩小变量范围，在此点附近进行人工蚁群移动，重复前述过程，直到网格的间距小于预先给定的精度，算法终止。

假设各变量分成N等份，n个变量变成n级决策问题，每一级有$N+1$个节点，如图 3.1 所示。共有$(N+1)\times n$个节点。从第 1 级到第n级之间的连接在一起，组成空间一个解，如图 3.1 状态为(3,2,1,$\cdots$,1)，其对应的解为$(x_1,x_2,\cdots,x_n)=$

$$(x_{1l}+\frac{x_{1u}-x_{1l}}{N}\times 3, x_{2l}+\frac{x_{2u}-x_{2l}}{N}\times 2, x_{3l}+\frac{x_{3u}-x_{3l}}{N}\times 1,\cdots, x_{nl}+\frac{x_{nu}-x_{nl}}{N}\times 1)。$$

对每一级设置 1 个蚂蚁（也可多于 1 个），每个蚂蚁只在本级节点之间转移，蚂蚁在本级中第j节转移到本级中第i个节点的概率p_{ij}为：

$$p_{ij}=\frac{\tau_{ij}}{\sum_{i=1}^{N}\tau_{ij}} \qquad (3.1)$$

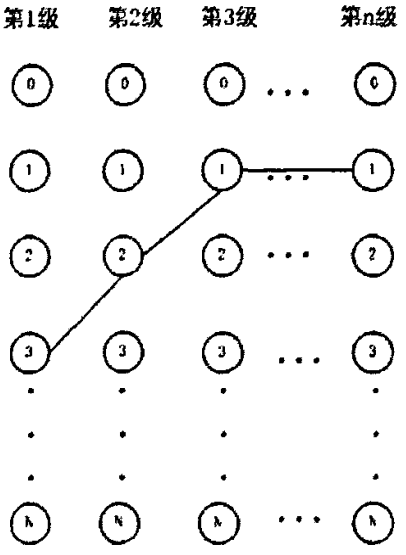

图 3.1 状态空间解

更新方程为：

$$\tau_{ij}^{new} = \rho\tau_{ij}^{old} + Q/f \tag{3.2}$$

其中：τ_{ij} 理解为第 j 级第 i 个节点的吸引强度；ρ 表示强度的持久性系数，一般取 0.5～0.9 左右；Q 为一正常数；f 为目标函数值，为保证 $f>0$，在原目标函数的基础上加上一个大的常数。

解连续优化问题的蚁群算法如下：

步骤 1 估计出各变量的取值范围：$x_{jl} \le x_j \le x_{ju}\ (j=1,2,\cdots,n)$；

步骤 2 对各变量分 N 等份，$h_j = \dfrac{x_{ju}-x_{jl}}{N}\ (j=1,2,\cdots,n)$；

步骤 3 若 $\max(h_1,h_2,\cdots,h_n)<\varepsilon$，算法停止，最优解为 $x_j^* = \dfrac{x_{jl}+x_{ju}}{2}\ (j=1,2,\cdots,n)$；否则转步骤 4；

步骤 4 $nc \leftarrow 0$（nc 为循环次数），给 τ_{ij} 矩阵赋相同的数值，给出 Q、ρ 的值；

步骤 5 假设蚂蚁数为 num_ant，对每个蚂蚁按转移概率 p_{ij} 选择下一个节点；

步骤 4 按更新方程修改吸引强度，$nc \leftarrow nc+1$；

步骤 6 若 nc<规定的循环次数，转步骤 5；否则，找出 τ_{ij} 矩阵中每列最大的元素对应的行 $(m_1,m_2,\cdots,m_n)$，缩小变量的取值范围：$x_{jl} \leftarrow x_{jl}+(m_j-\Delta)h_j$，

$x_{jm} \leftarrow x_{j1} + (m_j + \Delta)h_j$，$(j = 1,2,\cdots,n)$，转步骤 2；

3.3 数值分析

对于典型的优化问题 $\min f(x_1,x_2) = (x_1-1)^2 + (x_2-2.2)^2 + 1$

连续优化问题的蚁群算法的参数为：$[x_{1l},x_{1u}] = [0,2]$，$[x_{2l},x_{2u}] = [1,3]$，蚂蚁数 $num\_ant = 20$，$\varepsilon = 0.001$，$N = 10$，$Q = 10$，$NC = 100$（迭代次数），蚁群计算过程如图 3.2 所示。图 3.2 中的“0”表示迭代过程的变量范围的中点值。最后得到最优解 $(x_1,x_2) = (1.0366,2.1928)$，与真实解 $(x_1^*,x_2^*) = (1,2.2)$ 很接近。

连续优化问题的蚁群算法与网格法类似，网格法就是在变量区域内打网格，在网格点上求约束函数与目标函数的值，对于满足约束条件的点，再比较其目标函数的大小，从中选择小者，并把该网格点作为一次迭代的结果。然后在求出的点附近将分点加密，再打网格，并重复前述计算与比较，直到网格的间距小于预先给定的精度，终止迭代。网格法只利用了最小值这一点的信息，而连续优化问题的蚁群算法利用了每一点的信息，通过更新方程表现出来，使目标函数值小的空间点，其吸引强度大。因而连续优化问题的蚁群算法的效率比网格法的效率高。

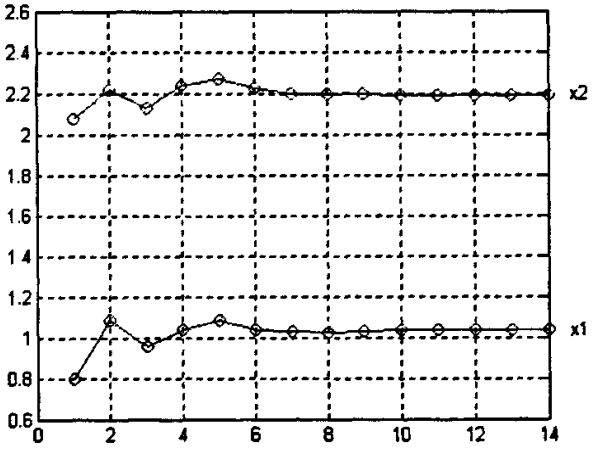

图 3.2　蚁群算法的计算过程

3.4 本章小结

对连续优化问题的蚁群算法提出了一种新的思路，把连续空间进行离散化，分成若干空间网格点，采用蚁群算法找出信息量大的空间网格点，缩小变量范围，在此点附近进行人工蚁群移动，直到网格的间距小于预先给定的精度。此方法简单直观。

4 聚类问题的蚁群算法

4.1 引言

聚类分析是按不同对象之间的差异，根据特定的准则作模式分类，其应用面相当广泛。大致分“分类数目未知”和“分类数目已知”两类问题。聚类分析方法比较多，比如对于“分类数目已知”聚类算法有 K-均值算法，ISODATA 算法，修正的 ISODATA 算法[49, 50]等。本章分别用蚁群算法和 K-均值算法与蚁群算法混合算法来解决此问题，并与 K-均值算法、模拟退火算法作了比较。

4.2 聚类问题的数学模型

已知模式样本集 $\{X\}$ 有 n 个样本和 K 个模式分类 $\{S_j, j=1,2,\cdots,K\}$，以每个模式样本到聚类中心的距离之和达到最小为准则，其数学模型为：

$$\min \sum_{j=1}^{K} \sum_{X \in S_j} \|X - m_j\| \tag{4.1}$$

式中 K 为聚类数目，m_j 为 j 类样本的均值向量。若模式样本 i 分配第 j 聚类中心，则令 $y_{ij}=1$，否则令 $y_{ij}=0$。$m_j = \dfrac{1}{\sum_{i=1}^{n} y_{ij}} \sum_{i=1}^{n} y_{ij} X_i$，$\sum_{j=1}^{n} y_{ij} = 1$ 表示模式样本 i 只能分配到一个聚类中心上，因此聚类问题的数学模型为：

$$\begin{aligned} &\min \sum_{i=1}^{n} \sum_{j=1}^{K} (y_{ij} \|X_i - m_j\|) \\ &s.t. \quad \sum_{j=1}^{K} y_{ij} = 1 \quad (i=1,2,\cdots,n) \\ &\qquad m_j = \frac{1}{\sum_{i=1}^{n} y_{ij}} \sum_{j=1}^{n} y_{ij} X_i \quad (j=1,2,\cdots,K) \\ &\qquad y_{ij} = 0,1 \end{aligned} \tag{4.2}$$

此问题是一个非线性规划问题，目前没有有效的算法解此问题。

4.3 K 均值算法

K-均值算法的步骤如下[49, 50]：

（1）任选 K 个初试聚类中心：$z_1, z_2, \cdots, z_K$；

（2）逐个将样本集 $\{X\}$ 中各个样本按最小距离原则分配给 K 个聚类中心的某一个 z_j；

（3）计算新的聚类中心$z'_j\ (j=1,2,\cdots,K)$，即$z'_j=\frac{1}{N_j}\sum_{X\in S_j}X$，其中$N_j$为第$j$个聚类域$S_j$包含的个数；

（4）若$z'_j\neq z_j\ (j=1,2,\cdots,K)$，转（2）；否则算法收敛，计算结束。

4.4 解聚类问题的模拟退火算法

其模拟退火算法的步骤为：

（1）给定起、止“温度”$T=100000$、$T_0=1$和退火速度$\alpha=0.9$，随机产生一个聚类方案$Y_0=(y_{ij})_{n\times K}$，计算每个模式样本到聚类中心的距离f_0；

（2）若$T>T_0$，转（3），否则算法停止，输出Y_0；

（3）随机产生模式样本i和随机产生聚类中心j，令$y_{ij}=1$，其它$y_{im}=0(m=1,2,\cdots,K,m\neq j)$，此时变量记为$Y_1$；

（4）计算每个模式样本到新的聚类中心的距离之和f_1，$\Delta E=f_1-f_0$，若$\Delta E\leq 0$，接受新值，$Y_0\leftarrow Y_1$，$T\leftarrow\alpha T$，转（2）；否则若$\exp(-\Delta E/T)>rand(0,1)$，也接受新值，$Y_0\leftarrow Y_1$，$T\leftarrow\alpha T$，转（2）；否则转（3）；

4.5 解聚类问题的蚁群算法及数值分析

4.5.1 解聚类问题的蚁群算法

聚类问题的蚁群算法思路如下：模式样本分配给第j个聚类中心$z_j\ (j=1,2,\cdots,K)$，蚂蚁就在模式样本i到聚类中心z_j的路径上留下外激素τ_{ij}，第i个蚂蚁选择聚类中心z_j概率为：

$$p_{ij}=\frac{\tau_{ij}}{\sum_{j=1}^{n}\tau_{ij}} \tag{4.3}$$

更新方程为：

$$\tau_{ij}^{new}=\rho\tau_{ij}^{old}+\frac{Q}{d_{ij}} \tag{4.4}$$

d_{ij}为模式样本i到聚类中心z_j的距离，ρ表示强度的持久性系数，一般取

0.5～0.99左右，Q为一正常数。

解聚类问题的蚁群算法如下：

（1）$nc \leftarrow 0$（nc为循环次数），给$\tau_{ij}\ (i,j=1,2,\cdots,n)$赋相同的数值，给出蚂蚁数$m$、$Q$、$\rho$的值，随机给出一个分配方案；

（2）对每个蚂蚁按转移概率p_{ij}选择下一个节点；

（3）计算新的聚类中心，计算每个模式样本到新的聚类中心的距离d_{ij}，按更新方程修信息强度；

（4）$nc \leftarrow nc+1$，若$nc>$规定的次数NC，停止运行，根据外激素输出最好的解；否则转（2）；

4.5.2 数值分析

例如对于14个2维样本[61]蝶形数据，最优聚类结果如图4.1所示。采用K-均值算法，此方法虽然简单，但其结果与初始聚类中心有关，假如初始聚类中心取14个样本中的任意两个，共有$C_{14}^{2}=91$种情况，经过验证有12种情况：$\{x_1,x_2\}$，$\{x_1,x_3\}$，$\{x_2,x_3\}$，$\{x_4,x_5\}$，$\{x_4,x_6\}$，$\{x_5,x_6\}$，$\{x_9,x_{10}\}$，$\{x_9,x_{11}\}$，$\{x_{10},x_{11}\}$，$\{x_{12},x_{13}\}$，$\{x_{12},x_{14}\}$，$\{x_{13},x_{14}\}$收敛不到最优聚类结果。如初始聚类中心取$\{x_1,x_2\}$时，聚类结果如图4.2所示。因此对K-均值算法改进的方法是，随机产生一个聚类方案，计算其聚类中心作初试聚类中心。其效果如表4.1所示。

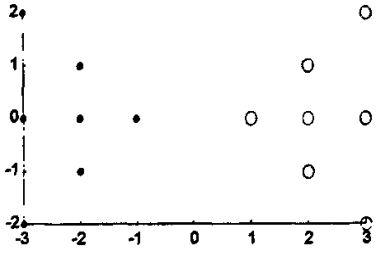

图4.1 最优聚类结果

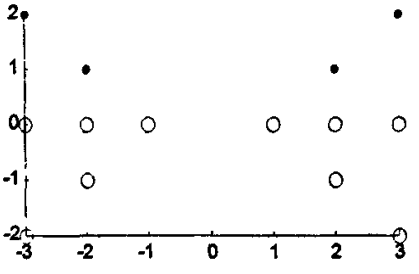

图4.2 初始聚类中心取$\{x_1,x_2\}$时K-均值算法聚类结果

采用模拟退火算法，参数设置如下：起始温度$T=1000$，终止温度$T_0=1$，退火速度$\alpha=0.95$。模拟退火算法的结果与初始聚类中心关系不大，但由于固体退火必须缓慢降温，才能使固体在每一温度下都达到热平衡，最终趋于平衡状态，因此，控制参数的值必须缓慢衰减，才能确保模拟退火算法最终趋于优化问题的整体最优解，因此迭代次数一般较长，且其正确率也不高。

采用蚁群算法参数设置如下：$\rho=0.9$，$Q=100$，$NC=100$。蚁群算法的结果与初始聚类中心关系也不大。与模拟退火算法类似，也采用概率改变变量，模拟退火算法在解的附近随机地找下一个解，以随机概率方式选取下一个解，而蚁群算法考虑到了附近解“外激素”，好的解附近的“外激素”较多，它被选取的概率就大，蚁群算法的迭代次数一般比较少，其方法相对比较有效。聚类问题是一个线性整数规划问题，特别当模式样本和分类数很大时，用蚁群算法解决多聚类问题确实有效。对于“分类数目未知”的情况，如何用蚁群算法来解，可以作进一步的研究。

表 4.1 蝶形数据各种算法测试结果（后 3 种算法各运行 100 次）

算法	正确率
K-均值算法（以任意两个样本作聚类中心）	79/91=86.8%
K-均值算法（随机产生一个聚类方案）	90%
模拟退火算法	78%
蚁群算法	92%

4.6 解聚类问题的与K-均值算法混合的蚁群算法及数值分析

4.6.1 解聚类问题的K-均值算法混合的蚁群算法

从实验可知，K-均值算法收敛速度比 ACO 快，但是其结果与初始聚类中心有关。而 ACO 较精确，但速度比较慢，需要对其改进。蚁群算法初始化时，各路径的信息素取相同值，让蚂蚁以等概率选择路径，这样使蚂蚁很难在短时间内从大量的杂乱无章的路径中，找出一条较好的路径，所以收敛速度较慢。假如初始化时就给出启发性的信息量，可以加快收敛速度。改进思路是与K-均值算法混合，具体方法是先用K-均值算法作快速分类，根据分类结果更新信息素，指导其它蚂蚁选择，其他步骤与同上。

根据分类结果更新信息素，有两种方法：

其一：根据分类结果，计算出聚类中心z_j（$j=1,2,\cdots,K$），若模式样本i分配到聚类中心z_j，计算模式样本i到聚类中心z_j的距离d_{ij}，则$\Delta\tau_{ij}=\frac{Q}{d_{ij}}$，其它路径不变。

其二：根据分类结果，计算出聚类中心z_j（$j=1,2,\cdots,K$），计算模式样本i

（$i=1,2,\cdots,n$）到聚类中心 z_j（$j=1,2,\cdots,K$）的距离 d_{ij}，所有路径的信息增量 $\Delta\tau_{ij}=\frac{Q}{d_{ij}}$。

4.6.2 数值分析

这里采用著名的 Iris[60]植物样本数据进行测试。Iris 植物样本数据分别属于 3 种植物的 150 个样本，每个样本均为代表植物 4 种特征的 4 维向量。K-均值算法初始聚类中心取 150 个样本中的任意 3 个，共有 $C_{150}^{3}=551300$ 种情况，每种情况运行一次（耗时接近 5 小时），只有 221234 次达到最优解。蚁群算法参数：$\rho=0.99$，$Q=80$，$NC=100$，蚂蚁数 $m=30$，各种算法各测试 50 次，结果如表 4.2 所示。从表 4.2 可以看出，其中蚁群算法中的 3 种算法都比 K-均值算法效果好，特别后两种算法效果更好。由于 K-均值算法有可能结果不太好，对于方法一，对后续的蚁群算法产生误导，而改进方法二中，信息增量是根据距离大小决定，会减少影响，因此效果比改进方法一好。

表4.2 Iris植物样本数据的结果比较

算法	平均值	最好解	最差解	最好解的次数	正确率
模拟退火算法	113.3637	78.6510	562.3526	5	10%
K-均值算法(551300 次)	91.9417	78.6510	682.8385	221234	40.13%
基本蚁群算法	90.0897	78.6510	143.2725	21	42%
改进方法一	80.3594	78.6510	113.1760	24	48%
改进方法二	80.1692	78.6510	110.9711	32	64%

4.7 本章小结

本章提出了两种求解聚类问题的蚁群算法。一种方法是把聚类问题变换成蚂蚁寻食过程，蚂蚁在模式样本到聚类中心的路径上留下外激素，引导其他蚂蚁选择；另一种是与 K-均值算法混合，用 K-均值算法作快速分类，根据分类结果更新信息素，指导其它蚂蚁选择。与其它算法比较，测试数据显示与 K-均值算法混合的算法效果相当好。

5 与模拟退火算法混合

5.1 引言

单纯依赖一种算法求解复杂问题，有时效果并不好，与其他算法进行混合，是一个很好的思路，本章讨论与模拟退火算法混合来解决圆排列问题，第 6 章、第 7 章和第 8 章分别讨论与遗传算法混合、与混沌理论混合和与粒子群混合算法来解决旅行商问题。与模拟退火算法混合有两种思路，一种是在模拟退火算法中运用蚁群算法思想找邻域的解，称为蚁群模拟退火算法；另一种是采用模拟退火算法生成信息素分布，蚂蚁算法寻优中采用模拟退火的在邻域内找另外一个解的策略，称为模拟退火蚁群算法。下面讨论用蚁群模拟退火算法解圆排列问题和用模拟退火蚁群算法解旅行商问题。

5.2 解圆排列问题的蚁群模拟退火算法

5.2.1 圆排列问题及与旅行商问题等价

实际工程中经常涉及到工件切割问题，如把一个矩形钢板切割成半径不等的圆，尽可能节省材料，这就是圆排列问题。所谓圆排列问题[39]是指给定 n 个大小不等的园 $c_1,c_2,\cdots,c_n$，现要将这 n 个圆排进一个矩形框中，且要求与矩形的底边相切。圆排列问题要求从 n 个圆的所有排列中找出有最小长度的圆排列。文献[39]用回溯法解决此问题，其最坏情况接近于枚举法，时间复杂性为 $O((n+1)!)$，属于 $NP-$完全问题，目前没有有效的算法解此问题。圆排列问题有很强的应用背景，如铺设半径大小不等的电缆管道、下水道等等，都可以转化为圆排列问题。

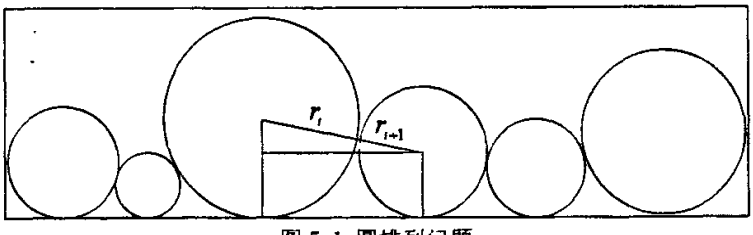

图 5.1 圆排列问题

已知圆 c_i 的半径 $r_i,(i=1,2,\cdots,n)$，假如排列方式为 $i_1,i_2,\cdots,i_n$，则长度为

$$D=r_{i_1}+2\sqrt{r_{i_1}r_{i_2}}+2\sqrt{r_{i_2}r_{i_3}}+\cdots+2\sqrt{r_{i_{n-1}}r_{i_n}}+r_{i_n} \tag{5.1}$$

圆排列问题要求所有排列中使长度找出最小。所有的排列有 $n!$ 个，去掉对称的排列，如 $1,2,\cdots,n-1,n$ 与 $n,n-1,\cdots,2,1$ 对称，共有 $\frac{1}{2}n!$ 种排列。

旅行商问题（TSP）是指有 n 个城市，城市 i，j 之间的距离为 d_{ij}，一个旅行商从城市 1 出发到其他每个城市去一次且只去一次，最后回到城市 1，旅行商问题要求从

2~n个城市的所有排列中找出总路线最短的路线。

假设把1~n个圆分别放置在1~n个城市中，城市i，j之间的距离d_{ij} $(i=1,2,\cdots,n; j=1,2,\cdots,n)$为

$$d_{ij}=2\sqrt{r_i r_j} \tag{5.2}$$

再增加一个城市0，它与城市j的距离d_{0j} $(j=1,2,\cdots,n)$为

$$d_{0j}=r_j \tag{5.3}$$

因此圆排列问题与旅行商问题等价，一个旅行商从城市0出发到其他每个城市去一次且只去一次，最后回到城市0，旅行商问题要求从1~n个城市的所有排列中找出总路线最短的路线。所以解圆排列问题就变为解旅行商问题。

旅行商问题也是一个NP-完全问题，目前求解旅行商问题的主要方法有动态规划方法[39]，分枝限界法[39]，模拟退火算法[45,2,53]，遗传算法[41,43,54]，启发式搜索法，神经网络算法[55]，蚁群算法等。各种算法各有千秋，模拟退火算法最早思想由Metropolis在1953年提出，1983年Kirkpatrick等成功地将退火思想引入组合优化领域。模拟退火算法是局部搜索算法的扩展，理论上来说，它是一个全局最优算法。如何在初始解附近找出一个“好的解”是一项关键技术，它直接影响算法的收敛速度。本文推荐采用蚁群算法思想的模拟退火算法来解决此问题，其主要思想利用点的邻接关系，距离近的邻点以较大的概率选为下一个访问点。

5.2.2 解旅行商问题的模拟退火算法

模拟退火算法要从邻域中随机产生另一个解，对于旅行商问题，它的邻域是指两条路径除局部有差别外，大多数路径相同。

解TSP问题的模拟退火算法的框架[63]：

给定起、止“温度”T、T_0和退火速度α，初始一条路径C_0；

While $(T>T_0)$ do

在C_0的邻域内产生另一条路径C_1；

计算两条路径所引起的目标函数（能量）值的变化ΔE；

若$\Delta E\le 0$，接受新值，否则若$\exp(-\Delta E/T)>rand(0,1)$（$rand(0,1)$表示0~1之间的随机数），也接受新值，否则就拒绝；

确定新的参数值，若扰动被接受，则$C_0\leftarrow C_1$，否则C_0不变；

若接受新值，降温$T \leftarrow \alpha T$，否则不降温；
End

5.2.3 几种算法的比较

模拟退火算法是依赖邻域结构的迭代方法，如何找领域的解直接影响收敛速度和最优解。按照上面方法把圆排列问题转化为旅行商问题。针对这个问题，在上面算法中，在C_0的邻域内产生另一条路径C_1，这里提出6种算法。为便于说明，假设本问题为$n=9$的圆排列问题。

算法 A 在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，在路径C_0中交换第j_1次和第j_2次访问的城市，其余不变，此时的路径为C_1。

比如$C_0=0\ \ 2\ \ 3\ \ 4\ \ 1\ \ 5\ \ 7\ \ 9\ \ 8\ \ 6$，$j_1=2$（第2次访问的城市是城市3），$j_2=7$（第7次访问的城市是城市9），则$C_1=0\ \ 2\ \ 9\ \ 4\ \ 1\ \ 5\ \ 7\ \ 3\ \ 8\ \ 6$。

算法 B 在第$1\sim n$个访问的城市中随机地选取第j_1次访问的城市，在路径C_0中交换第j_1次和第j_1+1次访问的城市，其余不变，此时的路径为C_1。

比如$C_0=0\ \ 2\ \ 3\ \ 4\ \ 1\ \ 5\ \ 7\ \ 9\ \ 8\ \ 6$，$j_1=2$，则

$C_1=0\ \ 2\ \ 4\ \ 3\ \ 1\ \ 5\ \ 7\ \ 9\ \ 8\ \ 6$。

算法 C 也称逆转算法，在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，在路径C_0中第j_1次到第j_2次访问的城市之间的子路径以反方向插入，其余不变，此时的路径为C_1。

比如$C_0=0\ \ 2\ \ 3\ \ 4\ \ 1\ \ 5\ \ 7\ \ 9\ \ 8\ \ 6$，$j_1=2$，$j_2=7$，则

$C_1=0\ \ 2\ \ 9\ \ 7\ \ 5\ \ 1\ \ 4\ \ 3\ \ 8\ \ 6$。

算法 D 在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，假设$j_1<j_2$，在路径C_0中将第j_1次访问的城市安排到第j_2次访问的城市之后，其余不变，此时的路径为C_1。

比如$C_0=0\ \ 2\ \ 3\ \ 4\ \ 1\ \ 5\ \ 7\ \ 9\ \ 8\ \ 6$，$j_1=2$，$j_2=7$，则

$C_1 = 0\ \ 2\ \ 4\ \ 1\ \ 5\ \ 7\ \ 9\ \ 3\ \ 8\ \ 6$。

蚁群模拟退火算法 I　上面算法没有利用城市间距离大小的信息，蚁群模拟退火算法 I 将利用点的邻接关系，依据蚁群算法的思想，距离近的邻接点以较大的概率被选为下一个访问点，所以在局部调整时依据此思想。

设 $d(i,j)$ 表示城市 i 与城市 j 的距离，在 $1\sim n$ 的城市中随机地选取城市 i_1，离城市 i_1 最远的城市的距离为：$d_{\max} = \max_j d(i_1,j)$，为了排除下一个访问点为它自己，令 $d(i_1,i_1) = d_{\max}$，则下一个访问点为城市 j 的概率为：

$$p_j = \frac{d_{\max} - d(i_1,j)}{\sum_{k=1}^{n}(d_{\max} - d(i_1,k))} \tag{5.4}$$

假设以（5.4）式的概率选取的是城市 j_1，在路径 C_0 中将城市 j_1 安排到城市 i_1 之后，其余不变，此时的路径为 C_1。

比如 $C_0 = 0\ \ 2\ \ 3\ \ 4\ \ 1\ \ 5\ \ 7\ \ 9\ \ 8\ \ 6$，$i_1 = 3$，$j_1 = 7$，则

$C_1 = 0\ \ 2\ \ 3\ \ 7\ \ 4\ \ 1\ \ 5\ \ 9\ \ 8\ \ 6$。

蚁群模拟退火算法 II　在蚁群模拟退火算法 I 中是在 $1\sim n$ 的城市中随机地选取城市 i_1，为了使路径总长度之和达到最小，优先解决薄弱环节，这里采用路径中相邻城市之间的距离大的两个城市以较大的概率被选取，在它们之间插入其他城市。用 $l(n)$ 数组记录路径 C_0 相邻城市之间的距离，具体数据如下：

$l(k) = d[c(k),c(k+1)]$，$k = 1,2,\cdots,n-1$

$l(n) = d[c(n),c(1)]$

选取城市 i 的概率为：

$$p_i = \frac{l(i)}{\sum_{k=1}^{n} l(k)} \tag{5.5}$$

按（5.5）式以概率选取的是城市 i_1，后面的步骤同蚁群模拟退火算法 I，按（5.4）式以概率选取的是城市 j_1，在路径 C_0 中将城市 j_1 安排到城市 i_1 之后，其余不变，此时

的路径为C_1。

5.2.4 算例分析

假设圆排列问题n取30, 50, 100，$r_i = i$（$i = 1,2,\cdots,n$）。假设算法的参数相同，起始温度$T = 100000$，终止温度$T_0 = 1$，退火速度$\alpha = 0.99$，各种算法各随机测试100次，结果如表5.1-5.3所示。

表5.1 $n = 30$的结果比较

算法	结果比较			
	平均时间/s	平均值	最好解	最差解
算法A	0.29	768.3933	758.8744	787.3225
算法B	0.28	775.9391	764.7513	789.9046
算法C	0.35	766.1576	756.4350	778.0075
算法D	0.27	767.8782	756.8798	782.2558
蚁群模拟退火算法I	0.49	767.7527	755.1965	776.5480
蚁群模拟退火算法II	0.67	765.7780	750.7518	770.4598

表5.2 $n = 50$的结果比较

算法	结果比较			
	平均时间/s	平均值	最好解	最差解
算法A	0.63	2066.8	2049.1	2071.5
算法B	0.58	2070.7	2064.2	2099.2
算法C	0.71	2067.6	2050.5	2086.5
算法D	0.58	2075.3	2044.6	2087.4
蚁群模拟退火算法I	0.78	2047.4	2041.7	2054.3
蚁群模拟退火算法II	0.98	2041.5	2037.5	2045.5

表5.3 $n = 100$的结果比较

算法	结果比较			
	平均时间/s	平均值	最好解	最差解
算法A	0.71	8107.3	8099.4	8116.4
算法B	0.69	8124.7	8105.2	8149.2
算法C	0.74	8086.6	8073.4	8099.7
算法D	0.67	8097.3	8096.7	8116.3
蚁群模拟退火算法I	0.82	8057.5	8042.3	8072.5
蚁群模拟退火算法II	1.05	8023.1	8019.8	8049.7

从表5.1-5.3可以看出，蚁群模拟退火算法I与蚁群模拟退火算法II都是采用蚁群算法思想的模拟退火算法，蚁群模拟退火算法II是最好的算法，其次是蚁群模拟退火算法I和算法C，算法C采用逆转策略，可以使迭代过程突破局部最优圈而跳到另一个搜索空间。算法A、算法B和算法D花费的时间少，说明它们很容易落入局

部最优解，效果较差。随着规模n的增大，运行时间都增大。特别当n增大时,采用蚁群模拟退火算法 II 的效果更好,优点更显著。本章把圆排列问题转化为 TSP 问题，TSP 问题是组合优化领域中的一个典型的问题，解决此问题有较大的现实意义。对于其他类似问题，如印刷电路板的钻空路线方案，连锁店送货路线问题等都可简化为 TSP 问题。本节讨论了 6 种算法，推荐采用蚁群模拟退火算法 II。

5.3 解旅行商问题的模拟退火蚁群算法

5.3.1 混合的基本思想

模拟退火算法用于优化问题的出发点是基于物理中固体物质的退火过程与一般优化问题的相似性。它具有大范围快速全局搜索能力，但对系统中的反馈信息利用不够，当求解到一定范围时往往做大量无为的冗余迭代，求精确解效率低。蚁群算法原理是一种正反馈机制，但初期信息素匮乏，求解速度慢。本文将是将模拟退火算法与蚁群算法的混合来解决旅行商问题，采用模拟退火算法生成信息素分布,利用蚁群算法求精确解，优势互补，期望获得优化性能和时间性能的双赢。在蚂蚁算法寻优中，采用模拟退火的在邻域内找另外一个解的策略，改善解的质量。

混合的思路是首先由模拟退火算法产生较优解，较优的路径留下信息素，其他不改变；然后让蚂蚁按照蚁群算法，完成一次遍历后，采用模拟退火的在邻域内找另外一个解，有可能邻域内找另外一个解不一定得到改善，接受准则采用模拟退火算法的思想，允许目标函数有限范围内变坏，为简化计算量并不按概率取舍，若路径长度差$\Delta E < e$接受，e为按允许目标函数变坏范围。

另外这里作如下改进，蚂蚁每次周游结束后，不论蚂蚁搜索到的解如何，都将赋予相应的信息增量，比较差的解也将留下信息素，这样就干扰后续的蚂蚁进行寻优，造成大量的无效的搜索。改进的方法是，只有比较好的解才留下信息素，即只有当路径长度小于给定的值才留下信息素。为了充分利用各蚂蚁所走过的路径信息，随时记录当前的最好解。也采用MAX-MIN Ant System技术[9]，即路径上的信息素浓度被限制在$[\tau_{min},\tau_{max}]$范围内。

5.3.2 找邻域解策略

模拟退火算法要从邻域中随机产生另一个解，对于旅行商问题，它的邻域是指两条路径除局部有差别外，大多数路径相同。由路径C_0邻域内产生另一条路径C_1，常用的有以下 4 种策略，这里假设有n个城市，与 5.2.3 节类似。

策略 A 在第1～n个访问的城市中随机地选取第j_1次和第j_2次访问的城市，在路径C_0中交换第j_1次和第j_2次访问的城市，其余不变，此时的路径为C_1。

策略 B 在第$1\sim n$个访问的城市中随机地选取第j_1次访问的城市，在路径C_0中交换第j_1次和第j_1+1次访问的城市，其余不变，此时的路径为C_1。

策略 C 在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，假设$j_1<j_2$，在路径C_0中将第j_1次访问的城市安排到第j_2次访问的城市之后，其余不变，此时的路径为C_1。

策略 D 也称逆转策略，在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，在路径C_0中第j_1次到第j_2次访问的城市之间的子路径以反方向插入，其余不变，此时的路径为C_1。

5.3.3 模拟退火蚁群算法

解 TSP 问题的模拟退火蚁群算法的算法如下：

步骤 1 利用模拟退火算法产生一个较优解，在这个路径留下信息素；

步骤 2 $nc\leftarrow 0$，（nc为迭代步数或搜索次数），将m个蚂蚁置于n个顶点上；

步骤 3 将各蚂蚁的初始出发点置于当前解集中，对每个蚂蚁$k(k=1,2,\cdots,m)$，按概率p_{ij}^k移至下一顶点j，将顶点j置于当前解集，完成一次遍历；

步骤 4 在邻域内找另外一个解，若路径长度差$\Delta E<e$接受新值，否则就拒绝；

步骤 5 计算各蚂蚁的路径长度L_k $(k=1,2,\cdots,m)$，记录当前的最好解；

步骤 6 对路径长度L_k小于给定值的路径，按更新方程（2）修改轨迹强度；

步骤 7 $nc\leftarrow nc+1$；

步骤 8 若$nc<$预定的迭代次数且无退化行为（即找到的都是相同解）则转步骤 2；

步骤 9 输出目前最好解。

5.3.4 算法测试

为了检验算法的有效性，分别选用 Oliver30（最好解为 423.7406）和 att48（TSPLIB 提供的最好解为 33522）作为实验例子来研究。并于基本蚁群算法、随机初始化蚁群算法（产生大量的路径如 100，从中选择比较优的如 30 条，使这些路径留下信息素）、基本遗传算法、模拟退火算法进行比较。模拟退火算法采用文献[42]的算法，起始温度$T=100000$，终止温度$T_0=1$，退火速度$\alpha=0.99$；遗传算法程序采

用 MATLAB 的遗传算法工具箱[56]，参数如下：染色体个数 $N=30$，交叉概率 $P_c=0.2$，变异概率 $P_m=0.5$，迭代次数 100；混合算法参数：$\alpha=1.5$，$m=30$，$\beta=2$，$\rho=0.9$，混合算法采用 4 种方法进行比较。对各种算法测试 20 次，结果如表 5.4 所示。图 2.5.1(第 2.5 节)是混合算法最好的解，总路程为 423.7406。2.5.2(第 2.5 节)是解 att48 混合算法最好的解，总路程为 33522。从表 1 可以看出，混合的 4 种算法大都比较好，策略 D（逆转策略）比其他策略效果好，策略 D 采用逆转策略，可以跳出局部极值，避免出现比较差的解。利用蚁群算法和模拟退火算法特性，提出的混合算法可以显著提高计算效率，具有较大的实用价值。

表 5.4 几种算法测试结果

算法	Oliver30			att48		
	平均值	最好解	最差解	平均值	最好解	最差解
模拟退火算法	438.5223	424.6918	479.8312	34958	35176	40536
遗传算法	483.4572	467.6844	502.5742	38541	38732	42458
基本蚁群算法	450.0346	441.9581	499.9331	35876	36532	42234
随机初始化蚁群算法	423.7406	429.7032	432.4568	33780	35533	36534
策略 A	438.4350	424.6354	457.9062	34991	35175	38684
策略 B	435.4220	424.2003	447.3223	34711	35197	37790
策略 C	438.4777	424.1972	465.9935	34035	35305	39367
策略 D	431.4987	423.7406	447.6865	33798	33522	36821

5.4 本章小结

提出了与模拟退火算法混合的两种算法。一种是在模拟退火算法中运用蚁群算法思想找邻域的解，称为蚁群模拟退火算法，并用该算法较好地解决圆排列问题；另一种是采用模拟退火算法生成信息素分布，蚂蚁算法寻优中采用模拟退火的在邻域内找另外一个解的策略，称为模拟退火蚁群算，并用该算法较好地解决了旅行商问题。若把这几种算法混合在一起，或与遗传算法的思想融在一起，可能效果会更好，因此还有许多工作有待研究。

6 与遗传算法混合

6.1 引言

遗传算法是由美国的John Holland教授于1975年首先提出的一类仿生型优化算法。它是以达尔文的生物进化论“适者生存、优胜劣汰”和孟德尔的遗传变异理论为基础，模拟生物界进化过程。它具有大范围快速全局搜索能力，但对系统中的反馈信息利用不够，当求解到一定范围时往往做大量无为的冗余迭代，求精确解效率低。蚁群算法原理是一种正反馈机制，但初期信息素匮乏，求解速度慢。文[57][58]将是将遗传算法与蚂蚁算法的融合，采用遗传算法生成信息素分布，利用蚂蚁算法求精确解，优势互补，期望获得优化性能和时间性能的双赢。本节文将遗传算法与蚁群算法的混合来解决旅行商问题，不仅仅采用遗传算法生成初始信息素分布，在蚂蚁算法寻优中，采用交叉和变异的策略，改善解的质量。

6.2 基本遗传算法

遗传算法[43]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。遗传算法基本步骤：

(1) 初始化群体；

(2) 计算群体上每个个体的适应度值；

(3) 按由个体适应度值所决定的某个规则选择将进入下一代的个体；

(4) 按概率 P_c 进行交叉操作；

(5) 按概率 P_m 进行突变操作；

(6) 没有满足某种停止条件，则转第(2)步，否则进入(7)。

(7) 输出种群中适应度值最优的染色体作为问题的满意解或最优解。

6.3 蚁群算法与遗传算法的混合

6.3.1 混合的基本思想

混合的思路是首先由遗传算法产生较优解，较优的路径留下信息素，其他不改变；然后让蚂蚁按照蚁群算法，完成一次遍历后，再让蚂蚁作遗传算法的交叉操作和变异操作，有可能经过交叉操作和变异操作的解不一定得到改善，只有改善的蚂蚁路径，

才代替原来的路径。

另外这里作如下改进，蚂蚁每次周游结束后，不论蚂蚁搜索到的解如何，都将赋予相应的信息增量，比较差的解也将留下信息素，这样就干扰后续的蚂蚁进行寻优，造成大量的无效的搜索。改进的方法是，只有比较好的解才留下信息素，即只有当路径长度小于给定的值才留下信息素。为了充分利用各蚂蚁所走过的路径信息，随时记录当前的最好解。也采用 MAX-MIN Ant System 技术[9]，即路径上的信息素浓度被限制在$[\tau_{min},\tau_{max}]$范围内。

6.3.2 变异操作

由路径C_0变异到另一条路径C_1，常用的有以下几种策略，与 5.2.3 节类似，局部有点区别，这里假设有n个城市。

变异策略 A 在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，在路径C_0中交换第j_1次和第j_2次访问的城市，其余不变，此时的路径为C_1。比如C_0=2 3 4 1 5 7 9 8 6，$j_1=2$（第 2 次访问的城市是城市 3），$j_2=7$（第 7 次访问的城市是城市 9），则C_1=2 9 4 1 5 7 3 8 6。

变异策略 B 在第$1\sim n$个访问的城市中随机地选取第j_1次访问的城市，在路径C_0中交换第j_1次和第j_1-1次访问的城市，其余不变，此时的路径为C_1。比如C_0=2 3 4 1 5 7 9 8 6，$j_1=3$，则C_1=2 4 3 1 5 7 9 8 6。

变异策略 C 也称逆转策略，在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，在路径C_0中第j_1次到第j_2次访问的城市之间的子路径以反方向插入，其余不变，此时的路径为C_1。比如C_0=2 3 4 1 5 7 9 8 6，$j_1=2$，$j_2=7$，则C_1=2 9 4 7 5 1 3 8 6。

变异策略 D 在第$1\sim n$个访问的城市中随机地选取第j_1次和第j_2次访问的城市，假设$j_1<j_2$，在路径C_0中将第j_1次访问的城市安排到第j_2次访问的城市之前，其余不变，此时的路径为C_1。比如C_0=2 3 4 1 5 7 9 8 6，$j_1=2$，$j_2=7$，则C_1=2 4 1 5 7 3 9 8 6。

6.3.3 交叉操作

交叉的方法很多，下面几种方法最常用：

交叉策略 A 在第二个串中随机选择一个交叉区域；将 old2 的交叉区域加到 old1 前面（或后面），删除 old1 中已在 old2 的交叉区中出现过的城市。例如两父串为：old1=1 2 3 4 5 6 7 8 9，old2=9 8 7 |6 5 4 3| 2 1，若交叉区域为：6 5 4 3，交叉后为：new1=6 5 4 3 1 2 7 8 9。

交叉策略 B 在第二个串中随机选择一个交叉区域；将 old2 的交叉区域加到 old1 对应的位置，删除 old1 中已在 old2 的交叉区中出现过的城市。例如两父串为：old1=1 2 3 4 5 6 7 8 9，old2=9 8 7 |6 5 4 3| 2 1，若交叉区域为：6 5 4 3，交叉后为：new1=1 2 6 5 4 3 7 8 9。

交叉策略 C 在第二个串中随机选择一个交叉区域；将 old2 的交叉区域加到 old1 的随机的位置，删除 old1 中已在 old2 的交叉区中出现过的城市。例如两父串为：old1=1 2 3 4 5 6 7 8 9，old2=9 8 7 |6 5 4 3| 2 1，随机产生城市 7，则加入到城市 7 后，若交叉区域为：6 5 4 3，交叉后为：new1=1 2 7 6 5 4 3 8 9。

交叉策略 D 在第二个串中随机选择一个交叉区域，如交叉区域为：6 5 4 3；将 old2 的交叉区域加到 old1 的城市 6 的位置，删除 old1 中已在 old2 的交叉区中出现过的城市。例如两父串为：old1=1 2 3 4 5 6 7 8 9，old2=9 8 7 |6 5 4 3| 2 1，交叉后为：new1=1 2 6 5 4 3 7 8 9。

6.3.4 遗传蚁群算法

解 TSP 问题的遗传蚁群算法如下：

步骤 1 利用遗传产生一个较优解，在这个路径留下信息素；

步骤 2 $nc \leftarrow 0$，（nc 为迭代步数或搜索次数），将 m 个蚂蚁置于 n 个顶点上；

步骤 3 将各蚂蚁的初始出发点置于当前解集中，对每个蚂蚁 $k(k=1,2,\cdots,m)$，按概率 p_{ij}^k 移至下一顶点 j，将顶点 j 置于当前解集，完成一次遍历；

步骤 4 根据交叉概率，选择若干组解，然后分组进行交叉的解，若新的目标函数变好，接受新值，否则就拒绝；

步骤 5 根据变异概率，判断是否变异，变异后的目标函数变好，接受新值，否则就拒绝；

步骤 6 计算各蚂蚁的路径长度 L_k $(k=1,2,\cdots,m)$，记录当前的最好解；

步骤 7 对路径长度 L_k 小于给定值的路径，按更新方程（2）修改轨迹强度；

步骤 8 $nc \leftarrow nc+1$；

步骤 9 若 $nc <$ 预定的迭代次数且无退化行为（即找到的都是相同解）则转步骤 2；

步骤 10 输出目前最好解。

6.4 算法测试

为了检验算法的有效性，分别选用 Oliver30（最好解为 423.7406）和 att48（TSPLIB 提供的最好解为 33522）作为实验例子来研究。并于基本蚁群算法、随机初始化蚁群算法（产生大量的路径如 100，从中选择比较优的如 30 条，使这些路径留下信息素）、基本遗传算法、模拟退火算法进行比较。

模拟退火算法采用文献[42]的算法，起始温度$T=100000$，终止温度$T_0=1$，退火速度$\alpha=0.99$；遗传算法程序采用 MATLAB 的遗传算法工具箱[56]，参数如下：染色体个数$N=30$，交叉概率$P_c=0.2$，变异概率$P_m=0.5$，迭代次数 100；混合算法参数：$\alpha=1.5$，$m=30$，$\beta=2$，$\rho=0.9$，混合算法分别采用 4 种交叉策略和 4 种变异策略组合起来 16 种方法进行比较。对各种算法测试 20 次，结果如表 6.1 所示。图 2.5.1(第 2.5 节)是混合算法最好的解，总路程为 423.7406。2.5.2(第 2.5 节)是解 att48 混合算法最好的解，总路程为 33522。

从表 6.1 可以看出，混合的 16 种组合大都比较好，特别交叉策略 B 和变异策略 B 的混合算法是最好的算法。交叉策略 B 比其他交叉策略效果好，因为其他交叉策略将交叉区域加入到随机的位置，有可能增加了部分差的解。在变异策略中，变异策略 B 效果比较好，其变异的幅度比较小，避免出现比较差的解。

表 6.1 几种算法测试结果

算法			Oliver30			att48		
			平均值	最好解	最差解	平均值	最好解	最差解
模拟退火算法			438.5223	424.6918	479.8312	34958	35176	40536
遗传算法			483.4572	467.6844	502.5742	38541	38732	42458
基本蚁群算法			450.0346	441.9581	499.9331	35876	36532	42234
随机初始化蚁群算法			423.7406	429.7032	432.4568	33780	35533	36534
混合算法	交叉策略 A	变异策略 A	439.4948	425.6490	456.7721	35036	35259	38588
		变异策略 B	441.9257	428.7296	455.2382	35230	35514	38459
		变异策略 C	437.0028	426.6002	446.2394	34837	35338	37698
		变异策略 D	438.7750	425.4752	455.2929	34979	35245	38463
	交叉策略 B	变异策略 A	438.9350	424.6354	457.9062	34991	35175	38684
		变异策略 B	431.4987	423.7406	447.6865	33798	33522	36821
		变异策略 C	435.4220	424.9003	447.3223	34711	35197	37790
		变异策略 D	439.4777	426.1972	465.9935	35035	35305	39367
	交叉策略 C	变异策略 A	444.1723	429.3803	459.4925	35409	35568	38818
		变异策略 B	438.5871	426.3076	455.5854	34964	35314	38488
		变异策略 C	440.4201	427.6016	454.8674	35110	35421	38427
		变异策略 D	439.5524	424.4643	461.7948	35040	35161	39012
	交叉策略 D	变异策略 A	439.0477	424.6727	451.8001	35000	35178	38168
		变异策略 B	436.0081	423.7406	460.6230	34758	35101	38913
		变异策略 C	438.8091	425.8201	455.4830	34981	35273	38479
		变异策略 D	436.4577	423.9490	457.3155	34794	35118	38634

6.5 本章小结

利用蚁群算法和遗传算法特性，提出的混合算法吸收了两种算法的优点，可以显著提高计算效率，具有较大的实用价值。本章用混合遗传蚁群算法解决了离散性优化问题，对于连续性优化问题有待于进一步研究。

7 与混沌理论混合

7.1 引言

混沌是自然界广泛存在的一种非线性现象，它看似混沌，却有着精致的内在结构，具有“随机性”、“遍历性”及“规律性”等特点，对初始条件极度敏感，能在一定范围内按其自身规律不重复地遍历所有状态，利用混沌运动的这些性质可以进行优化搜索[59,60]。根据混沌特性，融入到其它算法中，提出了一系列新的优化方法，如混沌遗传算法[61]。本文将混沌融入蚁群算法中，提出混沌蚁群（Chaos Ant Colony Optimization，简称 CACO）算法，利用混沌初始化进行改善个体质量和利用混沌扰动避免搜索过程陷入局部极值。解旅行商问题的仿真结果表明，CACO 算法较与其他算法比较，效果比较好。

7.2 混沌及运动特性

目前对混沌尚无严格的定义，一般将由确定性方程得到的具有随机性的运动状态称为混沌。Logistic 映射就是一个典型的混沌系统，迭代公式如下：

$$z_{i+1}=\mu z_i(1-z_i),\quad i=0,1,2,\cdots,\quad \mu\in(2,4] \tag{7.1}$$

式中 μ 为控制参量，当 $\mu=4$，$0\le z_0\le 1$，Logistic 完全处于混沌状态。利用混沌运动特性可以进行优化搜索，其基本思想是首先产生一组与优化变量相同数目的混沌变量，用类似载波的方式将混沌引入优化变量使其呈现混沌状态，同时把混沌运动的遍历范围放大到优化变量的取值范围，然后直接利用混沌变量搜索[59,60]。由于混沌运动具有随机性、遍历性、对初始条件的敏感性等特点，基于混沌的搜索技术无疑会比其它随机搜索更具优越性。本文将利用 $\mu=4$ 时的混沌特性，取(7.1)式的 Logistic 映射为混沌信号发生器。

7.3 基本蚁群算法改进

7.3.1 混沌初始化

蚁群算法初始化时，各路径的信息素取相同值，让蚂蚁以等概率选择路径，这样使蚂蚁很难在短时间内从大量的杂乱无章的路径中，找出一条较好的路径，所以收敛速度较慢。假如初始化时就给出启发性的信息量，可以加快收敛速度。改进的方法是利用混沌运动的遍历性，进行混沌初始化，每个混沌量对应于一条路径，产生大量的路径（如 100 条），从中选择比较优的（如 30 条），使这些路径留下信息素（与路径长度和成反比），各路径的信息量就不同，以此引导蚂蚁进行选择路径。

每个混沌量对应于一条路径是利用全排列构造的理论[62]。首先以（1，2，3，4）的全排列为例，讨论其构造，给出转换算法。所有不同排列的总计数为 4! =24，其

构造按词典序，则构造的第一位元素取最小标号，以后各位依次增大，1234 是首构造，首向量是 111，含义为每次都是取剩下物件的最小标号。按词典序构造，末构造为 4321，末向量为 432。序号 D、向量 V 和构造 C 就构成了 DVC 表，如表 6.1 所示。

表 6.1 1-4 排列的 DVC 表

D	V	C
1	111	1234
2	112	1243
3	121	1324
4	122	1342
5	131	1423
6	132	1432
7	211	2134
8	212	2143
9	221	2314
10	222	2341
11	231	2413
12	232	2431
13	311	3124
14	312	3142
15	321	3214
16	322	3241
17	331	3412
18	332	3421
19	411	4123
20	412	4132
21	421	4213
22	422	4231
23	431	4312
24	432	4321

由 DVC 表可知，D、V 和 C 之间是有关系等，它们之间有 D/V、V/D、V/C、C/V，D/C，C/D 六种转换，我们关心的 D/C 转换，目前无法直接写出转换公式，需要通过 D/V 转换，再经过 V/C 来完成。

D/V 转换公式如下：

$$\begin{cases} D_0 = D \\ v_i = \left\lceil \dfrac{D_{i-1}}{(n-i)!} \right\rceil \\ D_i = D_{i-1} - (v_i - 1)(n-i)! \quad i = 1,2,\cdots,n-1 \end{cases} \tag{7-2}$$

V/C 转换是通过向量 V 的指针功能来确定构造 C。如

$V = v_1 v_2 v_3 = 231$，则有

$v_1=2$　　$1\underline{2}34$　　$C_1=2$

$v_2=3$　　$13\underline{4}$　　$C_2=4$

$v_1=1$　　$\underline{1}3$　　$C_3=1$　　$C_4=3$

$C=C_1C_2C_3C_4=2413$

由公式(7.1)产生混沌量z_i（$0\le z_i\le 1$），则$D_0=n!z_i$，代入公式(7.2)，令$d_1=nz_i$得到：

$$\begin{cases} v_1=\lceil d_1\rceil \\ d_i=(n-i+1)(d_{i-1}-v_{i-1}+1) \quad i=2,3,\cdots,n-1 \\ v_i=\lceil d_i\rceil \end{cases} \tag{7.3}$$

再由V的指针功能来确定构造C，这样z_i与C一一对应。

7.3.2 选择较优解

蚂蚁每次周游结束后，不论蚂蚁搜索到的解如何，都将赋予相应的信息增量，比较差的解也将留下信息素，这样就干扰后续的蚂蚁进行寻优，造成大量的无效的搜索。改进的方法是，只有比较好的解才留下信息素，即只有当路径长度小于给定的值才留下信息素。

7.3.3 混沌扰动

蚁群利用了正反馈原理，在一定程度上加快了进化进程，但也存在一些缺陷，如出现停滞现象，陷入局部最优解。改进的措施加入混沌扰动，在调整信息量，在加入混沌扰量，以使解跳出局部极值区间。更新方程修改为：

$$\tau_{ij}(t+n)=\rho\cdot\tau_{ij}(t)+\Delta\tau_{ij}+qZ_{ij} \tag{7.4}$$

其中，Z_{ij}为混沌变量，由公式（7.1）迭代得到；q为系数。

7.4 混沌蚁群算法

改进后的解TSP问题的混沌蚁群算法如下：

步骤 1　$nc\leftarrow 0$，（nc为迭代步数或搜索次数），混沌初始化，调整各路径信息素，将m个蚂蚁置于n个顶点上；

步骤 2　将各蚂蚁的初始出发点置于当前解集中，对每个蚂蚁$k(k=1,2,\cdots,m)$，按概率p_{ij}^k移至下一顶点j，将顶点j置于当前解集；

步骤 3　计算各蚂蚁的路径长度 L_k $(k=1,2,\cdots,m)$，记录当前的最好解；

步骤 4　对路径长度 L_k 小于给定值的路径，按更新方程（7-4）修改轨迹强度；

步骤 5　$nc \leftarrow nc+1$；

步骤 6　若 $nc <$ 预定的迭代次数且无退化行为（即找到的都是相同解）则转步骤 2；

步骤 7　输出目前最好解。

7.5 算法测试

为了检验算法的有效性，来解决中国 31 个直辖市和省会城市的 CTSP 问题，数据来源于文献[42]和 pr76.tsp （TSPLIB 提供的最好解为 108159）。模拟退火算法采用文献[42]的算法，起始温度 $T=100000$，终止温度 $T_0=1$，退火速度 $\alpha=0.99$；遗传算法程序采用 MATLAB 的遗传算法工具箱[56]，参数如下：染色体个数 $N=30$，交叉概率 $P_c=0.2$，变异概率 $P_m=0.5$，迭代次数 100；混沌蚁群算法参数：$\alpha=1.5$，$m=30$，$\beta=2$，$\rho=0.9$，为了说明混沌初始化的优点，与随机初始化也作了比较，每种算法运行 50 次，结果如表 7.2 所示。从中可以看出基本蚁群算法上加入混沌初始化或混沌扰动后，效果比较好，同时加入混沌初始化或混沌扰动的混沌蚁群算法的效果更好。混沌蚁群算法解 CTSP 最好的解如图 7.1 所示，总路程为 15383km，混沌蚁群算法解 pr76.tsp 最好的解如图 7.2 所示，总路程为 108159。

表 7.2 结果比较

优化方法	CTSP			pr76.tsp		
	平均值（km）	最好解（km）	最差解（km）	平均值	最好解	最差解
模拟退火算法	16902	15398	18247	109008	108673	119625
遗传算法	16920	15387	17298	108994	108694	118473
文献[60]混沌搜索法	16879	15390	17396	108972	108586	115263
ACO	16248	15390	16854	108447	108354	114673
ACO+随机初始化	16222	15389	16628	108345	108346	113162
ACO+混沌扰动	16198	15387	16607	108337	108167	112315
ACO+混沌初始化+混沌扰动(CACO)	16179	15383	16432	108235	108159	110394

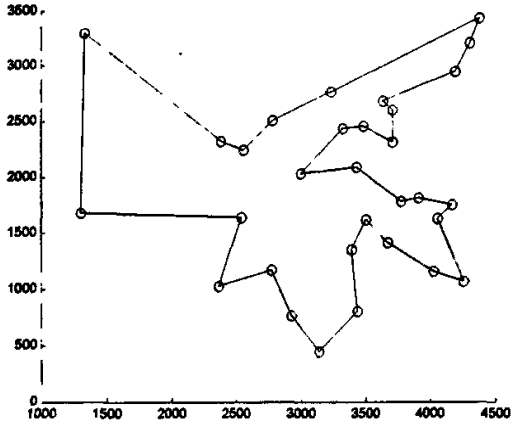

图 7.1 用混沌蚁群算法解 CTSP 最好的解

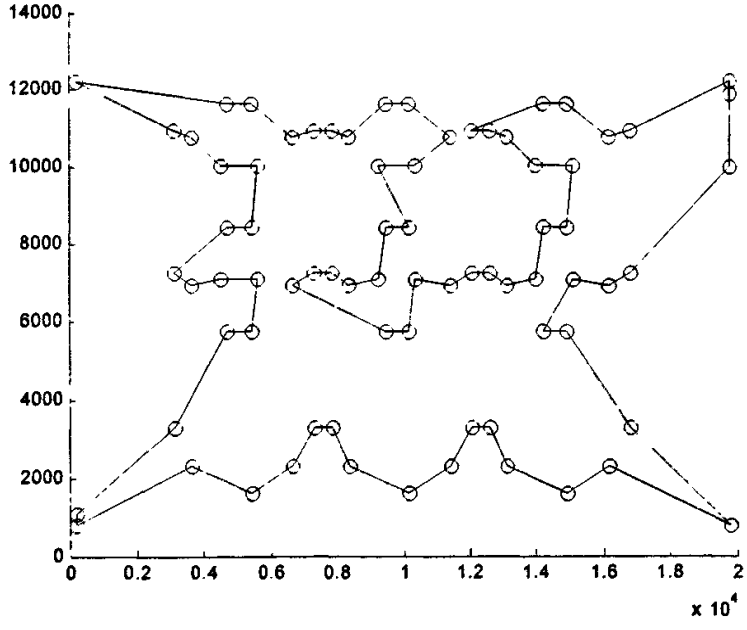

图 7.2 用混沌蚁群算法解 pr76.tsp 最好的解

7.6 本章小结

根据混沌的随机性、遍历性及规律性等特点提出的混沌蚁群算法，计算结果表明，可以显著提高计算效率，具有较大的实用价值。混沌信号产生机理简单，具有内在并行性，本文混沌信号取自 Logistic 映射，实际上混沌信号可取自其它混沌系统，至于哪个更好，有待进一步研究。

8 与粒子群优化算法混合

8.1 基本粒子群优化算法

粒子群优化(PSO: particle swarm optimization)算法是一种进化计算技术，最早是由 Kennedy 与 Eberhart 于 1995 年提出的[63, 64]。源于对鸟群捕食的行为研究的 PSO 同遗传算法类似，是一种基于迭代的优化工具。系统初始化为一组随机解，通过迭代搜寻最优值。目前已广泛应用于函数优化、神经网络训练、数据挖掘、模糊系统控制以及其他的应用领域。目前已提出了多种 PSO 改进算法[65, 66]，如自适应 PSO 算法、杂交 PSO 算法、协同 PSO 算法。

PSO 是模拟鸟群的捕食行为，设想这样一个场景：一群鸟在随机搜索食物。在这个区域里只有一块食物。所有的鸟都不知道食物在那里。但是他们知道当前的位置离食物还有多远，那么找到食物的最优策略是什么呢？最简单有效的就是搜寻目前离食物最近的鸟的周围区域。PSO 从这种模型中得到启示并用于解决优化问题。PSO 中，每个优化问题的解都是搜索空间中的一只鸟，我们称之为“粒子”。所有的粒子都有一个由被优化的函数决定的适应值，每个粒子还有一个速度决定他们飞翔的方向和距离，然后粒子们就追随当前的最优粒子在解空间中搜索。PSO 初始化为一群随机粒子(随机解)，然后通过迭代找到最优解。在每一次迭代中，粒子通过跟踪两个“极值”来更新自己。一个是粒子本身所找到的最优解，这个解叫做个体极值 *pbest*，另一个极值是整个种群目前找到的最优解，这个极值是全局极值 *gbest*，另外也可以不用整个种群而只是用其中一部分为粒子的邻居，那么在所有邻居中的极值就是局部极值。

在找到这两个最优值时，每个粒子根据如下的公式来更新自己的速度和新的位置：

$$v_{k+1} = c_0 v_k + c_1(pbest_k - x_k) + c_2(gbest_k - x_k) \tag{8.1.1}$$

$$x_{k+1} = x_k + v_{k+1} \tag{8.1.2}$$

其中：v_k 是粒子的速度向量；x_k 是当前粒子的位置；$pbest_k$ 表示粒子本身所找到的最优解的位置；$gbest_k$ 表示整个种群目前找到的最优解的位置；c_0，c_1，c_2 表示群体认知系数，c_0 一般取介于（0，1）之间的随机数，c_1，c_2 取（0，2）之间的随机数。v_{k+1} 是 v_k、$pbest_k - x_k$ 和 $gbest_k - x_k$ 矢量的和，其示意图如图 8.1.1 所示。在每一维粒子的速度都会被限制在一个最大速度 v_{max} ($v_{max} > 0$)，如果某一维更新后的速度超过用户设定的 v_{max}，那么这一维的速度就被限定为 v_{max}，即若 $v_k > v_{max}$ 时，$v_k = v_{max}$ 或

$v_k < -v_{max}$ 时，$v_k = -v_{max}$。

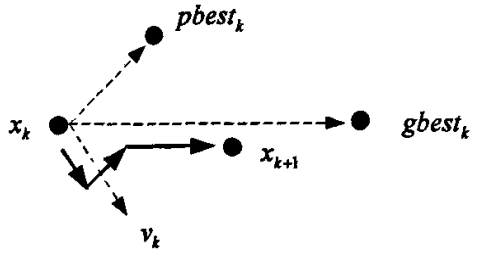

图 8.1.1 三种可能移动方向的带权值的组合

解优化问题的粒子群算法的程序框架为：

1 对每个粒子初始化，设定粒子数 n，随机产生 n 个初始解或给出 n 个初始解，随机产生 n 个初始速度；

2 根据当前位置和速度产生给个粒子的新的位置；

While（迭代次数<规定迭代次数）do

3 计算每个粒子新位置的适应值；对给个粒子，若粒子的适应值优于原来的个体极值 *pbest*，设置当前适应值为个体极值 *pbest*；

4 根据各个粒子的个体极值 *pbest* 找出全局极值 *gbest*；

5 按（8.1.1）式，更新自己的速度，并把它限制在 v_{max} 内；

6 按（8.1.2）式，更新当前的位置；

End

8.2 模拟退火思想的粒子群算法

8.2.1 几种模拟退火思想的粒子群算法

基本粒子群优化算法中，虽然粒子速度作了限制，不会变化太大，但位置更新时未作限制，就有可能新的位置会变得很坏，引起收敛速度缓慢，所以要对更新的位置也要作限制。限制的思路有两种方法：一种采用类似于速度限制的方法，给每一维变量限制一个范围。具体修改如下：

方法 1 其它步骤同基本粒子群算法的程序，第 6 步变为："按（8.1.2）式，更新当前的位置，并把它限制在 x_{max} 内"。

另一种思路采用模拟退火算法思想，模拟退火算法用于优化问题的出发点是基于物理中固体物质的退火过程与一般优化问题的相似性。算法的基本思想是从一给定解开始的，从邻域中随机产生另一个解，接受准则允许目标函数在有限范围内变坏，以一定概率接受新的解。因此有 3 种方法改进：

方法 2 按(8.1.2)式，计算新的位置，计算两个位置所引起的适应值的变化量 ΔE；

若 $\Delta E \le 0$，接受新值，否则若 $\exp(-\Delta E/T) > rand(0,1)$（$rand(0,1)$ 表示 0~1 之间的随机数），也接受新值，否则就拒绝，x_{k+1} 仍为 x_k。具体步骤如下：

1 对每个粒子初始化，设定粒子数 n，随机产生 n 个初始解或给出 n 个初始解，随机产生 n 个初始速度，给定起、止“温度” T、T_0 和退火速度 α；

2 根据当前位置和速度产生给个粒子的新的位置；

While（迭代次数<规定迭代次数）do

3 计算每个粒子新位置的适应值；对给个粒子，若粒子的适应值优于原来的个体极值 *pbest*，设置当前适应值为个体极值 *pbest*；

4 根据各个粒子的个体极值 *pbest* 找出全局极值 *gbest*；

5 按（8.1.1）式，更新自己的速度，并把它限制在 v_{max} 内；

6 按（8.1.2）式，更新当前的位置；

7 计算两个位置所引起的适应值的变化量 ΔE；若 $\Delta E \le 0$，接受新值，否则若 $\exp(-\Delta E/T) > rand(0,1)$（$rand(0,1)$ 表示 0~1 之间的随机数），也接受新值，否则就拒绝，x_{k+1} 仍为 x_k；

8 若接受新值，降温 $T \leftarrow \alpha T$，否则不降温。

End

方法 3 接受准则允许目标函数有限范围内变坏，并不按概率取舍，直接按 $\Delta E < e$，e 为按允许目标函数变坏范围。具体算法如下：

1 对每个粒子初始化，设定粒子数 n，随机产生 n 个初始解或给出 n 个初始解，随机产生 n 个初始速度；

2 根据当前位置和速度产生给个粒子的新的位置；

While（迭代次数<规定迭代次数）do

3 计算每个粒子新位置的适应值；

4 对给个粒子，若粒子的适应值优于原来的个体极值 *pbest*，设置当前适应值为个体极值 *pbest*；

5 根据各个粒子的个体极值 *pbest* 找出全局极值 *gbest*；

6 按（8.1.1）式，更新自己的速度，并把它限制在 v_{max} 内；

7 按（8.1.2）式，更新当前的位置；

8 计算两个位置所引起的适应值的变化量 ΔE；若 $\Delta E < e$，e 为按允许目标函数变

坏范围$\Delta E \le 0$，接受新值，否则就拒绝，x_{k+1}仍为x_k；

End

方法4 把方法1与方法3结合在一起，其它步骤同方法3，第7步变为："按(8.1.2)式，更新当前的位置，并把它限制在x_{max}内"。

8.2.2 算法测试

以一个简单的优化问题$\min f(x_1,x_2)=x_1^2+x_2^2$来测试各种算法。

在基本粒子群算法中，粒子数$n=4$，$c_0=1$，初始粒子位置为（2，1）、(-1，3)、(-1.5，-1)、(1.5，-1.5)，$v_{max}=5$，初试速度都为0；

方法1的参数与基本粒子群算法相同，$x_{max}=10$；

方法2的参数与基本粒子群算法相同，起始温度$T=100000$，退火速度$\alpha=0.99$；

方法3的参数与基本粒子群算法相同，$e=100$；

方法4的参数与基本粒子群算法相同，$e=100$，$x_{max}=10$。

测试的优化问题解为$(x_1^*,x_2^*)=(0,0)$，$f_{min}=0$。以目标函数$f<0.01$为停止条件，测试两种达到要求的迭代次数为衡量标准。各种算法各随机测试100次，结果如表8.2.1所示。

表8.2.1 各种策略测试结果

算法	结果比较		
	最快迭代次数	最慢迭代次数	平均迭代次数
基本粒子群算法	4	1546	312
方法1算法	4	1514	252
方法2算法	7	2234	579
方法3算法	3	1297	234
方法4算法	3	1204	215

方法1比基本粒子群方法有所改进。方法2不如方法3和方法4。因为方法2中，随着温度的降低，接受的概率变的很小，可能使有些粒子的位置停止不前，反而使收敛很慢。从表8.2.1可以看出，方法4算法是一种比较有效的算法，图8.2.1是方法4在x_1-x_2二维空间的迭代游历图。

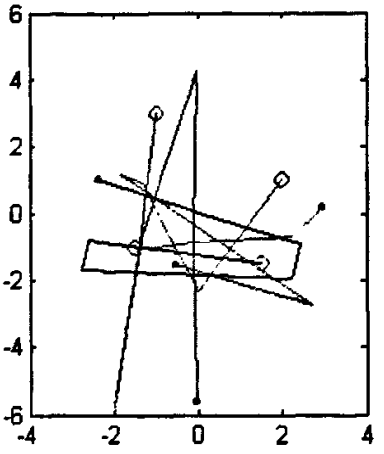

图 8.2.1 方法 4 的迭代过程（ 0 表示起始点，. 表示终点）

8.3 背包问题的混合粒子群优化算法

8.3.1 背包问题数学模型

背包问题是运筹学中一个典型的组合优化难题，有着广泛的应用背景，如货物装载问题，选址问题等。由于此问题比较简单典型，因此评价算法优劣常常以此问题作为的测试对象进行研究。背包问题属于 NP 问题，目前求解的方法有精确方法（如动态规划[39]、递归法[39]、回溯法[39]、分支限界法[39]等），近似算法（如贪心法[39]，Lagrange 法等）以及智能优化算法（如模拟退火算法[43]、遗传算法[43]，遗传退火进化算法[67]，蚁群算法[68,69]）。精确方法虽然可以得到准确解，但它属于枚举法，时间复杂性与物品数目成指数关系。近似算法和智能优化算法虽然不一定得到准确解，但可得到比较有效解，并且时间复杂性比较低。这里尝试采用粒子群优化算法解决此问题，粒子群优化算法属于智能优化算法的一种。

背包问题的描述有许多种形式，最典型的是 0-1 背包问题[1]。所谓 0-1 背包问题是指：给定 n 种物品和一个背包，物品 i 的质量是 c_i，其价值为 p_i，背包的质量容量为 C，如何选择物品，使得装入背包中物品的总价值最大？其数学模型为：

$$\begin{aligned} &\max \sum_{i=1}^{n} p_i x_i \\ &s.t. \sum_{i=1}^{n} c_i x_i \le C \\ &x_i = 0,\ 1 \quad (i = 1,2,\cdots,n) \end{aligned} \tag{8.3.1}$$

8.3.2 解 0-1 背包问题的混合粒子群算法

首先把原约束方程作为罚函数项加入到原目标中，变成无约束的优化问题[38]，即

$$\min f = -\sum_{i=1}^{n} p_i x_i + M\left\{\min\left\{0,\left[C - \sum_{i=1}^{n} c_i x_i\right]\right\}\right\}^2 \qquad (8.3.2)$$

其中 M 为一充分大的正数。

背包问题的解用向量 $X=(x_1,x_2,\cdots,x_n)^T$ 表示，粒子群算法的本质是利用个体极值信息和全局极值两个信息，来指导粒子下一步迭代位置。对于 0-1 背包问题，若按基本粒子群算法，其速度难于表达，故采用遗传算法交叉操作的思想：式（8.1.1）中的 c_0v_k 项可以看作遗传算法的变异操作，式（8.1.1）中的 $c_1(pbest_k - x_k)+c_2(gbest_k - x_k)$ 项可以看作遗传算法的交叉操作，让当前解与个体极值和全局极值分别作交叉操作，产生的解为新的位置。

交叉方法可以采用以下两种方法：

①交叉策略 A：

a. 两个串 old1 和 old2 交叉，在第二个串 old2 中随机选择一个交叉区域；

b. 将 old1 的相应的交叉区域由 old2 交叉区域代替。

例如两父串为

old1=1 0 0 1 0 1 1 1 1 0 0 1，old2=0 1 1 0 1 0 1 0 1 1 0 0

交叉区域为 0 1 0 1 1，交叉后为 new1=1 0 0 1 0 1 0 1 1 0 0 1。

②交叉策略 B：

a. 随机产生 k 个 1 到 n 的整数 $j_1,j_2,\cdots,j_k$；

b. 将 old1 的 $j_1,j_2,\cdots,j_k$ 的位置数值由 old2 相应的部分代替。

具体变异操作可以采用下面三种：

①变异策略 A：

a. 在解空间 $(x_1,x_2,\cdots,x_n)^T$ 中随机选择一块区域，如 $(x_i,x_{i+1},\cdots,x_j)^T$；

b. $(x_i,x_{i+1},\cdots,x_j)^T \leftarrow (\bar{x}_i,\bar{x}_{i+1},\cdots,\bar{x}_j)^T$。　//取反运算

如原来解为 1 0 0 1 0 1 1 1 1 0 0 1，随机产生区域为 1 1 0 0，则变异操作后的解为 1 0 0 1 0 1 1 0 0 1 1 1。

②变异策略 B：

a. 在解空间 $(x_1,x_2,\cdots,x_n)^T$ 中随机选择一块区域，如 $(x_i,x_{i+1},\cdots,x_j)^T$；

b. $(x_i,x_{i+1},\cdots,x_j)^T$ 取随机值。

③变异策略 C：

a. 在解空间 $(x_1,x_2,\cdots,x_n)^T$ 中随机选择一个 1 到 n 的整数 j；

b. $x_j \leftarrow \bar{x}_j$。

如原来解为 1 0 0 1 0 1 1 1 1 0 0 1，随机产生整数为 4，则变异操作后的解为 1 0 0 0 0 1 1 1 1 0 0 1。

解 0-1 背包问题的混合粒子群算法 hybrid _PSO 如下：

设定粒子数n_p，规定迭代次数n_{max}，随机产生n_p个初始解X_0；

根据当前位置根据式(8.3.2)计算适应值l_0，设置当前适应值为个体极值*plbest*，当前位置为个体极值位置*pxbest*，根据各个粒子的个体极值*plbest*，找出全局极值glbest和全局极值位置*gxbest*；

While（迭代次数<规定迭代次数n_{max}）do

for j=1:n_p

第j个粒子位置$X_0(j)$与*gxbest*交叉得到$X_1'(j)$；

$X_1'(j)$与*pxbest*交叉得到$X_1(j)$；

对$X_1(j)$进行变异操作；

根据当前位置计算适应值l_1；

如果$l_1(j)<plbest(j)$，则$pxbest(j)=X_1(j)$，$plbest(j)=l_1(j)$；

End

根据各个粒子的个体极值*plbst*，找出全局极值glbest和全局极值位置*gxbest*；

$X_0 \leftarrow X_1$；

End

输出全局极值glbest和全局极值位置*gxbest*。

该粒子群算法的时间复杂性估算如下：以交叉时间和变异操作花费最多，变异操作的时间与交叉操作时间相近，里面循环需要作$O(3n_p)$交叉（或变异）操作，外循环执行n_{max}次，所以时间复杂性大约为$O(3n_p n_{max})$。

8.3.3 数值仿真与分析

8.3.3.1 各种策略比较

采用文献[68]中的一个典型的背包问题数据，物品数量$n=10$，容量$C=269g$，$\{p_1,p_2,\cdots,p_{10}\}=\{55, 10, 47, 5, 4, 50, 8, 61, 85, 87\}$元，$\{c_1,c_2,\cdots,c_{10}\}=\{95, 4, 60, 32, 23, 72, 80, 62, 65, 46\}g$。2种交叉策略和3种变异策略，组合起来6

种混合粒子群算法进行比较，各种算法各测试 100 次，最优目标值为 295 元，最优解为：$X^*=\{0,1,1,1,0,0,0,1,1,1\}$。参数粒子数 $n_p=10$，最大迭代次数 $n_{max}=10$ 的结果如表 8.3.1 所示；若粒子数 $n_p=20$，最大迭代次数 $n_{max}=30$ 的结果如表 8.3.2 所示。

从表 8.3.1 和表 8.3.2 可以看出，只进行交叉操作的两个算法，效果比较差。效果最好的是交叉策略 A 和变异策略 C 的组合算法最好。并且交叉策略 A 比交叉策略 B 简单，变异策略 C 也比变异策略 A 和变异策略 B 比较简单。变异策略 A 和变异策略 B，由于变异的位数多，增加了部分差的解，因此效果不如变异方法 C。随着粒子数和最大迭代次数增加，计算效果变好，但其运算量也将增大。另外这里的交叉操作与遗传算法的交叉操作不同，遗传算法随机选择 2 个解进行交叉，而粒子群交叉操作是对每个粒子进行交叉操作，并且与个体极值和全局极值进行交叉，考虑了优生的思想，因此效果比传统的遗传效果好。它与蚁群算法相比，迭代次数明显减少。交叉策略 A 和变异策略 C 的组合算法的迭代过程如图 8.3.1 所示。

表 8.3.1　各种策略结果比较（$n_p=10$，$n_{max}=10$）

算法		平均值/元	最好解/元	最差解/元	最好解的次数
交叉策略 A		261.29	295	201	6
交叉策略 B		265.92	295	205	4
交叉策略 A	变异策略 A	280.91	295	237	15
	变异策略 B	280.49	295	231	13
	变异策略 C	285.40	295	246	28
交叉策略 B	变异策略 A	279.18	295	222	15
	变异策略 B	282.87	295	232	14
	变异策略 C	283.62	295	210	18

表 8.3.2　各种策略结果比较（$n_p=20$，$n_{max}=30$）

算法		平均值/元	最好解/元	最差解/元	最好解的次数
交叉策略 A		275.62	295	265	12
交叉策略 B		270.78	295	245	8
交叉策略 A	变异策略 A	275.52	295	245	19
	变异策略 B	294.80	295	287	90
	变异策略 C	294.98	295	294	98
交叉策略 B	变异策略 A	293.00	295	279	44
	变异策略 B	293.1	295	255	47
	变异策略 C	293.27	295	284	42

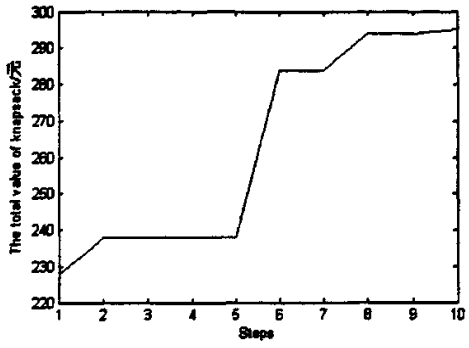

图 8.3.1 交叉策略 A 和变异策略 C 的组合算法的仿真过程

8.3.3.2 与其它算法比较

针对背包问题，对递归算法、回溯方法和蚁群算法进行了比较，采用文献[69]的测试方法，数据随机产生，各 c_i 和 p_i 在 1~100 随机生成，物品种类取值为 5~20，背包的容量 C 为总质量的 0.8 倍。每个算法运行 100 次，统计出平均运行时间，结果如表 8.3.3 所示。从表 8.3.3 可以看出交叉策略 A 和变异策略 C 的粒子群算法运行的效率较高，尽管比文献[69]蚁群算法运行稍长点，但得到精确解的次数较高（表 8.3.2）。

表 8.3.3 各种算法的平均运行时间比较

n	5~9	10	11	12	13	14	15	16	17	18	19	20
回溯方法[69]/s	<0.0001	0.240	2	3	7	16	33	42	88	190	393	620
蚁群算法[69]/ms	<1								40	50	56	66
递归算法/ms	<60	60	70	110	130	150	180	230	300	460	640	1060
交叉策略 A 和变异策略 C 的粒子群算法/ms	<50	50	65	78	84	96	100	120	130	170	180	200

8.4 混沌粒子群优化算法研究

8.4.1 基本粒子群算法不足

分析基本粒子群优化算法搜索过程有两个不足：

其一：初始化过程是随机的，随机过程虽然大多可以保证初始解群分布均匀，但对个体的质量不能保证，解群中有一部分远离最优解。如果初始解群比较好，将会有助于求解效率与解的质量。

其二：利用（8.1.1）、（8.1.2）式更新自己的速度和新的位置，本质是利用本身

信息、个体极值信息和全局极值三个信息，来指导粒子下一步迭代位置。这实际上是一个正反馈过程，当本身信息和个体极值信息占优势时，该算法很容易陷入局部最优解[70-73]。

8.4.2 混沌粒子群优化算法

将混沌引入粒子群算法就是对基本粒子群优化算法的不足提出改进，基本思路是：利用混沌运动的遍历性，产生大量初始群体，从中择优出初始群体；对当前粒子个体产生混沌扰动，以使解跳出局部极值区间。

设求解n维的优化问题：

$$\begin{aligned} &\min f(x_1,x_2,\cdots,x_n) \\ &s.t.\ \ a_i \le x_i \le b_i \end{aligned} \tag{8.4.1}$$

解优化问题的混沌粒子群优化算法步骤如下：

混沌初始化：随机产生一个n维每个分量数值在0-1之间的向量$z_1=(z_{11},z_{12},\cdots,z_{1n})$，根据(7.1)式，$z_{i+1j}=\mu z_{ij}(1-z_{ij})$ $(j=1,2,\cdots,n;i=1,2,\cdots,N-1)$得到$N$个$z_1,z_2,\cdots,z_N$。将$z_i$的各个分量载波到优化变量的取值范围：$x_{ij}=a_j+(b_j-a_j)z_{ij}$ $(j=1,2,\cdots,n;i=1,2,\cdots,N)$。计算目标函数，从$N$个初始群体中选择性能较好的$m$个解作为初始解，随机产生$m$个初始速度；

根据当前位置和速度产生各个粒子的新的位置；

随机产生一个n维每个分量数值在0-1之间的向量$u_0=(u_{01},u_{02},\cdots,u_{0n})$；

While（迭代次数k<规定迭代次数$n_{\max}$）do

For i=1:m

按（8.1.1）式，更新自己的速度，并把它限制在$v_{\max}$内；

根据(7.1)式，产生$u_1=(u_{11},u_{12},\cdots,u_{1n})$，$u_{1j}=4u_{0j}(1-u_{0j})$ $(j=1,2,\cdots,n)$，将u_1的各个分量载波到混沌扰动范围$[-\beta,\beta]$内，扰动量$\Delta x=(\Delta x_1,\Delta x_2,\cdots,\Delta x_n)$为$\Delta x_j=-\beta+2\beta u_{1j}$，$u_0=u_1$；$x_{ik+1}=x_{ik}+v_{ik+1}$，$x'_{ik+1}=x_{ik}+v_{ik+1}+\Delta x$，计算这两个位置的适应值$f$和$f'$，若$f'<f$则$x_{ik+1}=x'_{ik+1}$；

End For

$k=k+1$，计算第i个粒子的适应值f_i，若粒子的适应值优于原来的个体极值，设置当前适应值为个体极值$pfbest_{ik}$，设置当前位置为个体极值位置$pxbest_{ik}$；

根据各个粒子的个体极值 $pfbest_{it}$（$i=1,2,\cdots,m$），找出全局极值 $gfbest_t$ 和全局极值位置 $gxbest_t$；

End While

输出全局极值 $gfbest$ 和全局极值位置 $gxbest$。

8.4.3 算法测试

8.4.3.1 迭代次数测试

用CPSO对以下4个经常被国内外学者用来测试优化算法有效性的测试函数[70,71,73]进行优化计算，精度为 0.00001 时所需要的迭代次数作为比较，参数：初始群体数 $N=100$，粒子数 $m=50$，每种算法运行100次，结果如表8.4.1所示。

$$F_1=x_1^2+x_2^2,\ -1\le x_i\le 1$$

$$F_2=100(x_1^2-x_2)^2+(1-x_1)^2,\ -2\le x_i\le 2$$

$$F_3=\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{[1+0.001(x_1^2+x_2^2)]^2}+0.5,\ -1\le x_i\le 1$$

$$F_4=x_1^2+2x_2^2-0.3\cos 3\pi x_1-0.4\cos \pi x_2+0.7,\ -1\le x_i\le 1$$

表8.4.1 迭代次数结果比较

测试函数	优化方法	迭代次数最小值	迭代次数平均值	迭代次数最大值
F_1	PSO	5	34.90	107
	PSO+混沌初始化	3	33.42	110
	PSO+混沌扰动	4	31.45	115
	CPSO	3	30.85	74
F_2	PSO	5	170.52	766
	PSO+混沌初始化	10	144.42	616
	PSO+混沌扰动	7	146	505
	CPSO	2	143.92	595
F_3	PSO	8	33.46	143
	PSO+混沌初始化	3	31.90	89
	PSO+混沌扰动	4	32.35	110
	CPSO	2	27.64	69
F_4	PSO	8	83.33	315
	PSO+混沌初始化	4	88.02	289
	PSO+混沌扰动	6	78.12	375
	CPSO	4	74.94	282

CPSO 算法是指在基本粒子群优化算法上加入混沌初始化和混沌扰动。从表 8.4.1 中可以看出基本粒子群优化算法上加入混沌初始化或混沌扰动后，大多数情况效果比较好，少数情况效果稍差的原因是加入混沌后，搜索的区域较大，有可能使迭代次数增加。同时加入混沌初始化和混沌扰动的混沌粒子群优化算法的效果更好，迭代次数明显减少。

8.4.3.2 收敛性测试

同样以上面四个函数为测试函数，初始群体数 $N=100$，粒子数 $m=50$，以最大迭代次数 100 次为停止条件，每种算法运行 100 次，计算平均最优解，结果如表 8.4.2 所示。从表 8.4.2 中可以看出基本粒子群优化算法容易陷入局部最优，加入混沌初始化或混沌扰动后，或同时加入混沌初始化和混沌扰动，某种程度上可以跳出局部最优解。

表 8.4.2 平均最优解比较

测试函数	最优解	PSO	PSO-混沌初始化	PSO-混沌扰动	CPSO
F_1	0	0.0214	0.0163	0.0184	0
F_2	0	0.0235	0.0056	0.0095	0.0012
F_3	-1	-0.986	-0.997	-0.991	-1
F_4	0	0.0212	0.00724	0.00954	0.0031

8.4.3.3 与前人比较

为了更一步验证方法的有效性，考虑下面优化问题

$$F_5=\frac{1}{100}\sum_{i=1}^{100}x_i^4-16x_i^2+5x_i,\quad -10\le x_i\le 100,\quad i=1,2,\cdots,100$$

当 $x_i^*=-2.904$ 时，该函数的最小目标值为-78.3323，与前人的方法进行比较，每种算法均运行 10 次，结果如表 8.4.3 所示，从中看出本文的混沌粒子群算法的有效性。

表 8.4.3 与其他文献的结果比较

算法类型	迭代次数最小值	迭代次数平均值	迭代次数最大值
混沌优化方法[75, 76]	18760	29542	37553
文献[77]方法[75, 77]	19980	23664	27599
文献[76]方法[75, 76]	1197	5983	8892
文献[75]方法[75]	752	2146	3500
CPSO	674	1975	3277

8.5 粒子群优化算法在其他方面应用

8.5.1 指派问题的粒子群优化算法

在 8.3 节讨论了用粒子群优化算法解决背包问题，利用其方法也可解决类似的整数组合优化问题。比如指派问题，武器-目标分配问题，多处理机调度问题、集合划分问题、聚类问题等。下面简要地说明用粒子群优化算法解指派问题，武器-目标分配问题以及聚类问题等。

所谓指派问题就是把 n 个任务分配给 n 个人，使总效率最高。它是 0-1 规划的特例，目前最有效的解法是匈牙利法。指派问题的解用向量 $X=(x_1,x_2,\cdots,x_n)^T$ 表示，其含义是第 i 个人完成第 x_i 个任务，因此其解是 $1\sim n$ 的一个排列。采用这里采用遗传算法的思想的变异操作和交叉操作，具体操作不再详述。假设有 A、B、C、D 和 E 五项任务，由甲、乙、丙、丁和戊五人完成，其完成时间如下表 8.5.1。

表 8.5.1 指派问题数据

任务 / 人员	A	B	C	D	E
甲	12	7	9	7	9
乙	8	9	6	6	6
丙	7	17	12	14	9
丁	15	14	6	6	10
戊	4	10	7	10	9

采用粒子群算法参数设置如下：粒子数 $n_p=5$，最大迭代次数 $n_{\max}=20$，计算得到两组最优指派方案：甲-B，乙-D，丙-E，丁-C，戊-A 或甲-B，乙-C，丙-E，丁-D，戊-A。

8.5.2 武器-目标分配问题的粒子群优化算法

在 2.2 节讨论过用蚁群算法解武器-目标分配问题的方法，其实也可以采用粒子群优化算法来解。方法如下：WTA 问题的解用矩阵 $X=(x_{ij})_{m\times n}$ 表示，采用这里采用遗传算法的思想的变异操作和交叉操作。

交叉策略采用两种：

交叉策略 A：

(1)在第二个矩阵 old2 中随机选择若干行；

(2)将 old1 的相应的若干行由 old2 的行代替。

交叉策略 B：

(1)在第二个矩阵 old2 中随机选择一行；

(2)将 old1 的相应的行由 old2 的行代替。

变异操作可以采用下面两种：

变异策略 A：

(1)在解空间中随机选择若干行；

(2)这些行的值取随机值。

变异策略 B：

(1)在解空间中随机选择一行；

(2)该行的值取随机值。

优化问题（2.2.1）含有的罚函数项比较多，直接解决此问题，搜索的效率不高。在产生随机初始解时，让它满足$\sum_{j=1}^{n} x_{ij} \le r_i \ (i=1,2,\cdots,m)$，这样优化问题（2.2.1）简化为：

$$\min \sum_{j=1}^{n} \prod_{i=1}^{m} (1-p_{ij}x_{ij}) + N\sum_{j=1}^{n} [\min(0, s_j - \sum_{i=1}^{m} x_{ij})]^2 \qquad (8.5.1)$$

解 2.2.4.2 节的例子，采用粒子群算法参数设置如下：粒子数$n_p=30$，最大迭代次数$n_{max}=50$，2 种交叉策略和 2 种变异策略，组合起来 6 种方法进行比较，各种算法各测试 100 次，结果如表 8.5.2 所示，从表 8.5.2 可以看出，只进行交叉操作的两个算法，效果比较差。效果最好的是交叉策略 A 和变异策略 B 的组合算法最好。交叉策略 A 比交叉策略 B 让更多的优等粒子融入到当前解，改善了解的结构；由于变异策略 A 的位数变化多，增加了部分差的解，因此效果不如变异方法 B。另外这里的交叉操作与遗传算法的交叉操作不同，遗传算法随机选择两个解进行交叉，而粒子群交叉操作是对每个粒子进行交叉操作，并且与个体极值和全局极值进行交叉，考虑了优生的思想，因此效果比传统的遗传效果好。

表 8.5.2 结果比较

算法		平均值	最好解	最差解	最好解的次数
交叉策略 A		1.755	1.4	2.5	12
交叉策略 B		1.744	1.4	2.5	11
交叉策略 A	变异策略 A	1.448	1.4	1.6	60
	变异策略 B	1.42	1.4	1.6	82
交叉策略 B	变异策略 A	1.463	1.4	1.7	54
	变异策略 B	1.429	1.4	1.8	79

8.5.3 解聚类问题的粒子群算法

第 4 章讨论了采用蚁群算法求解聚类问题的算法，粒子群也可以解。聚类问题的解用矩阵$X=(x_{ij})_{m\times n}$表示，$X=(x_{ij})_{m\times n}$满足每行和为 1。先用K-均值算法作一快速分类，其结果作为其中一个粒子结果，再由粒子群优化算法在经过交叉操作后，再进行变异操作。结果如表 8.5.3 所示。从表 8.5.3 可以看出，粒子群算法几种算法都比K-均值算法效果好。交叉方法可以采用以下两种方法：

交叉策略 A：

(1)在第二个矩阵 old2 中随机选择若干行；

(2)将 old1 的相应的若干行由 old2 的行代替。

交叉策略 B：

(1)在第二个矩阵 old2 中随机选择一行；

(2)将 old1 的相应的行由 old2 的行代替。

变异操作可以采用下面两种：

变异策略 A：

(1)在解空间中随机选择若干行；

(2)这些行的值取随机值，需每行和为 1。

变异策略 B：

(1)在解空间中随机选择一行；

(2)该行的值取随机值，需每行和为 1。

对于 14 个 2 维样本蝶形数据，采用粒子群算法参数设置如下：粒子数 $n_p = 30$，最大迭代次数 $n_{max} = 50$，2 种交叉策略和由 2 种交叉策略与 2 种变异策略组合起来 4 种方法进行比较，各种算法各测试 50 次，从表 8.5.3 可以看出，混合的粒子群算法几种算法算法效果比较好。

表 8.5.3 结果比较

算法	平均值	最好解	最差解	最好解的次数	正确率/%
K-均值算法(91 次)	34.8408	26.8571	87.4	79	86.81
K-均值算法+交叉策略 A	27.0075	26.8571	34.3750	49	98
K-均值算法+交叉策略 B	27.3052	26.8571	34.3750	47	94
K-均值算法+交叉策略 A+变异策略 A	26.8571	26.8571	26.8571	50	100
K-均值算法+交叉策略 A+变异策略 B	26.8571	26.8571	26.8571	50	100
K-均值算法+交叉策略 B+变异策略 A	28.6016	26.8571	47.4222	44	88
K-均值算法+交叉策略 B+变异策略 B	27.4774	26.8571	42.8333	47	94

8.6 蚁群算法与粒子群优化算法的混合算法

8.6.1 引言

蚁群算法擅长解决离散问题的优化问题，而粒子群优化算法擅长连续问题的优化问题，特别如何把这两种算法结合在一起进行研究，总结出共同规律，无疑是一个新的方法。有两种混合思路，一种是在粒子群优化算法中加入遗传算法、模拟退火算法和蚁群算法思想，称为混合粒子群优化算法；另一种是在蚁群算法中引入粒子群优化算法思想，称为粒子群-蚁群算法。本节尝试结合这两种混合方法解决旅行商问题。

8.6.2 求解旅行商问题的混合粒子群优化算法

8.6.2.1 混合粒子群算法思路

粒子群算法的本质是利用本身信息、个体极值信息和全局极值三个信息，来指导粒子下一步迭代位置。对于 TSP 问题，其当前的位置是基本路径，若按基本粒子群算法，其速度难于表达。故采用这里采用遗传算法的思想来解决，(8.1.1) 式中的c_0v_k项可以看作遗传算法的变异操作，(8.1.1) 式中的$c_1(pbest_k - x_k)+c_2(gbest_k - x_k)$项可以看作遗传算法的交叉操作，让当前解与个体极值和全局极值分别作交叉操作，产生的解为新的位置。对于变异操作和交叉操作后，新的解可能比原来的解要坏，接受准则采用模拟退火算法的思想，允许目标函数有限范围内变坏，为简化计算量并不按概率取舍，直接按$\Delta E < e$，e为按允许目标函数变坏范围。下面讨论具体的操作过程。

8.6.2.2 变异操作和交叉操作

由路径C_0变异到另一条路径C_1，常用的有 6 种变异策略 A、B、C、D、E、F，分别类似与 5.2.3 节中的算法 A、B、C、D 和蚁群模拟退火算法 I、II。这里假不再重复叙述。

交叉的方法很多，主要有 4 种交叉策略 A、B、C、D，与 6.3.3 节中交叉策略 A、B、C、D 类似，这里假也不再重复叙述。

8.6.2.3 混合粒子群算法步骤

解 TSP 的混合粒子群算法如下：

设定粒子数n_p，规定迭代次数$n_{\max}$，随机产生n_p个初始解（初始路径）C_0；

根据当前位置计算适应值（各路径的长度）$ltsp0$，设置当前适应值为个体极值$plbest$，当前位置为个体极值位置$pcbest$，根据各个粒子的个体极值$plbest$，找出全局极值glbest和全局极值位置$gcbest$；

While（迭代次数<规定迭代次数$n_{\max}$）do

For j=1:n_p

第j个粒子路径$C_0(j)$与$gcbest$交叉得到$C_1'(j)$；

$C_1'(j)$与$pcbest$交叉得到$C''_1(j)$；

对$C''_1(j)$产生变异得到$C_1(j)$；

根据当前位置计算适应值$ltsp1$；

计算两个位置所引起的适应值的变化量ΔE；若$\Delta E < e$，e为按允许目标函数变坏范围$\Delta E \le 0$，接受新值，否则就拒绝，第j个粒子路径$C_1(j)$仍然为$C_0(j)$；

如果 ltspl(j)<plbest(j)，则$pcbest(j)=C_1(j)$，$plbest(j)=ltsp1(j)$；

End For

根据各个粒子的个体极值$plbest$，找出全局极值glbest和全局极值位置

gcbest；

$C_0 \leftarrow C_1$；

End While

最后输出全局极值glb*est*和全局极值位置*gcbest*。

混合粒子群算法的时间复杂性大可估算如下：以交叉时间花费最多，里面循环需要作$O(2n_p)$交叉操作，外循环执行$n_{\max}$次，所以时间复杂性大约为$O(2n_p n_{\max})$。因此该混合粒子群优化算法实质混合了遗传算法、模拟退火算法和蚁群算法的多种思想。

8.6.2.4 算法测试

分别采用模拟退火算法、遗传算法和混合粒子群算法来解决中国31个直辖市和省会城市的CTSP问题，数据来源于文献[42]。模拟退火算法采用文献[42]的算法，起始温度$T=100000$，终止温度$T_0=1$，退火速度$\alpha=0.99$；遗传算法程序采用MATLAB的遗传算法工具箱[56]，参数如下：染色体个数$N=30$，交叉概率$P_c=0.2$，变异概率$P_m=0.5$，迭代次数100；混合粒子群算法参数为：粒子数$n_p=30$，最大迭代次数$n_{\max}=2000$，$e=100$，混合粒子群算法分别4种交叉策略和6种变异策略，组合起来24种方法进行比较。对各种算法测试100次，结果如表8.6.1所示。

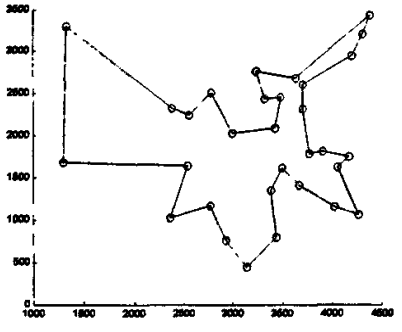

图8.6.1 目前用模拟退火算法解CTSP最好的解

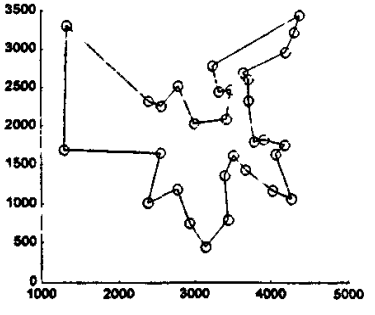

图8.6.2 用遗传算法解CTSP最好的解

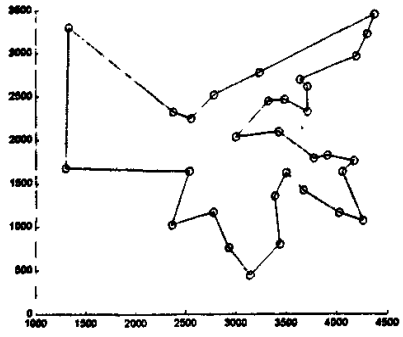

图 8.6.3 用混合粒子群算法解 CTSP 最好的解

表 8.6.1 几种算法测试结果

算法		平均值（km）	最好解（km）	最差解（km）
模拟退火算法		16902	15398	18247
遗传算法		16920	15387	17298
交叉策略 A	变异策略 A	16368	15390	17682
	变异策略 B	15738	15391	16649
	变异策略 C	15965	15383	17456
	变异策略 D	16854	15392	17346
	变异策略 E	16885	15383	18023
	变异策略 F	15849	15383	17367
交叉策略 B	变异策略 A	16844	15393	17673
	变异策略 B	16963	15389	17743
	变异策略 C	15893	15383	17896
	变异策略 D	16874	15390	18023
	变异策略 E	16894	15393	17931
	变异策略 F	15835	15383	16697
交叉策略 C	变异策略 A	16848	15394	16935
	变异策略 B	16383	15390	17849
	变异策略 C	16427	15383	17384
	变异策略 D	16853	15389	17673
	变异策略 E	16389	15383	17843
	变异策略 F	15733	15383	16682
交叉策略 D	变异策略 A	16383	15393	18024
	变异策略 B	16795	15390	17384
	变异策略 C	16383	15383	17032
	变异策略 D	15963	15389	18024
	变异策略 E	15837	15383	17473
	变异策略 F	15737	15383	16740

从表 8.6.1 可以看出，混合粒子群算法的 24 种组合大都比较好，特别交叉策略 D 和变异策略 F 的混合粒子群算法是最好的算法。图 8.6.1 是中国 31 城市的 CTSP 目前用模拟退火算法得到的最好解[42]，总路程为 15398km；图 8.6.2 的遗传算法的最好的解的总路程为 15387km；图 8.6.3 的混合粒子群最好的解总路程为 15383km，其路径为：北京—沈阳—长春—哈尔滨—呼和浩特—银川—兰州—西宁—乌鲁木齐—拉萨—成都—昆明—贵阳—南宁—海口—广州—长沙—武汉—南昌—福州—台北—杭州—上海—南京—合肥—郑州—西安—太原—石家庄—济南—天津。这说明采用混合粒子群是一个比较好的算法。用模拟退火算法解旅行商问题时，在邻域内找下一解实质上只进行变异操作，变异操作采用随机策略，未充分利用已知信息；而本节除了采用变异操作，还进行了交叉操作，在采用变异操作时，特别采用了蚁群算法的思想“距离近的邻接点以较大的概率被选”，因而效果比较好。用遗传算法解旅行商问题时，采用随机策略，虽然保证解分布比较均匀，但个体的质量不能保证，解群中的个体很大一部分远离最优解，这样在经过交叉操作后的个体的性能不能保证比原来的好；而本节采用的交叉操作，是与个体极优和群体极优进行交叉，无疑会改变下一代群体的质量，有助于提高搜索效率。

8.6.3 求解旅行商问题的粒子群-蚁群算法

8.6.3.1 粒子群-蚁群算法思想

蚁群算法利用了信息素进行传递信息，而粒子群优化算法利用了本身信息、个体极值信息和全局极值三个信息，来指导粒子下一步迭代位置。蚁群算法利用正反馈原理和某种启发式算法的有机结合，容易出现早熟现象以及陷入局部最优解。混合的思路是让蚂蚁也具有“粒子”的特性，首先蚂蚁按照蚁群算法，完成一次遍历后，再让蚂蚁根据局部最优解和全局最优解进行调整。调整思路如下：对于旅行商问题，其当前的位置是基本路径，让当前解与个体极值和全局极值分别作交叉操作，产生的解为新的位置，再作变异操作。变异操作和交叉操作可参看 5.2.3 节和 6.3.3 节。

8.6.3.2 粒子群-蚁群算法步骤

解 TSP 问题的混合的算法算法如下：

步骤 1　$nc \leftarrow 0$，（nc 为迭代步数或搜索次数），初始化，产生大量的路径（如 100 条），从中选择比较优的（如 30 条），使这些路径留下信息素，将 m 个蚂蚁置于 n 个顶点上；

步骤 2 根据当前位置计算适应值（各路径的长度）$ltsp0$，设置当前适应值为个体极值 $ptbest$，当前位置为个体极值位置 $pcbest$，根据各个粒子的个体极值 $ptbest$，

找出全局极值 *gtbest* 和全局极值位置 *gcbest*；

步骤 3　将各蚂蚁的初始出发点置于当前解集中，对每个蚂蚁 $k(k=1,2,\cdots,m)$，按概率 p_{ij}^{k} 移至下一顶点 j，将顶点 j 置于当前解集；

步骤 4　对每个蚂蚁进行如下操作，第 j 个蚂蚁路径 $C_0(j)$ 与 *gcbest* 交叉得到 $C_1^{'}(j)$，$C_1^{'}(j)$ 与 *pcbest* 交叉得到 $C''_1(j)$，对 $C''_1(j)$ 产生变异得到 $C_1(j)$，根据当前位置计算路径长度 *ltsp*1，若新的目标函数变好，接受新值，否则就拒绝，第 j 个粒子路径 $C_1(j)$ 仍然为 $C_0(j)$，重新找出各个蚂蚁子的个体极值 *ptbest* 和极值位置 *pcbest*，找出全局极值 *gtbest* 和全局极值位置 *gcbest*；

步骤 5　计算各蚂蚁的路径长度 L_k $(k=1,2,\cdots,m)$，记录当前的最好解；

步骤 6　对路径长度 L_k 小于给定值的路径，按更新方程（1.2）修改轨迹强度；

步骤 7　$nc \leftarrow nc+1$；

步骤 8　若 $nc <$ 预定的迭代次数且无退化行为（即找到的都是相同解）则转步骤 2；

步骤 9　输出目前最好解。

8.6.3.3 算法测试

为了检验算法的有效性，分别采用模拟退火算法、遗传算法和混合算法来解决 30 个城市的 Oliver30。模拟退火算法采用文献[42]的算法，起始温度 $T=100000$，终止温度 $T_0=1$，退火速度 $\alpha=0.99$；遗传算法程序采用 MATLAB 的遗传算法工具箱[56]，参数如下：染色体个数 $N=30$，交叉概率 $P_c=0.2$，变异概率 $P_m=0.5$，迭代次数 100；混合算法参数：$\alpha=1.5$，$m=30$，$\beta=2$，$\rho=0.9$，混合算法分别采用 4 种交叉策略和 4 种变异策略(A、B、C 和 D)组合起来 16 种方法进行比较。对各种算法测试 20 次，结果如表 8.6.2 所示。

从表 8.6.2 可以看出，混合粒子群算法的 16 种组合大都比较好，特别交叉策略 B 和变异策略 B 的混合粒子群算法是最好的算法。交叉策略 B 比其他交叉策略效果好，因为其他交叉策略将交叉区域加入到随机的位置，有可能增加了部分差的解。在变异策略中，变异策略 B 效果比较好，其变异的幅度比较小，避免出现比较差的解。最好的解的总路程为 423.7406，如图 2.5.1(第 2.5 节)所示，图 8.6.4 是交叉策略 D 和变异策略 D 的混合粒子群算法得到的较好的解，总路程为 423.9490。

利用蚁群算法和粒子群优化算法特性，提出的混合算法可以显著提高计算效率，具有较大的实用价值。蚁群算法和粒子群优化算法研究处于初期，还有许多问题值得研究，如算法的收敛性、理论依据等。本节用混合算法解决了离散性优化问题，对于连续性优化问题有待于进一步研究。

表 8.6.2 几种算法测试结果

算法			平均值	最好解	最差解
模拟退火算法			438.5223	424.6918	479.8312
遗传算法			483.4572	467.6844	502.5742
基本蚁群算法			550.0346	491.9581	599.9331
混合算法	交叉策略 A	变异策略 A	439.4948	425.6490	456.7721
		变异策略 B	441.9257	428.7296	455.2382
		变异策略 C	437.0028	426.6002	446.2394
		变异策略 D	438.7750	425.4752	455.2929
	交叉策略 B	变异策略 A	438.9350	424.6354	457.9062
		变异策略 B	431.4987	423.7406	447.6865
		变异策略 C	435.4220	424.9003	447.3223
		变异策略 D	439.4777	426.1972	465.9935
	交叉策略 C	变异策略 A	444.1723	429.3803	459.4925
		变异策略 B	438.5871	426.3076	455.5854
		变异策略 C	440.4201	427.6016	454.8674
		变异策略 D	439.5524	424.4643	461.7948
	交叉策略 D	变异策略 A	439.0477	424.6727	451.8001
		变异策略 B	436.0081	423.7406	460.6230
		变异策略 C	438.8091	425.8201	455.4830
		变异策略 D	436.4577	423.9490	457.3155

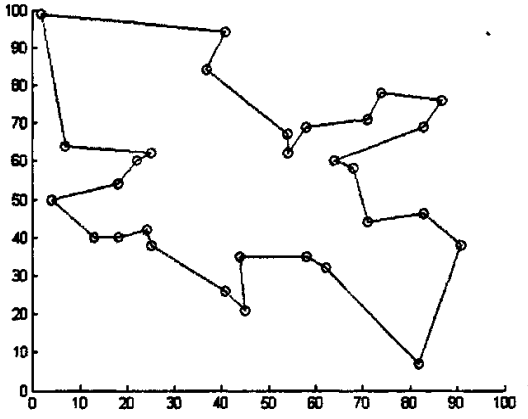

图 8.6.4 用交叉策略 D 和变异策略 D 的混合算法解 Oliver30 的最好解

8.7 本章小结

在计算智能领域目前主要有两种基于群智能的算法：蚁群算法和粒子群优化算法。前者是对蚂蚁群落食物采集过程的模拟，粒子群优化算法也是起源对简单社会系统的模拟，模拟鸟群觅食的过程。蚁群算法擅长解决离散问题的优化问题，而粒子群优化算法擅长连续问题的优化问题。本章先对粒子群算法进行了研究，提出了模拟退火思想的粒子群算法、混沌粒子群优化算法来解连续性优化问题；提出了混合其他算法的粒子群优化算法来求解背包问题。

最后提出了与粒子群优化算法的混合两种算法，根据蚁群算法与粒子群优化算法的特性，提出了两种混合算法。一种是结合遗传算法、蚁群算法和模拟退火算法的思想提出混合粒子群算法。另一种是粒子群-蚁群算法，首先随机产生若干组比较好的解生成信息素分布，然后由蚁群算法根据累计更新的信息素找出若干组解后，再由粒子群算法进行交叉、变异操作，得到更有效的解。与模拟退火算法、标准遗传算法和标准蚁群算法进行比较，混合算法效果很好。

9 最短路的蚁群算法收敛性分析

9.1 引言

蚁群算法的发展需要坚实的理论基础，这方面的工作还极其缺乏。关于蚁群算法收敛性的数学证明并不太多，大多数文献只研究了某些特定的蚁群算法的收敛性，对于一般性蚁群算法的收敛性证明未给出。Gutjahr[78]在其论文中借助图论工具证明了蚁群算法的收敛性，该文将问题实例转化为构造图，将可行解编码转化为构造图中的路，在此基础上，在某些给定条件下，蚁群算法可以任意接近1的概率收敛到全局最优解，但这只是一个初步工作。Thomas Stuezle[79]从极限理论对一种特殊的蚁群算法收敛性进行了分析。孙焘等[80]提出了一种可用于函数优化的简单蚂蚁算法，并对其收敛性作了研究。段海滨等[81]提出与文献[79]很类似的方法，对算法的全局收敛性进行了研究。由于蚂蚁算法在实际应用中形式千变万化，蚁群算法最初出发点是在一个图中找最短路，本章从一个最简单的最短路问题的蚁群算法入手，对其收敛性及有关参数问题进行一些初步分析。

9.2 最短路的蚁群算法收敛性分析

假设A、B之间有m条路径AC_1B、AC_2B、…、AC_mB（如图9.1），路径长度分别为d_1、d_2、…、d_m，不失一般性，假设$d_1 \le d_2 \le \cdots \le d_m$。$n$只蚂蚁在$A$、$B$之间往返爬行，依照蚁群算法，随着时间的推移，大多数蚂蚁应在路径AC_1B上，那么就认为A、B之间的最短路径是AC_1B，下面推导寻找最短路的蚁群算法的选择AC_1B的概率接近1条件。

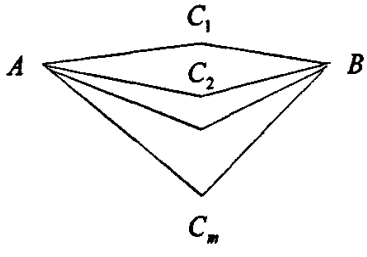

图9.1 最短路问题

假设$q_{i,k}$表示蚂蚁爬行第k趟后留在路径AC_iB上的平均信息素，$p_{i,k}$表示蚂蚁爬行第k趟后选择路径AC_iB的平均概率。初始时刻，各条路径上的信息量相等为C。

定理1 当$\alpha \ge 0$，$\beta \ge 0$时，$q_{1,k} \ge q_{2,k} \ge \cdots \ge q_{m,k}$，$p_{1,k} \ge p_{2,k} \ge \cdots \ge p_{m,k}$，$k=1,2,\cdots$。

证 当$\alpha \geq 0$，$\beta \geq 0$时，则

$p_{i,0} = \dfrac{\dfrac{C^{\alpha}}{d_i^{\beta}}}{\sum_{j=1}^{m} \dfrac{C^{\alpha}}{d_j^{\beta}}}$，由$d_1 \leq d_2 \leq \cdots \leq d_m$，可知：$p_{1,0} \geq p_{2,0} \geq \cdots \geq p_{m,0}$

$q_{i,1} = \rho C + np_{i,0}\dfrac{Q}{d_i}$，从而$q_{1,1} \geq q_{2,1} \geq \cdots \geq q_{m,1}$

$p_{i,1} = \dfrac{\dfrac{q_{i,1}^{\alpha}}{d_i^{\beta}}}{\sum_{j=1}^{m} \dfrac{q_{j,1}^{\alpha}}{d_j^{\beta}}}$，$p_{1,1} \geq p_{2,1} \geq \cdots \geq p_{m,1}$

以此类推，得到：$q_{i,k} = \rho q_{i,k-1} + np_{i,k-1}\dfrac{Q}{d_i}$，$p_{i,k} = \dfrac{\dfrac{q_{i,k}^{\alpha}}{d_i^{\beta}}}{\sum_{j=1}^{m} \dfrac{q_{j,k}^{\alpha}}{d_j^{\beta}}}$

由数学归纳法可知：当$\alpha \geq 0$，$\beta \geq 0$时，$q_{1,k} \geq q_{2,k} \geq \cdots \geq q_{m,k}$，$p_{1,k} \geq p_{2,k} \geq \cdots \geq p_{m,k}$。

定理 1 说明每趟运行完后，AC_1B上的平均信息素最多，选择路径AC_1B上平均概率最大。

定理2 当$\alpha \geq 1$，$\beta \geq 0$时，$\dfrac{q_{j,k}}{q_{1,k}} - \dfrac{q_{j,k-1}}{q_{1,k-1}} < 0$，$k = 1,2,\cdots$，$j = 2,3,\cdots,m$。

证：$\dfrac{q_{j,k}}{q_{1,k}} - \dfrac{q_{j,k-1}}{q_{1,k-1}} = \dfrac{\rho q_{j,k-1} + np_{j,k-1}\dfrac{Q}{d_j}}{\rho q_{1,k-1} + np_{1,k-1}\dfrac{Q}{d_1}} - \dfrac{q_{j,k-1}}{q_{1,k-1}} = \dfrac{nQ\left(\dfrac{p_{j,k-1}q_{1,k-1}}{d_j} - \dfrac{p_{1,k-1}q_{j,k-1}}{d_1}\right)}{\left(\rho q_{1,k-1} + np_{1,k-1}\dfrac{Q}{d_1}\right)q_{1,k-1}}$

若要求$\dfrac{q_{j,k}}{q_{1,k}} - \dfrac{q_{j,k-1}}{q_{1,k-1}} < 0$，即要求：$\dfrac{p_{j,k-1}q_{1,k-1}}{d_j} < \dfrac{p_{1,k-1}q_{j,k-1}}{d_1}$

把$p_{j,k-1}$和$p_{1,k-1}$的公式代入上式得到：$\dfrac{q_{j,k-1}^{\alpha}q_{1,k-1}}{d_j^{\beta+1}} < \dfrac{q_{1,k-1}^{\alpha}q_{j,k-1}}{d_1^{\beta+1}}$

化简得到：$\dfrac{q_{j,k-1}^{\alpha-1}}{d_j^{\beta+1}} < \dfrac{q_{1,k-1}^{\alpha-1}}{d_1^{\beta+1}}$

由于$q_{j,k-1}<q_{1,k-1}$，$d_j>d_1$，因此$\alpha\geq 1$，$\beta\geq 0$就可保证$\frac{q_{j,k-1}^{\alpha-1}}{d_j^{\beta+1}}<\frac{q_{1,k-1}^{\alpha-1}}{d_1^{\beta+1}}$。

从而，$\frac{q_{j,k}}{q_{1,k}}-\frac{q_{j,k-1}}{q_{1,k-1}}<0$，$k=1,2,\cdots$，$j=2,3,\cdots,m$。

定理 2 说明随着时间的推移，$\frac{q_{j,k}}{q_{1,k}}$越来越小。

定理 3 当$\alpha\geq 1$，$\beta\geq 0$时，$p_{1,k}>p_{1,k-1}$，$k=1,2,\cdots$。

证 因为$p_{1,k}=\frac{\frac{q_{1,k}^{\alpha}}{d_1^{\beta}}}{\sum_{j=1}^{m}\frac{q_{j,k}^{\alpha}}{d_j^{\beta}}}=\frac{1}{1+(\frac{d_1}{d_2})^{\beta}(\frac{q_{2,k}}{q_{1,k}})^{\alpha}+(\frac{d_1}{d_3})^{\beta}(\frac{q_{3,k}}{q_{1,k}})^{\alpha}+\cdots+(\frac{d_1}{d_m})^{\beta}(\frac{q_{m,k}}{q_{1,k}})^{\alpha}}$

$$\frac{1}{p_{1,k}}-\frac{1}{p_{1,k-1}}=(\frac{d_1}{d_2})^{\beta}[(\frac{q_{2,k}}{q_{1,k}})^{\alpha}-(\frac{q_{2,k-1}}{q_{1,k-1}})^{\alpha}]+(\frac{d_1}{d_3})^{\beta}[(\frac{q_{3,k}}{q_{1,k}})^{\alpha}-(\frac{q_{3,k-1}}{q_{1,k-1}})^{\alpha}]+$$
$$\cdots+(\frac{d_1}{d_m})^{\beta}[(\frac{q_{m,k}}{q_{1,k}})^{\alpha}-(\frac{q_{m,k-1}}{q_{1,k-1}})^{\alpha}]$$

由定理 2 可知，$\frac{q_{j,k}}{q_{1,k}}<\frac{q_{j,k-1}}{q_{1,k-1}}$

因此$\frac{1}{p_{1,k}}-\frac{1}{p_{1,k-1}}<0$，即$p_{1,k}>p_{1,k-1}$。

定理 3 说明随着时间的推移，选择路径AC_1B的平均概率越来越大。

定理 4 当$\alpha\geq 1$，$\beta\geq 0$时，$\lim_{k\to\infty}p_{1,k}=1$

证 由定理 3 可知，当$\alpha\geq 1$，$\beta\geq 0$时，$p_{1,k}>p_{1,k-1}$，序列$p_{1,k}$单调递增，根据高等数学极限理论“单调有界数列必有极限，并收敛到其上确界”可知，所以当$k\to\infty$时，有$\lim_{k\to\infty}p_{1,k}=1$

定理 4 的$\alpha\geq 1$，$\beta\geq 0$是$\lim_{k\to\infty}p_{1,k}=1$的充分条件，不是必要条件。定理 4 说明随着时间的推移，选择路径AC_1B的平均概率接近于 1。

9.3 仿真算例

假设A、B之间有 4 条路径AC_1B、AC_2B、AC_3B、AC_4B，路径长度分别为 1、

1.05、1.1、1.15，蚂蚁数$n=100$，初试信息量$C=30$，$Q=1$，$\rho=0.6$。当$\alpha=2$，$\beta=2$时，算法很快收敛，信息素和选择概率的结果如图 9.2 和图 9.3；当$\alpha=0.8$，$\beta=1$时，算法运行了 200 趟后还没有收敛，信息素和选择概率的结果如图 9.4 和图 9.5；当$\alpha=0.8$，$\beta=0$时，算法运行了 200 趟后还没有收敛，信息素和选择概率的结果如图 9.6 和图 9.7。当$\alpha=1.1$，$\beta=-0.9$时，信息素和选择概率的结果如图 9.8 和图 9.9，虽然收敛，但不一定收敛到最短路，如图 9.8 和图 9.9 就收敛到路径AC_2B中。尽管$\alpha\geq1$，$\beta\geq0$是收敛的充分条件，但从图 9.4-9.9 可以看出，$\alpha<1$，$\beta<0$的效果不好。因此建议参数：$\alpha\geq1$，$\beta\geq0$。

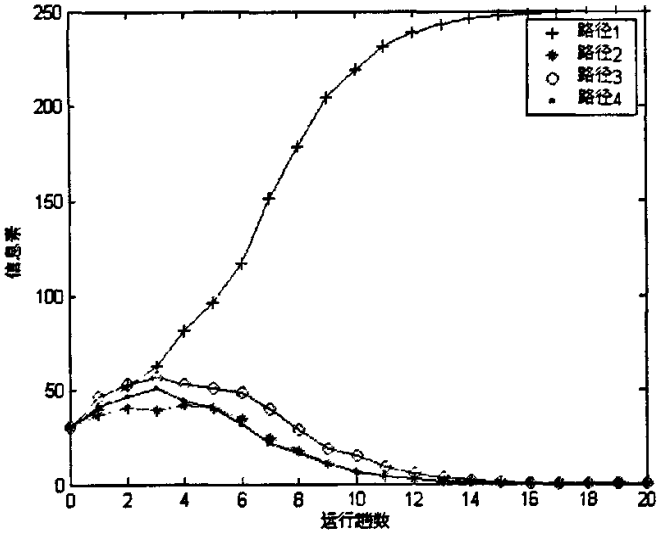

图 9.2 当$\alpha=2$，$\beta=2$时蚁群算法信息素变化规律

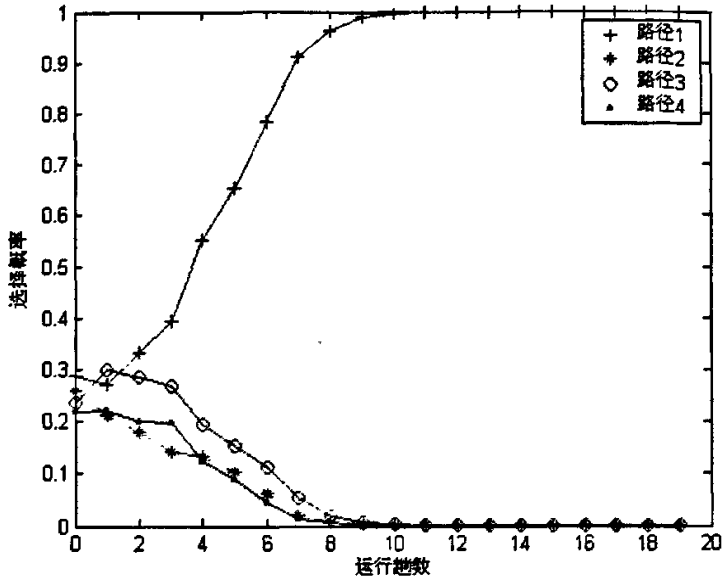

图 9.3 当$\alpha=2$，$\beta=2$时蚁群算法选择概率变化规律

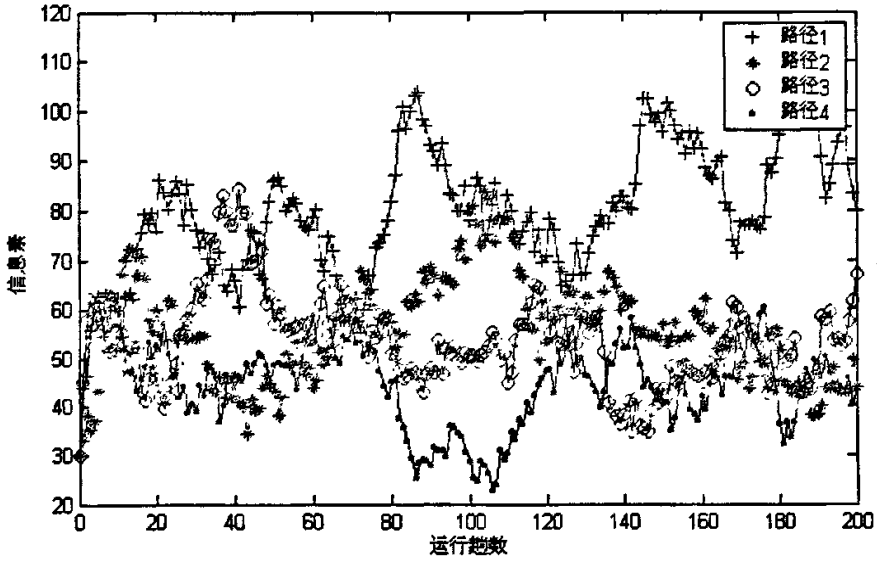

图 9.4 当$\alpha=0.8$，$\beta=1$时蚁群算法信息素变化规律

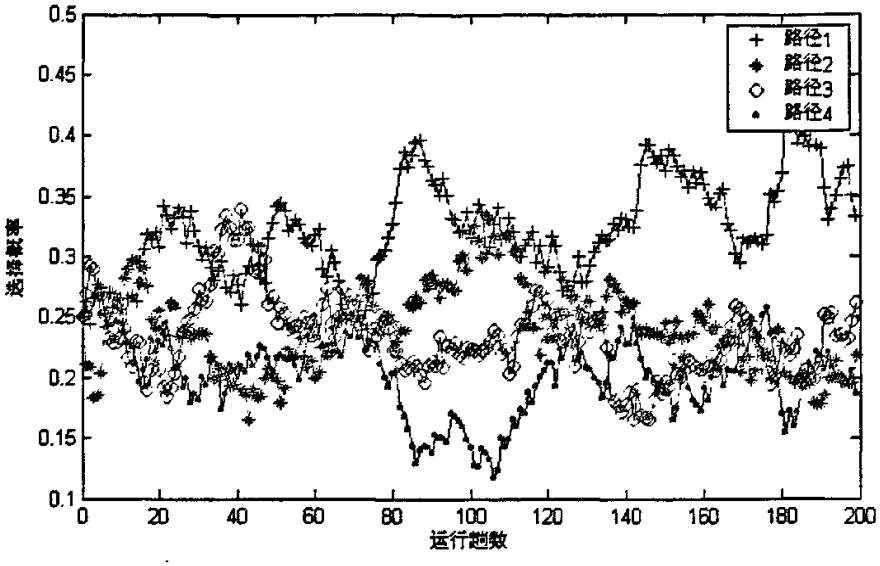

图 9.5 当$\alpha=0.8$，$\beta=1$时蚁群算法选择概率变化规律

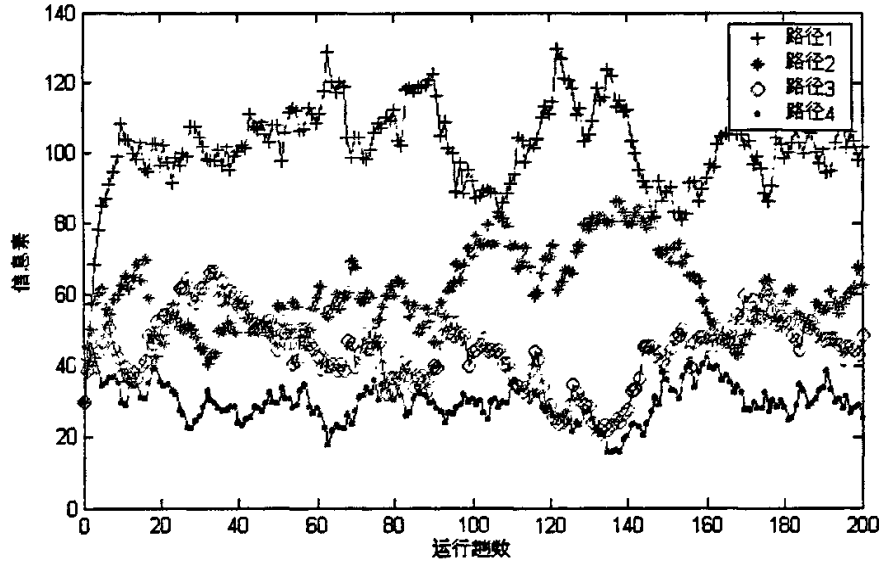

图 9.6 当$\alpha=0.8$，$\beta=0$时蚁群算法信息素变化规律

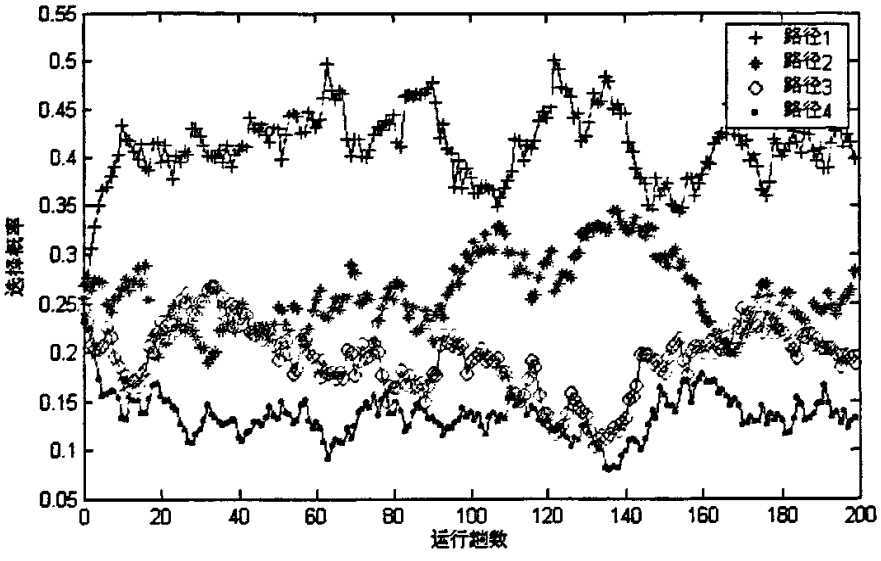

图 9.7 当$\alpha = 0.8$，$\beta = 0$时蚁群算法选择概率变化规律

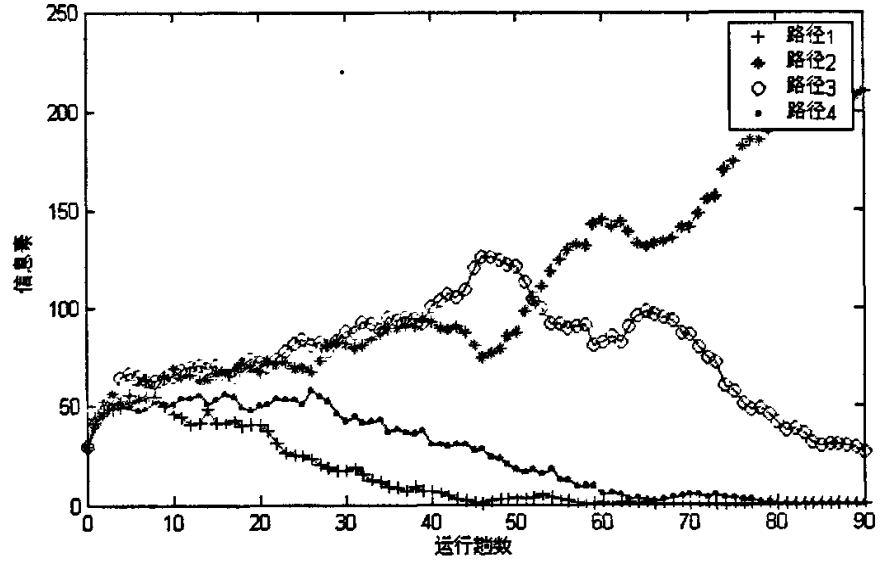

图 9.8 当$\alpha = 1.1$，$\beta = -0.9$时蚁群算法信息素变化规律

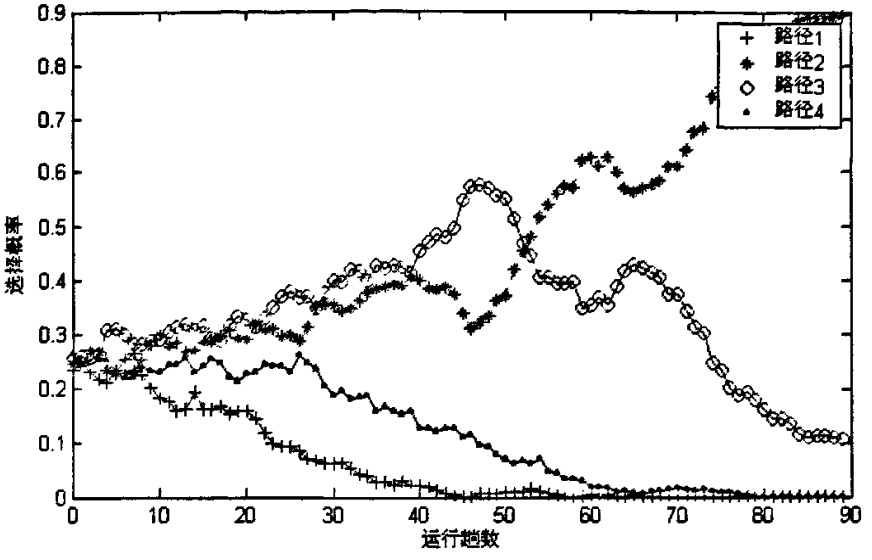

图 9.9 当$\alpha=1.1$，$\beta=-0.9$时蚁群算法选择概率变化规律

9.4 本章小结

用蚁群算法来求解两个点之间多路径的最短路问题的收敛性进行了研究。虽然研究的对象比较简单，但其具有代表性。复杂的问题可以转化为此问题，如多点的最短路问题可以完全枚举出两点之间各种路径，从而转化成两个点之间多路径的最短路问题。对于旅行商问题也可以转化成两个点之间多路径的最短路问题。那么由收敛性得到的一些结论同样适用。不对旅行商问题进行转化，直接讨论其收敛性，若按本章的方法比较复杂，简便的方法有待进一步研究。对于更一般性优化问题，采用蚁群算法来解，其收敛性分析应具体算法具体分析。

结束语

经过十年多的发展，蚁群算法凭借其简单的算法结构和突出的问题求解能力，吸引了众多研究者的目光，并取得了令人注目的成果，国内已出版了有关蚁群算法的专著[82]，有许多硕士[83, 84, 85]和博士[86, 87, 88]以蚁群算法为研究对象。大量的研究成果证明社会型生物，或者说自然系统为人工智能处理系统和算法的设计提供了有益的启发。人工群智能系统作为其中的一个重要组成部分已显示出潜在的优势：灵活、稳定、分布式控制和自组织能力。而且工程问题中日益复杂的信息处理需求，尤其是动态特性突出的问题，为群智能的应用研究提供了广阔的空间。

蚁群算法是从真实蚁群的觅食行为中受到启发而发展起来的一种新型的模拟进化算法，它是一种并行算法，所有“蚂蚁”独立行动，没有监督机构；它是一种协作算法，人工蚂蚁通过间接通讯、相互协作来寻找问题的最优解；它是一种鲁棒算法，只要对算法作小小的修改，就可以运用于别的优化问题。目前，该算法在组合优化、函数优化、机器人路径规划、数据挖掘等领域的应用已显示出其求解复杂优化问题的优势。

本课题研究了蚁群算法的理论，对其作了改进，将其应用在很多领域中。本论文在以下几个方面取得了一些成果：

（1）提出了一种解决一般性非线性整数规划问题的蚁群算法。对几个典型的整数规划问题，如武器-目标分配、多处理机调度、可靠性优化等问题，根据各问题的特点，采用了不完全相同方法，并与其他方法作了比较，效果比有效。提出了一种求解旅行商问题的多样信息素的蚁群算法。根据蚁群算法信息素更新的特性，把蚁群的三种不同的信息素更新方式混合在一起，既利用了局部信息，又考虑了整体信息。提出一种解决连续优化问题的蚁群算法。思路把连续空间进行离散化，分成若干空间网格点，采用蚁群算法找出信息量大的空间网格点，缩小变量范围，在此点附近进行人工蚁群移动，直到网格的间距小于预先给定的精度。

（2）对模式识别中典型问题-聚类问题进行了研究，提出了两种求解聚类问题的蚁群算法。一种方法是把聚类问题变换成蚂蚁寻食过程，蚂蚁在模式样本到聚类中心的路径上留下外激素，引导其他蚂蚁选择；另一种是与K-均值算法混合，用K-均值算法作快速分类，根据分类结果更新信息素，指导其它蚂蚁选择。测试数据显示与K-均值算法混合的算法效果相当好。

（3）提出了与模拟退火算法混合的两种算法。一种是在模拟退火算法中运用蚁群算法思想找邻域的解，称为蚁群模拟退火算法，并用该算法解决圆排列问题；另一种是采用模拟退火算法生成信息素分布，蚂蚁算法寻优中采用模拟退火的在邻域内找另外一个解的策略，称为模拟退火蚁群算法，并用该算法解决旅行商问题。

（4）提出了与遗传算法混合的蚁群算法。由遗传算法生成初始信息素分布，在蚂蚁算法寻优中，采用交叉和变异的策略。以旅行商问题的典型的测试问题为测试对象，与其他算法进行了比较，该算法有其优越性。

（5）提出了与混沌理论混合的蚁群算法。采用混沌初始化进行改善个体质量和利用混沌扰动避免搜索过程陷入局部极值。与其他算法进行了比较，该算法比较有效。

（6）对群智能算法中另一种算法-粒子群算法进行了研究，提出了模拟退火思想的粒子群算法、混沌粒子群优化算法来解连续性优化问题；提出了混合其他算法的粒子群优化算法来求解背包问题。根据蚁群算法与粒子群优化算法的特性，提出了两种混合算法。一种是结合遗传算法、蚁群算法和模拟退火算法的思想提出混合粒子群算法。用该算法求解著名的旅行商问题，被证实是一种比较有效的方法。另一种是粒子群-蚁群算法，首先随机产生若干组比较好的解生成信息素分布，然后由蚁群算法根据累计更新的信息素找出若干组解后，再由粒子群算法进行交叉、变异操作，得到更有效的解。与模拟退火算法、标准遗传算法和标准蚁群算法进行比较，效率很高。

（7）对求解最短路问题的蚁群算法的收敛性进行了探索性分析，提出的定理给出了寻找最短路的蚁群算法收敛的充分条件。

由于该算法出现的时间不长，对其研究还远不像其它启发式算法那样形成系统的理论，其相关数学分析还比较薄弱。算法中参数的选择更多的是依靠实验和经验，没有理论来指导，且计算时间偏长，容易出现停滞现象，这些都表明该算法在理论和实践方面尚有许多问题需要更深入的研究与解决。在今后的工作中，以下几个方面有待进一步的探索和研究。

（1）进一步研究真实蚁群（包括其它群居动物）的行为特征。因为蚁群算法是受蚁群觅食行为的启发而发展起来的一种模拟进化算法，通过对真实蚁群的深入研究有助于进一步改进蚁群算法。

（2）进一步研究算法的收敛性。目前，蚁群的收敛性证明还很不完善，给出更强的收敛性证明并得出收敛速度的估计将会加速算法的发展。

（3）进一步改进蚁群算法，提高其收敛速度。算法的收敛速度一直是人们关注的问题，虽然改进的蚁群算法的收敛速度有了很大的提高，但对于求解大规模的优化问题还不是很理想。

（4）进一步研究蚁群算法的理论性分析和各种参数设置问题。蚁群算法算法的数学理论基础相对薄弱，算法中涉及各种参数设置的一直没有确切的理论依据，通常都是按照经验型方法确定，对具体问题和应用环境的依赖性比较大。

（5）研究蚁群的并行实现。蚁群本质上的并行性为其并行实现提供了坚实的基础，如何实现蚁群之间的通讯以及蚁群的调度是并行实现的关键。

（6）进一步研究与其它算法（先进技术）的混合。实验结果显示，蚁群算法与

其它算法（先进技术）的结合有助于改善其自身或相应技术方法的性能。可以尝试与其它算法（先进技术）的混合，如免疫算法、禁忌搜索算法、神经网络、模糊逻辑和支持向量机等。

总之，对蚁群的研究还处于起步阶段，算法中的许多问题有待进一步研究。

致谢

本文的研究是在导师杨静宇教授的悉心指导下完成的。杨老师宽广的胸怀、严谨的治学态度、深厚的学术功底、敏锐的洞察力以及平易近人的工作风格都表现了当代学者的大家风范，这一切都使我受益非浅，终身难忘。在论文完成之际，谨向我的导师杨静宇教授表示诚挚的谢意和崇高的敬意！

感谢603教研室的夏德深教授、任明武副教授、朱近副教授、赵建副教授、陆建峰副教授、於东军博士等老师的关心和支持。感谢陈伏兵、陈科、吴陈、郑宇杰、王卫东、张生亮等博士的帮助。

感谢江苏科技大学的吴小俊教授、刘同明教授、张再跃教授、邓志良教授、韩斌副教授等老师在三年的博士学习期间从不同方面给予过的关心和帮助。感谢江南大学的王士同教授在学业上给予的关心和帮助。

感谢我的父母对我多年的养育之恩，是他们在精神上和物质上给我以关心和帮助，使我能够顺利完成学业。感谢我的岳父岳母在生活上和学业上对我的关心和支持。

最后，感谢我的妻子和儿子，是他们在精神、学业和生活上给我全力的关心和鼓励，我能够顺利完成学业和他们的无私奉献是分不开的。

感谢所有帮助过我的人们。

参考文献

1. Colorni A, Dorigo M, Maniezzo V. An investigation of some properties of an ant algorithm[A]. Proc. Of the Parallel Problem Solving from Nature Conference(PPSN' 92)[C].Brussels, Belgium:Elsevier Publishing, 1992, 509-520
2. 张纪会,徐心和.具有变异特征的蚁群算法[J].计算机研究与发展,1999,36(10):1240-1245
3. 马良,项培军.蚂蚁算法在组合优化中的应用[J].管理科学学报,2001,4(2):32-37
4. 彭喜元,彭宇,戴毓丰.群智能理论及应用[J].电子学报,2003,31(12A):1982-1988
5. Clolrni A, Dorigo M, Maniezzo V. Distributed optimization by ant colonies[A]. Proceedings of the First European Conference of Artificial Life(ECAL' 91)[C]. Elsevier, 1991:134-142
6. M Dorigo, V Maniezzo, A Colorni. The ant system: optimization by a colony of cooperating agents[J]. IEEE Transactions on System, Man and Cybernetics PartB, 1996, 26(1):29-42
7. L M Gambardella, M Dorigo. Solving symmetric and asymmetric TSPs by ant colonies [A]. Proceedings of the IEEE International Conference on Evolutionary Computation(ICEC' 96)[C]. Nagoya Japan, 1996:622-627
8. M Dorigo, L M Gambardella. A study of some properties of Ant Q [A]. Proceedings of PPSN IV Fourth International Conference on Parallel Problem Solving From Nature[C].Springer, Berlin, 1996:656-665
9. Stutzle T, Hhoos H. The MAX MIN ant system and local search for the traveling salesman problem [A]. In Proceedings of the IEEE International Conference on Evolutionary Computation(ICEC' 97)[C].Indianapolis, USA, 1997:309-314
10. Bernd Bullnheimer, Richard F Hartl, Christine Straub. A new rank based version of the ant system a computational study [DB/OL]. http://www.wu.wien.ac.at/am
11. Caro G Di, Dorigo M. AntNet: Distributed stigmergetic control for communications networks [J]. Journal of Artificial Intelligence Research(JAIR), 1998, 9:317-365
12. Lianyuan Li, Zemin Liu, Zheng Zhou. A new dynamic distributed routing algorithm on telecommunication networks [A]. International Conference on

Communication Technology Proceedings[C]. Beijing China, 2000.1:849-852

13. Gunes M, Sorges U, Bouazizi I. ARA the ant colony based routing algorithm for MANETs[A]. Proceedings International Conference on Parallel Processing Workshops[C]. Uuncouver, B C, Canada, 2002:79-85
14. DiCaro, G Dorigo M. Mobile agents for adaptive routing[A]. Proceedings of the Thirty First Hawaii International Conference on System Sciences[C]. Kohala Coast, HI USA, 6-9 Jan 1998.7:74-83
15. Lumer E, Faieta B. Diversity and adaptation in populations of clustering ants[A]. Proc of the 3 Conf On Simulation of Adaptive Behavior[C]. MIT Press, 1994:499-508
16. Parpinelli R S, Lopes H S, Freitas. Data mining with an ant colony optimization algorithm[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(4):321-332
17. Maniezzo V, Colorni A, Dorigo M. The Ant System Applied to the Quadratic Assignment Problem[R]. Tech Rep IRIDIA/94 28, Universit Libre de Bruxelles, Belgium. 1994
18. Talbi E G, Roux O, Fonlupt, C Robillard D. Parallel ant colonies for the quadratic assignment problem[J]. Future Generation Computer Systems, 2001, 17(4):441-449
19. Colorni A, Dorigo M, Maniezzo V, Trubian M. Ant system for job-shops cheduling[J]. Belgian Journal of Operations Research, Statistics and Computer Science, 1994, 34(1):39-53
20. Stutzle T. An ant approach to the flow shop problem[A]. Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing[C]. Aachen, Germany, 1997.3:1560-1564
21. Mc Mullen, Patrick R. An ant colony optimization approach to address ing a JIT sequencing problem with multiple objectives[J]. Artificial Intelligence in Engineering, 1998, 15(3):309-317
22. Lee Zne- Jung, Lee Chou-Yuan, Su Shun-Feng. An immunity based ant colony optimization algorithm for solving weapon-target assignment problem[J]. Applied Soft Computing Journal, 2002, 2(1):39-47
23. Bullnheimer B, R F Hartl, C Strauss. An improved ant system algorithm for the vehicle routing problem[DB/OL]. The 6th Viennese workshop on Optimal Control, Dynamic Games[C]. Nonlinear Dynamics and Adaptive Systems,

http://citeseer.nj.nec.com/, 1997-05-21

24. Maniezzo V, Carbonaro A. An ants heuristic for the frequency assignment problem[A]. Proceedings of MIC' 99[C].1999:927-935

25. De Campos, Luis M, Fernndez Luna, Juan M, Gumez Jos, APuerta, Jos M. Ant colony optimization for learning Bayesian networks [J]. International Journal of Approximate Reasoning, 2002, 31(3):291-311

26. Silva De A, Ramalh R M. Ant system for the set covering problem[A]. IEEE International Conference on Systems[C].Man, and Cybernetics, Tucson, AZ USA, 2001.5:3129-3133

27. 现代应用数学手册编委会.现代应用数学手册：运筹学与最优化理论卷[M].北京：清华大学出版社，1997：195-220.

28. 陈永忠，陈顺怀.整数规划的遗传算法[J].交通部上海船舶运输科学研究所学报,2000,23(1):42-46.

29. 丰建荣，刘志河，刘正和.混合整数规划问题遗传算法的研究及仿真实现[J].系统仿真学报，2004，16(4):845-848.

30. 谢云.用模拟退火算法并行求解整数规划问题[J].高技术通讯，1991,1(10):21-26.

31. 谭瑛,高慧敏,曾建潮.求解整数规划问题的微粒群算法[J].系统工程理论与实践，2004，24（5）：126-129.

32. 黄樟灿,吴方才,胡晓林.基于信息素的整数规划的演化求解[J].计算机应用研究，2001，18(7)：27-29.

33. 林锦，朱文兴.凸整数规划问题的混合蚁群算法[J].福州大学学报,1999，27（6）:5-9.

34. Wacholder E. A neural network-Based optimization algorithm for the static weapon-target assignment problem [J]. ORSA Journal on Computing,1989(4): 232-246

35. Lly S P. A algorithm for the weapon-target assignment problem [J].Defence Techique,1991(a):45-56

36. 王永寿译.武器-目标分配问题的一种算法.现代防御技术,1993,23（1）：13-23

37. 朱齐丹，胡绍勇，宋福香等.解武器-目标分配问题的神经网络方法.哈尔滨工业大学学报,1997（3）：36-41

38. 马振华等.运筹学与最优化理论卷[M].北京：清华大学出版社,1998:185-188

39. 王晓东.计算机算法设计与分析[M].北京：电子工业出版社，2001：102-104

40. 曹新谱.算法设计与分析[M].长沙：湖南科学技术出版社,1984：113-114

41. 刑文循，谢金星. 现代优化计算方法[M]. 北京：清华大学出版社, 1999：40-45
42. 康立山，谢云，尤矢勇等. 模拟退火算法[M]. 北京：科学出版社，1994：150-151.
43. 王凌. 智能优化算法及其应用[M]. 北京：清华大学出版社，2001:17-59
44. 曹晋华，程侃. 可靠性数学引论[M]. 北京：科学出版社，1986:70-74
45. Tillman F A , Hwang C L ,Kuo W. 系统可靠性最优化[M] . 刘炳章译. 北京：国防工业出版社，1988:5-15
46. 薛嘉庆. 最优化原理与方法[M]. 北京：冶金工业出版社，1983:315-330
47. 陈峻，沈洁，秦玲. 蚁群算法求解连续空间优化问题的一种方法[J]. 软件学报，2002，13 (12)：2317-2323
48. 汪镭，吴启迪. 蚁群算法在连续空间寻优问题求解中应用[J]. 控制与决策，2003，18 (1)：45-48
49. 黄振华，吴诚一. 模式识别[M]. 杭州：浙江大学出版社，1991：40-62
50. 蔡元龙. 模式识别[M]. 西安：西北电讯工业出版社, 1986：17-32
51. 谢维信. 工程模糊数学方法[M]. 西安：西安电子科技大学出版社，1991：142-160
52. 高国华，沈林成，常文森. 求解 TSP 问题的空间锐化模拟退火算法[J]. 自动化学报, 1999，25(3):425-428
53. Kirkpatrick S, Gelatt JR, Vecchi JR. Optimization by simulated annealing [J]. Science， 1983, 220:671-680
54. 谢胜利，唐敏，董金祥. 求解 TSP 问题的一种改进的遗传算法[J]. 计算机工程与应用, 2002，38 (8)：58-60
55. 张立明. 人工神经网络的模型及其应用[M]. 上海：复旦大学出版社, 1994：97-98
56. 高尚. 基于 MATLAB 遗传算法优化工具箱的优化计算[J]. 微型电脑应用，2002, 18(8):52-54
57. 丁建立，陈增强，袁著祉. 遗传算法与蚂蚁算法的融合[J]. 计算机研究与发展，2003，40 (9)：1351-1356
58. 丁建立，陈增强，袁著祉. 遗传算法与蚂蚁算法融合的马尔可夫收敛性分析[J]. 自动化学报，2004，30 (4)：629-634
59. 李兵, 蒋慰孙. 混沌优化方法及其应用[J] . 控制理论与应用, 1997, 14(4):613-615
60. 张国平, 王正欧, 袁国林. 求解一类组合优化问题的混沌搜索法[J]. 系统工程理论与实践，2001，21 (5)：102-105
61. 唐巍, 郭镇明, 唐嘉亨等. 复杂函数优化的混沌遗传算法[J] . 哈尔滨工程大学学报，2000，21 (5)：1-5
62. 花栅. 计算机科学数学[M]. 哈尔滨：哈尔滨船舶工业学院出版社，1994：97-101
63. Eberhart R. C.， Kennedy J. A New Optimizer Using Particles Swarm Theory[C].

Proc. Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995:30-43

64. Shi Y. H., Eberhart R. C.. A Modified Particle Swarm Optimizer[C]. IEEE International Conference on Evolutionary Computation, Anchorage, Alaska, May 4-9, 1998:69-73

65. 李爱国，覃征，鲍复民等. 粒子群优化算法[J]. 计算机工程与应用,2002,38 (21) :1-3

66. 杨维，李歧强. 粒子群优化算法综述[J] . 中国工程科学，2004，6 (5)：87-94

67. 金慧敏，马良. 遗传退火进化算法在背包问题中的应用[J]. 上海理工大学学报，2004，26(6). 561-564

68. 马良，王龙德. 背包问题的蚂蚁优化算法[J]. 计算机应用，2001，21(8):4-5

69. 于永新，张新荣. 基于蚁群系统的多选择背包问题优化算法[J]. 计算机工程,2003,29(20). 75-76,84

70. 柯晶，钱积新，乔谊正.一种改进粒子群优化算法[J]. 电路与系统学报，2003，8 (5)：87-91

71. 吕振肃，候志荣.自适应变异的粒子群优化算法[J]. 电子学报，2004，32 (3)：416-420

72. 谢晓锋，张文俊，杨之廉.微粒群算法综述[J]. 控制与决策，2003，18(2)：129-133

73. 李炳宇，萧蕴诗，汪镭.一种求解高维复杂函数优化问题的混合粒子群优化算法[J]，信息与控制，2004，33 (1)：27-30

74. 高鹰，谢胜利. 免疫粒子群优化算法[J]. 计算机工程与应用，2004,40 (6)：4-6

75. 姚俊峰，梅炽，彭小奇等.混沌遗传算法及其应用[J]. 系统工程，2001，19 (1)：70-74

76. 王子才，张彤.基于混沌变量的模拟退火优化方法[J]. 控制与决策，1999，14(4)：382-384

77. 杨若黎，顾基发. 一种高效的模拟退火全局优化算法[J] . 系统工程理论与实践，1997，17 (5)：29-35

78. Gutjahr W J. A graph-based ant system and its convergence[J]. Future Generation Computer System,2000,16(8):873-888

79. Thomas Stuezle, Dorigo Macro. A Short Convergence Proof for a Class of Ant Colony Optimization Algorithms[J]. IEEE Trans. on Evolutionary Computation, 2002, 6(4): 358-365

80. 孙焘,王秀坤，刘业欣,等. 一种简单蚂蚁算法及其收敛性分析[J]. 小型微型计算机系统，2003，24 (8) : 1524-1526

81. 段海滨,王道波.蚁群算法的全局收敛性研究及改进[J]. 系统工程与电子技

术,2004，26（10）：1506-1509
82. 吴启迪，汪镭. 智能蚁群算法及其应用[M]. 上海：上海科技教育出版社，2004
83. 李天成. 应用智能蚂蚁算法解决旅行商问题[D]. [硕士论文]. 厦门：厦门大学，2002
84. 赵强蚁. 群算法在中压城市配电网规划中的应用[D]. [硕士论文]. 成都：四川大学，2003
85. 沈彬. 改进蚁群算法在物流配送中的应用研究[D]. [硕士论文]. 杭州：浙江大学，2004
86. 王笑蓉. 蚁群优化的理论模型及在生产调度中的应用研究[D]. [博士论文]. 杭州：浙江大学，2003
87. 胡小兵. 蚁群优化原理、理论及其应用研究[D]. [博士论文]. 重庆：重庆大学，200
88. 闻育. 复杂多阶段动态决策的蚁群优化方法及其在交通系统控制中的应用[D]. [博士论文]. 杭州：浙江大学，2004

附 录
攻读博士学位期间发表（含录用）的学术论文

1. 高尚，杨静宇，吴小俊. 圆排列问题的蚁群模拟退火算法. 系统工程理论与实践，2004，24（8）
2. 高尚. 解旅行商问题的混沌蚁群算法. 系统工程理论与实践，2005，25（9）
3. 高尚，韩斌，吴小俊，杨静宇. 求解旅行商问题的混合粒子群优化算法. 控制与决策，2004，19(11)（EI Compendex检索05048803654）
4. 高尚，杨静宇. 混沌粒子群优化算法研究. 模式识别与人工智能，（已录用）
5. 高尚，杨静宇. 背包问题的混合粒子群优化算法. 中国工程科学，（已录用）
6. 高尚. 武器-目标分配的蚁群算法. 计算机工程与应用，2003，39（3）
7. 高尚，杨静宇，吴小俊. 聚类问题的蚁群算法. 计算机工程与应用，2004，40（8）
8. 高尚，杨静宇，吴小俊. 求解指派问题的交叉粒子群优化算法. 计算机工程与应用，2004，40（8）
9. 高尚，杨静宇，吴小俊. 可靠性优化的蚁群算法. 计算机应用与软件，2004，21（12）
10. 高尚，杨静宇，吴小俊，刘同明. 基于模拟退火算法思想的粒子群优化算法. 计算机应用与软件，2005，22（1）
11. 高尚. 基于Rough 集理论和神经网络的武器系统参数费用模型. 系统工程理论与实践. 2003, 23(4)
12. 高尚，杨静宇. 武器-目标分配问题的粒子群优化算法. 系统工程与电子技术，2005，27（7）
13. 高尚，钟娟，莫述军. 连续优化问题的蚁群算法研究. 微机发展，2003，13（1）
14. 高尚，钟娟，莫述军. 多处理机调度问题的蚁群算法. 微型电脑应用. 2003，19（4）
15. 高尚，杨静宇. 非线性整数规划的蚁群算法. 2005年全国自动化新技术会议. 南京理工大学学报. 2005，29（5A）
16. 高尚，候志远. 集合划分问题的粒子群优化算法. 华东船舶工业学院学报, 2005, 19(6)
17. 高尚. 求解旅行商问题的模拟退火算法. 华东船舶工业学院学报，2003，17(3)
18. GAO Shang, YANG Jing-yu, WU Xiao-jun. The Ant Colony Algorithm for Clustering Problems. The proceedings of the China Association for Science and Technology. 2005, Vol.2
19. 高尚，杨静宇. 多处理机调度问题的粒子群优化算法. 计算机工程与应用，2005，41(27)

20. 高尚，杨静宇.求解聚类问题的混合粒子群优化算法.科学技术与工程，2005，5(23)
21. 高尚，杨静宇.最短路的蚁群算法收敛性分析.科学技术与工程，2006，6(3)（已录用）