基于细粒度模型的并行蚁群优化算法

朱海梅，朱庆保
（南京师范大学 计算机系，江苏 南京 210097）

摘　要：蚁群优化算法的研究和应用已取得了不少重要成果，然而在大规模优化应用中还存在搜索时间长的问题，为此研究了一种基于细粒度模型的并行蚁群算法。实验结果表明，该算法与最新的改进算法相比，搜索速度提高数十倍至数百倍以上。

关键词：蚁群优化算法；蚁群系统；并行算法；细粒度模型；TSP 问题
中图法分类号：TP301.6　　**文献标识码**：A　　**文章编号**：1001-3695(2004)11-0059-03

Ant Colony Optimization Parallel Algorithm Based on Fine-grained Model

ZHU Hai-mei, ZHU Qing-bao
(*Dept. of Computer Science, Nanjing Normal University, Nanjing Jiangsu* 210097, *China*)

Abstract: Although a good many important results have been achieved about the research of Ant Colony Optimization (ACO) algorithm and its application, the time required to find good optimal solution is unbearable for large scale optimization problems. Therefore, ant colony optimization parallel algorithm base on fine-grained model is proposed to improve its performance in this paper. The results of experiment show that the algorithm described in this paper make the searching speed hundreds of times faster than the latest improved one.

Key words: Ant Colony Optimization Algorithm; Ant Colony System; Parallel Algorithm; Fine-grained Model; TSP Problems

1　引言

蚁群优化算法是模拟自然界中真实蚁群的觅食行为而形成的一种模拟进化算法，是由意大利学者 M. Dorigo 等人[1~3]于 1990 年提出的，当时称之为蚂蚁系统(Ant System, AS)。用该方法求解旅行商问题(Traveling Salesman Problem, TSP)、资源二次分配问题、作业调度问题等，取得了一系列较好的实验结果。受其影响，蚁群系统模型逐渐引起了其他研究者的注意，并用该方法求解一些实际问题[4~6]。近几年，在欧、美召开的有关国际学术会议上，蚁群优化算法已成为研究热点之一。

对于蚁群优化算法，M. Dorigo 等人先后提出了 AS[2]，ACS (Ant Colony System)[3] 两个版本。1999 年，M. Dorigo 等人把 AS, ACS 等算法纳入到统一的组合优化框架，称为蚁群优化(Ant Colony Optimization, ACO)[7]。该文虽然提出了不同于 ACS 的 ACO 算法，但没有给出较详细的算法描述，也没有给出实验数据，它所提出的 ACO 算法的实验表明，该算法并不比 ACS 优越。因此，目前国际上广泛流行的 ACO 算法主要引用了文献[7]给出的 ACO 概念，而不是文献[7]提出的 ACO 算法，采用的算法主要是 AS, ACS 或其改进版[8~10]。

十多年的研究结果已经表明：蚁群优化算法用于组合优化具有很强的发现较好解的能力，具有分布式计算、易于与其他方法相结合、鲁棒性强等优点。蚁群优化算法在动态环境下也表现出高度的灵活性和健壮性，如在电信路由控制方面的应用被认为是目前较好的算法之一[6]。然而，蚁群算法存在搜索时间长、易于停滞(搜索到一定程度后，所有个体所发现的解完全一致，不能对空间进一步搜索)。针对这些缺陷，不少学者提出了改进算法[11~13]。例如，文献[11]提出了一种最大、最小算法，其基本思想是对路径上的信息素进行限制，以克服停滞问题，并且仅让每一代中的最好个体所走路径上的信息作调整，以加快收敛速度。文献[12]则提出了一种新的信息素更新策略：①局部信息素修改时，挥发系数动态改变；②全局信息素更新时，将蚂蚁所走的较短的那些路径上的信息加强，而较差的那些路径上的信息减弱。文献[13]提出了一种基于蚂蚁进化规则的算法。还有不少其他改进文献，例如引入交叉算子以提高搜索多样性；引入分支因子 r 作为衡量群体多样性的指标，当 r 低于某一值时，便对各路径上的信息作动态调整，以期望避免过早出现停滞现象等。这些研究对算法有一定程度的改进，但对提高搜索速度效果不是特别明显。在大规模优化问题上，速度慢仍然是困扰蚁群算法应用的一大难题。为了解决这一难题，笔者研究了基于细粒度模型的并行蚁群优化算法。实验结果表明，该算法与最新的其他改进算法相比，搜索速度提高数十倍至数百倍以上。这种算法未检索到国内外相关报道资料。

2　蚁群算法的基本原理与问题描述

2.1　蚁群算法的基本原理

自然界中蚁群觅食要经若干条路径从蚁穴到达食物源，最终所有蚂蚁选择了一条最短路径进行觅食。为什么蚂蚁能找到最短路径呢？经研究发现，蚂蚁在运动过程中能够在所经过

收稿日期：2003-11-11；**修返日期**：2004-03-09

的路径上留下一种称为信息素的物质,而且蚂蚁在运动过程中能够感知这种物质的存在及其强度,并以此指导自己的运动方向,它们倾向于朝着该物质强度高的方向移动。因此,由大量蚂蚁组成的集体行为便表现出一种信息正反馈现象:某一路径越短,该路径上走过的蚂蚁就越多,则留下的信息素强度就越强,后来者选择该路径的概率就越大。蚂蚁个体之间就是通过这种信息交流来选择最短路径并达到搜索食物的目的。蚁群算法就是模拟蚁群这一觅食行为的一种优化算法。

2.2 问题描述与定义

由于蚁群算法是模拟蚂蚁觅食行为的一种优化算法,与TSP问题比较贴近,且有TSP问题库进行比较,因此,很多对蚁群算法的研究大多以TSP问题作为衡量算法优劣的标准或范例。因此,本文也基于TSP问题,一方面,本文方法可用于求解大规模TSP问题;另一方面,其思想方法可推广到其他优化问题。为了叙述简便,首先作出如下定义:

记 AS 为二维平面上的凸多边形有限区域,其内部分布着 n 个城市,令 $C=\{c_1,c_2,\cdots,c_n\}$ 为 n 个城市的集合,$R=\{1,2,\cdots,n\}$ 为城市的下标集或序号集,在 AS 中建立直角坐标系 $\sum_0$,则 $c_i\in C,i\in R$。在 $\sum_0$ 都有确定的坐标 (x_i,y_i),记作 $c_i(x_i,y_i)$。从图的角度,n 个城市构成了顶点集,各城市在AS中的连线构成边集,借此概念,称城市 $i\in R$ 为节点,任意两城市间连线为边,记作 $e_{ij},i,j\in R$。

定义1 $d(c_i,c_j)$ 为任意两个城市 c_i,c_j 之间的距离或边长,$i,j\in R$,简记作 d_{ij},且满足 $d_{ij}=d_{ji}$,d_{ij} 由式(1)计算;距离 d_{ij} 又可以记作边长 d_l。

$$d(c_i\cdot c_j)=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2} \tag{1}$$

很显然,任意两个城市间的距离构成了一个距离矩阵。

定义2 $ant=\{1,2,\cdots,k,\cdots,m\}$ 表示所有蚂蚁的集合,$k\in ant$ 表示某只蚂蚁,$\tau_{ij}(t)$ 表示蚂蚁在 t 时刻残留在 $e_{ij}(i,j\in R)$ 上的信息量。

定义3 设蚂蚁 k 在 t_0 时刻从节点 i 出发,遍历所有节点后,t_e 时刻返回节点 i,任意时刻所处的节点位置记作 $P(t_i)$。令 $\text{tabu}_k=\{P(t_0),P(t_1),\cdots,P(t_j)\}$ 为蚂蚁 k 从 t_0 时刻到 t_j 时刻已走节点位置的集合(等价于已走节点集合),t_{j+1} 时刻,$\forall P(t_{j+1})\in C$ 且 $\forall P(t_{j+1})\in \text{tabu}_k$,则称 $\forall P(t_{j+1})$ 为 t_{j+1} 时刻禁入点。

很显然,tabu_k 是第 k 只蚂蚁已走位置的集合,它随着蚂蚁的行走动态调整。按该定义,这些位置不允许再走,因此,称 tabu_k 为禁忌表。Tabu_k 中各节点位置在 AS 中的连线称为路程,路程长度记作L,用式(2)计算,其中 d_l 由式(1)计算:

$$L=\sum_{l=1}^{e+1}d_l \quad d_l=d(c_i,c_j),c_i,c_j\in C,i,j\in R \tag{2}$$

定义4 $BR_i(c_i(x_i,y_i))=\{c\mid c\in C,d(c,c_i)\leqslant d_{\min}\},i\in R$ 称 c_i 的邻居节点集。其中,$d_{\min}$ 是根据具体问题和所需邻居集大小设定的距离阈值。

定义5 $\eta_{ij}=1/d(c_i,c_j)$ 称蚂蚁从节点 i 选择节点 j 的启发函数。

3 基于细粒度并行模型的蚁群优化算法

3.1 并行性分析

从ACS算法的原理步骤可以看出,每只蚂蚁沿所有城市周游一周的过程是完全独立的,选择下一个城市的概率计算、修改局部信息素也可以独立进行。基于这种并行性,完全可以将 m 只蚂蚁分配在 m 个处理机上同时进行并行搜索。这种并行处理可以在以全局存储器为中心的并行机上进行,也可用小规模专用局域网实现。

3.2 算法的改进策略

考察TSP问题,商人要遍历 n 个城市(每个城市只走一次),且使总路径最短。在这一过程中有两个值得注意的问题:①对于一个有很多城市的地区,当商人从周围不同的边部城市出发时,根据ACS算法,从当前城市选择下一个城市的概率大小主要依据信息素和两城市间的距离长短,因此,他所选择的路径不相同的概率是很大的。也就是说,商人最终得到的最短路径可能是相同的,但从不同边部出发点所走的路径过程可能是不同的。②在一个较复杂的、有若干城市的地图上,在一条遍历所有城市的最短路径中,城市 i 连接到城市 j,j 不可能是离城市 i 最远的那些城市。根据第①点,可以将 m 只蚂蚁均匀地分配在 $m(m<n)$ 个边部城市并同时分配到 m 个处理机上,由 m 只蚂蚁(m 个处理机)完成从不同的边部城市为出发点的搜索。由于不同出发点构成的路径不同,各蚂蚁重复搜索的概率较小,增加了搜索的多样性,可以加快搜索速度。

在ACS算法中,当 k 从城市 i 选择下一个城市 j 时,需将 $n-1$ 个城市与禁忌表比较,再计算 $n-1$ 个城市的转移概率,需较长的计算时间。根据上述第②点,将下一个城市的选择范围仅限于离当前城市 i 较近的部分城市,仅对这部分城市的转移概率进行计算即可。当城市的选择范围限于 n/w 个城市时,这一计算过程的计算速度可提高近 w 倍。

由于蚂蚁间的相互联系和对搜索的重要贡献是留下信息素,信息素的更新策略也是决定收敛速度的关键之一。

综合以上几点,研究了一种基于求解TSP问题的细粒度模型蚁群优化算法。

3.3 细粒度模型蚁群优化算法

根据第2节定义和上述策略,细粒度模型蚁群优化算法步骤如下(因各蚂蚁算法相同,以下以一只蚂蚁的算法为例):

(1)初始化。将 n 个城市分别按 x,y 坐标排序,均匀地在地图周围找出 m 个边部城市($m<n$),将 m 只蚂蚁分别固定的放置在 m 个边部城市上(每个城市放置一只蚂蚁,并分配到一个分处理器),并将出发城市设置到 tabu_k 中。

以 n 个城市分别为起点按距离排序,从而按定义4确定 n 个节点的邻居节点集 $BR_i,i=1,2,\cdots,n$,可取邻居节点集的邻居数 $|BR_i|=n/w$,$w\in[1,30]$,n 越大,w 取值可越大;若 n 较小时,w 取较小的值。设置代数计数器 NC=MAX,并设定其他参数的初值。

(2)$\forall k$,以当前城市 i 为中心,按最近邻居原则选择下一个节点 $j\in R,j\in BR_i$。首先从 BR_i 个城市中找出 z 个未走过的城市 $\{j_1,j_2,\cdots,j_p,j_z\}$,即 $j_p\notin \text{tabu}_k$。

(3)在 z 个城市中,按式(3)或式(4)选择节点 j:

$$j=\begin{cases}\underset{j\notin \text{tabu}_k}{\operatorname{argmax}}\{[\tau_{ij}(t)][\eta_{ij}(t)]^{\beta} & \text{if} \quad q\leqslant q_0\\ S & \text{otherwise}\end{cases} \tag{3}$$

式(3)中,$0<q_0\leqslant 1$,是初始设定的参数;q 是一个随机数,$q\in(0,1)$;S 是根据式(4)决定的随机变量。

$$p_{ij}^{k}=\begin{cases}\dfrac{[\tau_{ij}(t)^{\alpha}][\eta_{ij}]^{\beta}}{\sum_{k\notin tabu_k}[\tau_{ik}(t)]^{\alpha}[\eta_{ik}]^{\beta}} & j\notin tabu_k\\ 0 & j\in tabu_k\end{cases}\quad(4)$$

式中,$p_{ij}^{k}(t)$表示在 t 时刻蚂蚁 k 由节点 i 转移到节点 j 的概率;α 表示在边 e_{ij} 上残留信息的重要程度;β 表示启发信息的重要程度。当 $q>q_0$ 时,计算 z 个城市的转移概率 p_{ij}^{k},并根据赌轮盘规则选择城市 j。

将 j 加入禁忌表 $tabu_k$。

(4)局部信息素自适应更新。随着时间的推移,以前留下的信息逐渐消逝,用参数 1-ρ 表示信息消逝程度,$k\in ant$ 走完一个边后,即按式(5)进行局部信息更新。

$$\tau_{ij}^{new}=(1-\rho)\tau_{ij}^{old}+\rho\Delta\tau_{ij}\quad(5)$$

$$\Delta\tau_{ij}=\frac{Q_1}{L_z}$$

$$L_z=\sum_{l_1}^{h}d_{l_1}^{1}+\sum_{l_2}^{h}d_{l_2}^{2}+\cdots+\sum_{l_m}^{h}d_{l_m}^{m}=\sum_{k=1}^{m}\sum_{l=1}^{h}d_{l}^{k}$$

$$l=1,2,\cdots,h,\ \forall i,j\in R$$

式(5)中,Q_1 为一个较大的常数,h 是 k 在本次周游中已走过的城市边数;l_l 是 k 已走过的边的边长,由式(1)计算,l_z 就是所有蚂蚁在本次周游中已走过的边的累加总长。

该策略的主要思想是:根据实验,当 k 从一个具有很多城市的复杂地图的边部城市出发时,它在出发点附近的局部区域走的路径往往是较优路径的一部分,走到地图中心区,路径错综复杂,与其他蚂蚁走的路径重叠增多,互相影响加大,走出较优路径的概率降低。根据这一实验结果,为避免中心区信息素堆积过多,以及体现多蚂蚁的共同影响,每只从边部城市出发的蚂蚁留下的信息素应随着向中心区延伸而逐渐减弱,当走到最远处时,已渗透到另一边部其他蚂蚁的领地,为了不过大的干扰其他蚂蚁开始走出的较优结果,其信息素应减到最小。该策略保证了各蚂蚁所留信息素的均衡性,保证了对搜索的均衡贡献和相互协作,体现出了群体的力量,这是本文算法能大幅度提高收敛速度的关键之一。

(5)信息素通信。$\forall k\in ant$ 走完一个边并更新完信息素后即将信息素传给服务器,由服务器将更新后的信息素传给其他 $m-1$ 个分处理器。

$\forall k$ 选择完节点 j 后,令 $i_{new}=j;j_{new}=j_{old}+1$;返回步骤(2)开始选择下一个节点,直到所有蚂蚁周游完一周。

(6)$\forall k$ 周游完一周后,用式(2)计算路程长度 L_k,并将 L_k 及路径表传送给服务器,由服务器找出 L_{kmin} 并保存,$L_{kmin}=\min L_k$,$k\in ant$。

(7)由服务器仅对最佳路径上的信息素按式(6)进行全局信息素更新,并将更新后的信息素传送给分处理器。

$$\tau_{ij}^{new}=(1-\alpha)\tau_{ij}^{old}+\alpha\Delta\tau_{ij}^{k}\quad(6)$$

$$\Delta\tau_{ij}^{k}=\begin{cases}\sum_{l=1}^{n}\dfrac{Q}{d_l} & \text{if } l\in \text{global-best-tour}\\ 0 & \text{otherwise}\end{cases}$$

式(6)中,Q 为常数,$l\in$ global-best-tour,表示蚂蚁 k 所走的城市边属于最佳路径,α 为全局信息素挥发系数。

(8)服务器将本次周游得到的与 L_{kmin} 已得到的最优长度 l_d 比较,若有 $l_{kmin}<l_d$ 则用 l_{kmin} 替换 l_d,同时替换最优路径表。

(9)设置的计数值 NC-1 不等于 0,清空并初始化禁忌表。重复上述过程,直到 NC-1=0 为止。

4　实验比较

为了验证、比较算法的效果,笔者从 TSPLIB 下载了多个著名的 TSP 问题的范例进行仿真实验,都取得了非常好的结果。为了与最新文献的算法进行比较,选了几种与文献[12,13]相同 TSP 问题的实验结果列入本文。由于对各参数的选择还没有指导理论,因此本实验的参数均通过实验确定。

图 1 是用不同方法优化 Eil76 TSP 问题的收敛特性比较。其中,横坐标为蚂蚁周游次数,纵坐标为蚂蚁周游取得的最佳路径长度。曲线 1 为本算法收敛特性,曲线 2 为 ACS 算法的收敛特性,曲线 3 为文献[13]的改进算法结果。曲线 1、曲线 2 摘自文献[13]。实验结果表明,ACS 算法和文献[13]算法均需周游 2 600 次以上还不能收敛到较优值,本文算法仅需 55 次即收敛到最优值。

图 2 示出了用不同算法优化 Kroa100 TSP 问题的收敛特性比较。曲线 1 为本算法收敛特性,曲线 2 为 ACS 算法的收敛特性,曲线 3 为文献[12]的改进算法结果。曲线 1、曲线 2 摘自文献[12]。从图 2 中可见,本文算法仅需 469 次即收敛到最优值,ACS 算法 30 000 次以上还不能收敛到最优值,文献[12]改进算法需 30 000 次左右才收敛到较优值。

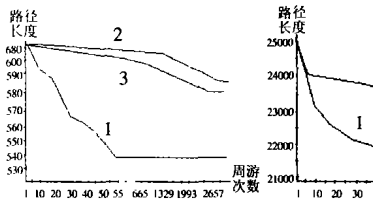

图 1　求解 Ei176 收敛特性比较

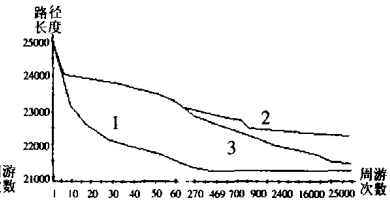

图 2　求解 Kroa100 收敛特性比较

为了更直观地进行数字比较,把实验结果比较总结于表 1。

表 1　最优解结果比较

	TSPLIB 最优结果	本文优化结果	ACS 优化结果	文献[12]算法结果	文献[13]算法结果
Eil76	538	538	590	无	585
Kroa100	21 282	21 282	22 469	21 300	22 389

表 1 中,TSPLIB 最优结果指网上 TSP 问题库公布的最优结果。表 2 示出了收敛代数的实验比较(m 只蚂蚁各周游一次称一代)。

表 2　收敛代数实验结果比较

最优解代数	同结果代数	ACS 最优代数	文献[12]最优代数	文献[13]最优代数	最小倍数	同比倍数	
Eil76	55	21	>3 000	无	>3 000	>54	>142
Kroa100	470	270	>30 000	>25 000	>30 000	>53	>80

表 2 中,最优解代数是指本文算法达到最优解时的蚂蚁周游代数;同结果代数是指本算法达到 ACS 或文献[12,13]中最好结果时所用代数;最小倍数是指其他算法最少代数与本算法最优解代数之比;同比倍数是指是与同结果代数之比,是指其他算法无该项实验。

从表 1 和表 2 所示的实验对比可见,本算法结果与 TSBLIB 给出的最优结果一致,优于最新的改进算法,收敛速度大幅提高,蚂蚁周游代数减少数十倍至上百倍。

5　结束语

提出了几种改进策略,并用并行处理模型使优化的收敛速度大幅提高。解决了蚁群优化算法速度慢这一瓶颈问题,使其用于大规模优化成为可能。本文虽以 TSP 问题(下转第 126 页)

循环过程中若落入局部最优的"陷阱"时要采用静态爬山方法。

(1)交配。从整个任务分配方案集中以一定的百分比 P_C 随机选择出一个供交叉的子集,利用它进行交叉繁殖。具体实现方式是对 TA.S 中的某些节进行重组,仍以三个 Agent 和七个任务为例,设有两个父代分配方案 TA1,TA2,其中TA1.S = 00000001011010,TA1.R[1]:1→2→3,TA1.R[2]:4→5,TA1.R[3]:6→7;TA2.S = 00011000100001,TA2.R[1]:1→4→6,TA2.R[2]:2→7,TA2.R[3]:3→5。在 TA1.S 中随机选定几节替换到 TA2.S 中就形成了一个新的二进制串 S′,假如选定 TA1.S 中的第 1、第 4、第 5 节(从左到右),替换后 S′ = 00011001010001,同时,为保持数组 R 的一致性,从 TA1.R 作相应调整,调整有几种方案,从中选出一种或几种有效分配方案,就得到下一代的一组分配方案。

(2)变异。除了交叉之外,还应对任务分配方案集以某个较小的比例 P_k 选出一个子集进行变异。变异的方式有多种,可以是对 TA.S 中的某些节求反,也可以是互换一方案中几个 Agent 所处理的任务,还可以是其他方式。由于变异不会造成个体数量变化,所以每次变异后对一种原方案只保留一种有效变异方案。

(3)复制。对于没有进行交叉和变异的方案,直接加入到新的任务分配方案集中。

(4)选择。交叉和变异产生的方案称为新方案。计算新方案的目标函数 cost′,并令 Δcost = cost - cost′,若 Δcost < 0,表明新方案优于原方案,将新方案加入到新任务分配方案集中;若 Δcost ≥ 0 表明原方案优于新方案,此时计算概率值 $p = e^{\frac{\Delta cost}{temp}}$,temp 是当前温度。由系统产生一个(0,1)上的随机数 r,若 p ≤ r,则放弃新方案,若 p > r,则将新方案加入到新任务分配方案集中。我们可以对 p 算式作一个简单分析:在一定温度进行方案选择时,temp 也可看作常数,所以此时 p 只与 Δcost 有关,而且随 Δcost 递增而递减。Δcost 越小,表明新方案虽次于原方案,但接近原方案,因而是较"好"的方案,而此时 p 也就越大,p > r 成立的可能性也越大,越容易被接受;反之同理。

可以看出,选择过程选取了两类新方案:①优于原方案的新方案;②次于原方案但目标函数比较接近原方案的新方案。从而使新方案集的整体水平优于原方案集。

5 结束语

本文综合了遗传算法和模拟退火算法,提出在基于多 Agent 的汉字签名认证系统的任务分配新算法。由于遗传算法和模拟退火算法的独特性,使得本文所给出的任务分配算法有区别于传统方法的新特点。但是目标函数的选择、初始任务分配方案集选多大为宜是值得进一步研究的问题。

参考文献:

[1] Gerhard Weiss. Multi-Agent Systems:A Modern Approach to Distributed Artificial Intelligence[M]. MIT Prus,1999.

[2] J H Holland. Adaptation in Natural and Artificial System[M]. Michigan:University of Michigan Press,Ann Ardor,1975.

[3] S Kirkpatrick,C D Gellatt,Jr M P Vecchi. Optimization by Simulated Annealing[J]. Science,1983,(5).

[4] D Fitoussi,M Fermenhalts. Choosing Social Laws for Multi-Agent Systems,Minimulity and Simplicity[J]. Artificial Intelligence,2000,119(1-2):61-101.

[5] 邵斌,王国钧. 多 Agent 系统的性能评价[J]. 微电子学与计算机,2003,20(8):80-81.

[6] 周昌乐. 手写汉字的机器识别[M]. 北京:科学出版社,1997.

作者简介:

邵斌(1971-),男,浙江湖州人,系副主任,讲师,在读硕士,研究方向为人工智能、中文信息处理;严智敏(1971-),女,浙江湖州人,讲师,学士,研究方向为形式逻辑、算法设计;王国钧(1946-),男,浙江湖州人,副教授,研究方向为算法设计与分析、图像处理。

(上接第 61 页)为例,但其思想方法可用于其他优化问题。

然而,蚁群算法是一种新的模拟进化算法,其研究才开始不久,还没有形成系统分析的方法和坚实的数学基础,各种实验参数的确定也没有理论指导。目前国际上的诸多研究成果还都是基于实验分析,因此,还有许多理论问题有待进一步研究。但可以推断,随着研究的深入,蚁群算法也将与其他模拟进化算法一样,获得越来越多的应用。

参考文献:

[1] A Colorni,M Dorigo,V Maniezzo. Distributed Optimization by Ant Colonies[C]. Proceedings of ECAL91 European Conference of Artificial Life,Paris,France,Elsevier Publishing,1991. 134-144.

[2] M Dortgo,Vittorio Maniezzo,Alberto Colorni. Ant System:Optimization by Ant Colonies Cooperating Agents[J]. IEEE Transactions on Systems,Man,and Cybernetics-Part B: Cybernetics,1996,26(1):29-41.

[3] M Dortgo,L M Gambardella. Ant Colony System:A Cooperative Learning Approach to the Traveling Salesman Problem[J]. IEEE Trans. Evolutionary Computation,1997,1(1):53-66.

[4] Hoshyar R,Jamali S H,Locus C. Ant Colony Algorithm for Finding Good Interleaving Pattern in Turbo Codes[J]. IEEE Proceedings Communications,2000,147(5):257-262.

[5] Young-Jae Jeon,Jae-Chul Kim. Application of Ant Colony Algorithm for Loss Minimization in Distribution Systems[J]. Transactions of the Korean Institute of Electrical Engineers,2001,50(41):88-96.

[6] Gianni Di Caro,Marco Dorigo. Ant Net:Distributed Stigmergetic Control for Communications Networks[J]. Journal of Artifcial Intelligence Research,1998,(9):317-355.

[7] Dorigo M,Di Caro G. Ant Colony Optimization:A New Meta-heuristic[C]. Proceedings of the 1999 Congress on Evolutionary Computation,Washington,DC,USA,1999. 1477.

[8] Gambardella L M,Dorgio M. An Ant Colony System Hybridized with a New Local Search for the Sequential Ordering Problem[J]、INFORMS Journal on Computing,2000,12(3):55-237.

[9] N Meuleau,M Dortgo. Ant Colony Optimization and Stochastic Gradient Descent[C]. Artif. Life,2002,8(2):103-121.

[10] Guest Editorial. Special Section on Ant Colony Optimization[J]. IEEE Transactions on Evolutionary Computation,2002,6(4).

[11] T Stutzle,H H Hoos. MAX-MIN Ant System[J]. Future Gener. Comput. Syst.,2001,16(8):889-914.

[12] Seung Gwan Lee,Tae Ung Jung,Tae Choong Chung. An Effective Dynamic Weighted Rule for Ant Colony System Optimization[C]. Proceedings of the 2001 Congress on Evolutionary Computation, NJ, USA,IEEE Press,2001. 393-397.

[13] Cheng-Fa Tsai,et al. A New Approach for Solving Large Traveling Salesman Problem Using Evolution ant Rules[C]. Neural Networks, IJCNN '02,Proceedings of the 2002 International Joint Conference, Honolulu,HI,USA,IEEE Press,2002. 1540-1545.

作者简介:

朱海梅(1973-),女,江苏扬州人,硕士研究生,主要研究方向为智能控制;朱庆保(1955-),男,江苏南京人,教授,硕士生导师,主要研究方向为智能控制。