第31卷 第1期 计 算 机 工 程 2005年1月
Vol.31 №1 Computer Engineering January 2005
·人工智能及识别技术· 文章编号：1000—3428(2005)01—0157—03 文献标识码：A 中图分类号：TP301.6

基于粗粒度模型的蚁群优化并行算法

朱庆保
(南京师范大学计算机系，南京 210097)

摘 要：为了改进蚁群优化算法的收敛速度，研究了一种基于粗粒度模型的并行蚁群优化算法，该算法将搜索任务划分给q个子群，由这些子群并行地完成搜索，可使搜索速度大幅度提高。实验结果表明，用该算法求解TSP问题，收敛速度比最新的改进算法快百倍以上。

关键词：蚁群优化算法；蚁群系统；并行算法；粗粒度模型

Ant Colony Optimization Parallel Algorithm Based on Coarse-grained Model

ZHU Qingbao
(Department of Computer Science , Nanjing Normal University, Nanjing 210097)

【Abstract】 In order to improve the speed of convergence of ant colony optimization, a parallel algorithm based on coarse-grained model is proposed in the paper, search tasks are assigned to q ant subgroups ,and parallel searching are finished by q subgroups. Results of experiment show that the algorithm described in this paper makes the searching speed hundreds of times faster than the latest improved algorithm.

【Key words】 Ant colony optimization algorithm; Ant colony system; Parallel algorithm; Coarse-grained model

蚁群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优化算法，是由意大利学者M.Dorigo等人于20世纪90年代提出的[1,2]，当时称之蚂蚁系统（Ant System 简称AS)。M.Dorigo等人用该方法求解旅行商问题(Traveling Salesman Problem 简称TSP)、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它研究者的注意，并用该方法求解一些实际问题。继AS算法，M.Dorigo等人又提出了改进的ACS模型[2]，1999年又将AS、ACS(Ant Colony System)纳入蚁群优化的框架，称为ACO(Ant Colony Optimization)[3]。近几年，蚁群优化算法成为研究热点之一。

蚁群算法已得到较广泛的应用，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[4,5]。其中，文献[4]提出了一种新的改进的信息素更新策略：其一，局部信息素修改时，挥发系数动态改变。其二，全局信息素更新时，则将蚂蚁所走路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[5]提出了一种基于蚂蚁进化规则的算法。还有不少其它改进文献。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。在大规模优化问题中，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，作者研究了基于粗粒度模型的并行蚁群优化算法，取得了很好的效果。

1 蚁群算法的基本原理与问题描述

1.1 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。

1.2 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。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，其思想可推广到其它优化问题。

为了叙述简便，首先作出如下定义：

记A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A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_{ij}。

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

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 $ant_i=\{1, 2, \cdots k, \cdots, m\}$表示第$i$组$m$只蚂蚁的集合，$i=1,2,\cdots, q$，$k \in ant_i$表示第$i$组的某只蚂蚁，$\tau_{ij}(t)$表示

基金项目：江苏省教育厅自然科学基金资助项目(2001SXXTSJB111)
作者简介：朱庆保(1955－)，男，教授，主研方向：人工智能与智能控制
定稿日期：2003-10-10 **E-mail**：qbzhu@email.njnu.edu.cn

蚂蚁l时刻在$e_{ij}(i、j \in R)$上残留的信息量。

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_c时刻返回节点i. 任意时刻所处的位置为P，$\forall P$在$\sum_0$都有确定的坐标(x, y)，在节点的位置记作$P(t_i)$，若有$P_i(x_i, y_i)$的坐标位置与$c_i(x_i, y_i)$相同，称P_i和c_i等价。令$tabu_k = \{P(t_0), P(t_1), \cdots, P(t_i)\}$为蚂蚁$k$从$t_0$时刻到$t_i$时刻已走节点位置的集合(等价于已走节点集合)，$t_{i+1}$时刻，$\forall P(t_{i+1}) \in C$且$\forall P(t_{i+1}) \in tabu_k$，则称$\forall P(t_{i+1})$为$t_{i+1}$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。若用Σ表示可行点集，则有$\Sigma = C - tabu_k$。

定义4 记$T = \{t_0, t_1 \cdots, t_c\}$且有$t_0 < t_1 <, \cdots, < t_c$，$\forall P \in A$，$\forall P_i \in \Sigma, i \in R$若映射$f$: T→A使得$f(t_0)=P_{i0}, f(t_1)=P_{i1}, f(t_c)=P_{i0}$，则称像集$PL=f(T)$为$k$从$P_{i0}$走到$P_{i0}$的一条封闭路径上的节点集，这些节点在A中的连线称封闭路程，路程长度记作L，用式(2)计算，其中d_l由式(1)计算

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

定义5 $br_i(c_i(x_i, y_i)) = \{c \mid c \in C, d(c, c_i) \le d_{\min}\}$为$c_i$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6 $\eta_{ij} = 1/d(c_i, c_j)$，称蚂蚁从节点$i$为选择节点$j$的启发函数。

2 基于粗粒度模型的并行蚁群算法

目前的ACO(包括AS、ACS及其改进版)算法用于小规模优化问题时效果还是较好的，随着城市数的增多，其搜索时间迅速增加，以致于达到令人不能容忍的程度。目前已有的改进算法虽然使收敛速度有一定的提高，但效果不是特别明显，例如最大最小算法是最著名的改进算法，在其实验报告中，在超过100个城市的较大规模的TSP问题优化中，搜索代数仍需要上万代。因此，要大幅度提高搜索速度，必须从并行处理上进行考虑，为此，作者研究了基于粗粒度模型的并行算法，取得了很好的效果。

2.1 并行性分析

用蚁群算法求解TSP问题就是让m只蚂蚁遍历n个城市后返回出发点，从而搜索到一条最短路径。由于每只蚂蚁沿所有城市周游一周的过程(包括其中的计算)是完全独立的，基于这种并行性，完全可以将任务分到q个处理器上同时进行并行搜索。

2.2 粗粒度模型并行算法

该算法的基本思路是将蚁群分成q个子群并分配到q个处理器(或专用小型局域网工作站)，由q个处理器并行地完成最优路径的搜索。同时，q个子群都出现停滞的概率是很小的，这样可大大改进已有算法在上述两方面的缺陷，从而使这种并行蚁群优化算法用于大规模优化问题成为可能。图1示出了局域网并行处理模型示意图。

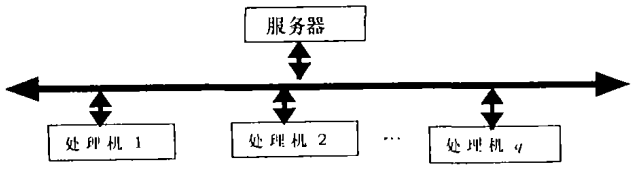

图1 并行处理示意图

(1) 粗粒度模型并行算法的通信策略

在粗粒度模型中，每个子群对全局搜索的贡献主要是留在路径上的信息素及每个子群找到的最优路径。按ACS算法，每只蚂蚁每走过一个边都要修改局部路径的信息素，q个处理器若将这些修改后的信息素进行通信交流，通信量较大，为此，每组蚂蚁周游一周后，仅将最佳路径上的信息素得到的最佳路径进行交流。并保留最佳路径。

(2) 粗粒度模型并行策略下的子群算法

由于各子群算法相同，因此，仅以一个子群算法为例，根据第1节定义，各子群算法步骤如下：

Step1: 初始化：将n个城市分别按x、y坐标排序，找出q个边部城市，将q组蚂蚁分别分配到这q个城市上，同时设置到$tabu_{m\times p}$其中，$j=1,2,\cdots, p$，$p=m/2$。$tabu_{k\times 2}$是蚂蚁k_1、$k_2 \in ant_i$共用的禁忌表(因此处以一组为例，故省略了下标i，考虑每对蚂蚁的算法相同，以下以其中的一对蚂蚁为例，故再将下标i省略，记为$tabu_{k\times 2}$)。以n个城市分别为起点按距离排序，从而确定n个节点的邻居节点集br_i，$i=1,2\cdots$，n;可设邻居节点的数量为n/w，$n/w \in [1, 20]$，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β、α和ρ的值。

Step2: k_1，$k_2 \in ant_i$分别以各自所在节点i、h为中心(若为起始出发节点，则$i=h$)，分别按最近邻居原则选择下一个节点j和x，$j, x \in R$，$j \in br_i$，$x \in br_h$，分别从br_i和br_h个城市中找出z_1、z_2个未走过的城市$j_1, j_2 \cdots j_{z1}$，$j_1, j_2 \cdots x_1, x_2 \cdots x_{z1}, x_{z2}$即$j_p, x_p \notin tabu_{k\times 2}$。

Step3: k_1在z_1个城市中，按式(3)或式(4)选择节点j，k_2在z_2个城市中选择x，算法与k_1相同，以下以k_1为例。

$$j = \begin{cases} \arg\max_{l \notin tabu_{k\times 2}} \{[\tau_{il}(t)][\eta_{il}(t)]^{\beta}\} & if \quad q \le q_0 \\ S & otherwise \end{cases} \tag{3}$$

式中，$0<q_0\leqslant 1$，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。

$$p_{ij}^k = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{k \notin tabu_{k\times 2}} [\tau_{ik}(t)]^{\alpha}[\eta_{ik}]^{\beta}} & j \notin tabu_{k\times 2} \\ 0 & j \in tabu_{k\times 2} \end{cases} \tag{4}$$

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$tabu_{k\times 2}$。将k_2用同样算法选择的x也加入$tabu_{k\times 2}$。

本组所有蚂蚁均按上述算法选择一个节点，并添加到各自的禁忌表中。

Step4: 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$1-\rho$表示信息消逝程度，每只蚂蚁走完一个边后，即按式(5)进行局部信息更新。

$$\tau_{ij}^{new} = (1-\rho)\tau_{ij}^{old} + \rho\Delta\tau_{ij}$$
$$\Delta\tau_{ij} = \frac{Q_1}{L_z}, \tag{5}$$
$$L_z = \sum_{l_1}^{v} d_{l_1}^1 + \sum_{l_2}^{v} d_{l_2}^2 + \cdots + \sum_{l_m}^{v} d_{l_m}^m = \sum_{k=1}^{m}\sum_{l=1}^{v} d_l^k$$

$$d_l \subseteq d_{ij}, \ l = 1,2,\cdots x, \ \forall \ i, j = 1,2,3\cdots n.$$

式中，Q_1为一个较大的常数，v是蚂蚁k在本次周游中已走过的城市边数；d_l是蚂蚁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。

该策略保证了各蚂蚁所留信息素的的自适应性和均衡性，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提高收敛速度的关键之一。

Step5: m只蚂蚁选择完节点j后，令$i_{new}=j$; $j_{new}=j_{old}+1$; 返Step2开始选择下一个节点。直到所有蚂蚁周游完一周。

Step6： m只蚂蚁周游完一周后、用式(2)计算路程长度L_k，找出L_{kmin}并保留，$L_{kmin}=\min L_k$。

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

$$\Delta\tau_{ij}^{k}=\begin{cases}\sum_{l=1}^{n}\frac{Q}{d_l} & if\ l\in global-best-tour\\ 0 & otherwise\end{cases} \quad (6)$$

Step7： 按式(6)进行全局信息素更新。m只蚂蚁走完全部路程并对最优个体完成倒位运算后，仅对最佳路径上的信息素按式(6)进行更新。

式中，Q为常数，l∈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。

Step 8： 信息素通信：即将更新后的信息素和最优路径传给服务器，由服务器将更新后的信息素传给其它m-1个分处理器。

Step 9: 服务器将各组蚂蚁本次周游得到的L_{kmin}与已得到的最优长度l_{il}比较，若有$l_{kmin}<l_{il}$则用l_{kmin}替换l_{il}同时替换最优路径表。

Step10： 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。

3 实验比较

为了验证、比较算法的效果，作者从TSPLIB下载了多个著名的TSP问题的范例进行了仿真实验，都取得了非常好的结果。下面是仿真8个处理器并行处理得到的实验结果。

图2 是用不同方法优化pr107 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，2为用文献[5]改进算法得到的收敛特性。

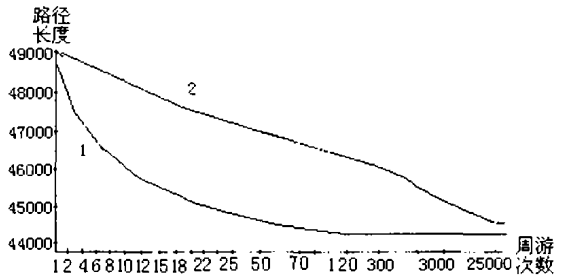

图1 求解pr107 收敛特性比较

表1 实验结果比较

	TSPLIB 最优结果	本文算法 优化结果	ACS 优化结果	改进ACS 优化结果	本文算法 收敛代数	ACS或改进 算法收敛代数
Pr107	44 303	44 302	44 483	无实验	110	20 000
Lin318	41 345	41 358	43 677	42 070	300	10 000

实验结果表明，ACS算法需周游25 000次以上才收敛到较优值，本文算法仅需100多次即收敛到最优值。实验结果见表1。

为了和最著名的改进算法相比较，表1还列出了lin318实验结果。

在表1中的TSP最优结果指TSP问题库公布的最新优化解。改进算法指文献[4]提出的算法，Lin318的ACS结果和改进算法结果均摘自该文献。从结果看，本文算法结果大大优于ACS算法和最著名的改进算法。收敛速度则快数十至数百倍。

4 结语

实验表明，蚁群算法搜索速度慢的主要原因是搜索过程经常处于停滞状态，也就是搜索后期大多数周游过程所有蚂蚁所走的路径相同。仅靠q_0控制增加随机搜索，其效率很低，原因在于随机选择节点产生的路径使解变优的概率很低。本文介绍的并行算法将搜索任务划分到m个子蚁群，由m个子蚁群独立的进行并行搜索，且每组蚂蚁负责一个出发点城市的路径搜索，由于出发点不同，搜索路径的过程是完全不同的，因而大大提高了搜索的多样性，大幅度减少了停滞现象，因此使收敛速度大幅度提高，特别是在大规模优化应用时具有明显优势。

然而，蚁群优化算法是一种新的模拟进化算法，其研究才开始不久，还没有像GA等算法那样形成系统分析的方法和坚实的数学基础，所以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，蚁群算法也将同其它模拟进化算法一样，获得越来越多的应用。

参考文献

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

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

3 Dorigo M, Di Caro G. Ant Colony Optimization: A New Meta-heuristic. Proceedings of the 1999 Congress on Evolutionary Computation, Washington, DC, USA, IEEE Press, 1999, 2: 1477

4 Lee S G, Jung T U, Chung T C. An Effective Dynamic Weighted Rule for Ant Colony System Optimization. Proceedings of the 2001Congress on Evolutionary Computation , NJ, USA, IEEE Press 2001, 2:393-397

5 Tsai C F, Tsai C W. A New Approach for Solving Large Traveling Salesman Problem Using Evolution Ant Rules. Proceedings of the 2002 International Joint Conference on Neural Networks,2002, IJCNN '02, Honolulu, HI, USA , IEEE Press,2002,2: 1540- 1545

（上接第121页）

参考文献

1 Kent S, Atkinson R. Security Architecture for the Internet Protocol. RFC 2401, 1998-11

2 Gammage N, Waters G. Securing the Smart Network with Motorola Security Processors. Available on e-www.motorola.com, 2003-03

3 Motorola. MPC8260 PowerQUICC II™ Family Reference Manual. Available on e-www.motorola.com. ,2002-05

4 Motorola. MPC180LMB Security Processor Users Manual. Available on e-www. motorola.com, 2002-03