求解自适应组合优化蚁群算法的研究

孙泽宇[1],邢萧飞[2,3]
SUN Ze-yu[1],XING Xiao-fei[2,3]

1.洛阳理工学院 计算机与信息工程系,河南 洛阳 471023
2.中南大学 信息科学与工程学院,长沙 410083
3.日本筑波大学 计算机与科学系,日本 筑波 305-8573
1.Computer and Information Engineering Department,Luoyang Institute of Science and Technology,Luoyang,Henan,471023,China
2.School of Information Science and Engineering,Central South University,Changsha 410083,China
3.Department of Computer Science,Graduate School of Systems and Information Engineering,University of Tsukuba.Tsukuba 305-8573,Japan
E-mail:sundazhi1977@163.com

SUN Ze-yu,XING Xiao-fei.Research on solution to adaptive ant colony algorithm of combinatorial optimization.Computer Engineering and Applications,2009,45(35):31-33.

Abstract: It costs large quantity of time and tends to be trapped into local optimizing computation by using traditional ant colony algorithm of combinational optimization in the process of computing.Besides,it also causes much useless redundant iterated codes and leads to low efficiency.In order to solve these problems,an adaptive algorithm of combinatorial optimization is proposed in this paper.Through the ways of changing the iteration of pheromones,analyzing the selection of parameters and increasing the partial update modes of pheromones,the overall computing efficiency and convergence rate is enhanced and the scope of optimization is expanded.What's more,it decreases the useless iterated codes and occurrence of stagnation the simulation experiment of the combinatorial optimization on Traveling Salesman Problem proves the feasibility and validity of this algorithm.
Key words: Ant Colony Optimization;adaptive;combinatorial optimization;pheromone;Traveling Salesman Problem

摘 要:传统的组合优化蚁群算法在求解过程中要消耗大量的时间,极易陷入局部最优化求解等弊端,同时还会产生大量无用的冗余迭代码,运算效率低。对此,提出了自适应组合优化蚁群算法。通过对改变信息素的迭代、参数选择的分析和增加对信息素局部更新方式,提高了整个系统运算速度及收敛速度,扩充了优化的范围,克服了无用迭代码的产生,减少了停滞现象的出现。通过该算法对旅行商问题进行仿真实验,其结果表明了该算法的可行性和有效性。
关键词:蚁群算法;自适应;组合优化;信息素;旅行商问题
DOI:10.3778/j.issn.1002-8331.2009.35.010 **文章编号:**1002-8331(2009)35-0031-03 **文献标识码:**A **中图分类号:**TP18

1 引言

蚁群算法(Ant Colony Optimization)是由 Dorigo、Maniezzo 和 Colorni 于 1991 年提出的,这是最初的 ACO 算法[1-2]。该算法早期应用于旅行商问题(Traveling Salesman Problem,TSP)、指派问题(Quadratic Assignment Problem,QAP)、调度问题(Scheduling Problem,SP)等问题求解。经过大量实验数据可以证明,该算法具有较强的鲁棒性和搜索能力,特别适合于在离散空间进行多点优化问题的求解。但是随着问题的深入研究,许多学者发现在解决实际问题时会出现运算周期过长、易陷入局部最优化求解和收敛速度慢等缺点[3]。基于这些问题提出了一种改进后的自适应组合优化蚁群算法来求解 TSP 优化组合问题。该算法通过对信息量更新,有效地抑制了过早收敛现象的发生,同时改进了蚁群算法在全局收敛性,提高了收敛速度,对算法优化减少了迭代次数,提高运算效率,从而较好地解决一系列组合优化问题。通过对该算法的仿真,证明了数据的有效性和可行性。

2 蚁群算法的基本原理

2.1 蚁群行为描述

蚂蚁属于群居昆虫,个体行为极其简单,但由这些简单的个体所组成的蚁群却表现出行为的复杂性和有序性。相互协作的一群蚂蚁很容易找到从蚁穴到食物源的最短路径。蚂蚁在寻找食物的过程中会留下一种称之为信息素(pheromone)的物质来进行信息传递。后面的蚂蚁可根据前边的蚂蚁所留下的信息

基金项目:国家公派留学基金项目(No.2009103025)。
作者简介:孙泽宇(1977-),男,讲师,研究方向:分布式计算与并行算法研究;邢萧飞(1979-),男,博士,研究方向:无线传感器网络覆盖与并行算法研究。
收稿日期:2009-08-14 **修回日期:**2009-10-15

素来选择自己所走路径,蚂蚁通过这种信息素并且根据信息素的浓度来指导前进方向的选择。当一条路径上信息素越多时,其他蚂蚁将以越高的概率来选择该路径,从而该路径上的信息素会被加强。由于大量蚂蚁组成的蚁群的集体行为便表现出一种信息正反馈现象,最优路径上的信息素浓度越来越大,而其他路径上的信息素浓度会消减。最终整个蚁群系统会找出最优路径。蚂蚁算法是基于以上原理产生的,它是一种随机搜索算法,与其他模型进化算法一样,是通过候选解组成的群体进化过程来寻找最优解。

2.2 蚁群算法基本模型

以 TSP 问题作为蚁群算法的基本框架。TSP 问题的目标函数就是寻找能过 n 个城市各一次且最后回到出发点的一条长度最短的 Hamilton 回路,即遍历所有顶点当且仅当一次的最短回路。设有 n 个城市 $d_{ij}(i,j=1,2,3,\cdots,n)$ 代表城市 i 和城市 j 之间的距离。$m=\sum_{i=1}^{n} b_i(t)$ 表示 t 时刻位于城市 i 的蚂蚁数目。$\tau_{ij}(t)$ 表示 t 时刻在城市 i 和城市 j 之间的信息量,即蚁群所释放真实信息量。η_{ij} 是边 (i,j) 一个启发式因子,表示城市 i 转移到城市 j 的启发程度,η_{ij} 在蚁群算法中是不改变的。初始时刻,各条路径上的信息素量相等,$\tau_{ij}(0)=C$,C 为常数。每个蚂蚁都有一个禁忌表 $tabu_k$,并把第一个元素设置为其初始城市。用 $P_{ij}^k(t)$ 表示在 t 时刻蚂蚁 k 由城市 i 转移到城市 j 的概率[4-5]。在选路过程中概率函数是:

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)\tau_{ij}^{\beta}(t)}{\sum\limits_{s\in allowed}\tau_{is}^{\alpha}(t)\tau_{is}^{\beta}(t)} & j\in allowed\\ 0 & \text{otherwise}\end{cases}\tag{1}$$

其中 $allowed=\{0,1,2,\cdots,n-1\}-tabu_k$ 表示蚂蚁 k 下一步允许选择的所有城市,列表 $tabu_k$ 记录了当前蚂蚁 k 所走过的城市,当所有 n 个城市都写入到禁忌表 $tabu_k$ 中时,蚂蚁 k 就完成一次循环,所经过的路径就是问题的一个解。对于启发函数 $\eta_{ij}(t)$ 与城市 i 和城市 j 之间的函数表达式为:

$$\eta_{ij}(t)=\frac{1}{d_{ij}}\tag{2}$$

其中 d_{ij} 表示城市 i 与城市 j 之间的距离。α 是启发式因子,表示轨迹的相对重要性,反映蚂蚁在运动过程中所积累的信息量在指导蚁群搜索中的相对重要程度,其值越大,选择以前所走过的路径可能性越大,蚂蚁之间协作性越强;其值越小,则易使蚁群的搜索过早陷入局部最优。β 是期望启发式因子,表示能见度的相对需要性。反映了启发式信息在指导蚁群搜索过程中的相对重要程度,其大小反映了蚁群寻优过程中先验性、确定性因素的作用强度[6]。其值越大,则该状态转移概率越大;其值越小,则易产生随机搜索,在这种情况下很难找到最优解。随着时间的推移,蚂蚁所走过路径上的信息素会逐渐挥发,这就要求对路径上的信息素的浓度进行调整。在 $t+n$ 时刻,路径 (i,j) 上的信息素量可调整为:

$$\tau_{ij}(t+n)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}(t)\tag{3}$$

$$\Delta\tau_{ij}(t)=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t)\tag{4}$$

式(2)中的 ρ 表示信息素挥发系数,$1-\rho$ 表示为信息素残留因子。ρ 的取值范围是 $\rho\subset[0,1)$,$\Delta\tau_{ij}$ 表示在某条边上的累加新增信息素之和,$\Delta\tau_{ij}^k$ 表示 t 时刻与 $t+n$ 时刻之间第 k 个蚂蚁在此边上留下信息素的数量。

对于具体的算法的不同,$\Delta\tau_{ij}^k$ 表示形式也不同 Dorigo.M 给出三种不同模型,分别为蚁周系统(ant cycle system)、蚁量系统(ant quantity system)、蚁密系统(ant density system)。在 ant cycle system 模型中

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}\dfrac{Q}{L_{ij}} & \text{蚂蚁 } k \text{ 在 } t \text{ 和 } t+1 \text{ 之间经过}(i,j)\\ 0 & \text{otherwise}\end{cases}\tag{5}$$

信息素强度 Q 是蚂蚁循环一周时释放在所经路径上的信息素总量。L_k 是蚂蚁 k 在本次循环中所走的路径长度。

在 ant quantity system 模型中

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}\dfrac{Q}{d_{ij}} & \text{蚂蚁 } k \text{ 在 } t \text{ 和 } t+1 \text{ 之间经过}(i,j)\\ 0 & \text{otherwise}\end{cases}\tag{6}$$

在 ant density system 模型中

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}Q & \text{蚂蚁 } k \text{ 在 } t \text{ 和 } t+1 \text{ 之间经过}(i,j)\\ 0 & \text{otherwise}\end{cases}\tag{7}$$

公式(6)和(7)是利用局部信息,即蚂蚁完成一步就要对所走路径进行信息素的更新,而公式(5)是利用整体信息,即蚂蚁完成一次循环后对所走路径进行信息素的更新。在求解 TSP 过程中,ant cycle system 性能较为稳定。

3 算法的改进策略

蚁群算法通过正反馈机制来强化较好的解,但容易出现停滞和陷入局部最优解[7-8]。针对这个问题,对自适应信息素更新方式、算法优化等两方面进行了改进。从而增大路径选择概率,加快收敛速度,并能避免出现过早收敛于当前循环最短路径。

3.1 局部更新规则

蚂蚁从城市 i 转移到城市 j 后,对路径 (i,j) 上的信息素按公式(8)更新[5,7]:

$$\tau_{ij}(t+1)=(1-\xi)\tau_{ij}(t)+\xi\Delta\tau_{ij}(t)\tag{8}$$

其中 $\xi\in(0,1)$,$\Delta\tau_{ij}=\Delta\tau(0)$ 是信息量的初值[5,9]。这样可以有效地避免陷入局部最优,减小了已选择过的路径再次被选择的概率,从而有效地避免蚂蚁收敛到同一路径,提高了算法的全局搜索能力。

3.2 全局更新规则

蚂蚁的搜索主要集中在当前循环为止所找出的最短路径的领域区,全局更新规则是在所有的蚂蚁都完成它们的路径之后执行[10],更新信息素公式:

$$\tau_{ij}(t+1)=(1-\xi\cdot\tau_{ij}(t))\cdot\tau_{ij}(t)+\xi\cdot\tau_{ij}(t)\Delta\tau_{ij}(t)\tag{9}$$

$$\Delta\tau_{ij}^{k}(t)=\begin{cases}\dfrac{Q}{L_{\min}} & \text{边}(i,j)\in L_{\min}\\ -\dfrac{Q}{L_{\max}} & \text{边}(i,j)\in L_{\max}\\ 0 & \text{otherwise}\end{cases}\tag{10}$$

其中,ξ 是信息素挥发系数,如对 ξ 做一定的变化,可以防止该路径上的信息素浓度无限地增加,可减少陷入局部最优化的可能,从而加快了收敛速度。$L_{\min}$,$L_{\max}$ 是蚂蚁所走过最短和最长路径[11]。在算法的初期阶段,让蚂蚁走过的各路径都有机会获得信息量的更新,进而提高蚂蚁选择相应路径的概率。随着某

些路径上信息素的不断积累，在循环次数达到一定数目时，缩小了最优路径与最差路径的边之间的信息量差距，从而使蚂蚁的搜索行为集中于最优路径的附近[12]。

3.3 算法描述

3.3.1 自适应信息素挥发因子的调整

为了在保证收敛速度的前提下更好地找到全局最优，对挥发系数ρ做了自适应的改变，并对局部路径上的信息素进行更新，有利于算法收敛，同时也提高了全局的最优化[13]。由于信息量的挥发系数ρ的存在，使那些从未被搜索到信息量会减少到接近0，降低了算法的全局搜索能力。当ρ太大，以前搜索过的路径可能被再次选择，影响算法的全局收敛能力，易于陷入局部最优解。因此对ρ值做自适应改变，ρ的初始值$\rho(t_0)=1$，当算法在求得最优值N次时没有明显改进时，降低ρ值，公式为：

$$\rho(t)=\begin{cases}0.95\rho(t-1) & 0.95\rho(t-1)\geqslant\rho_{\min}\\ \rho_{\min} & \text{otherwise}\end{cases}\tag{11}$$

所降低的ρ值，不能小于$\rho_{\min}$。这样可以防止ρ过小，降低算法的收敛速度。

3.3.2 算法步骤

步骤 1 初始化参数：令时间t=0，循环次数N_c=0，设置$\eta_{ik}=d_{ik}$；τ_{ij}=0；$\Delta\tau_{ij}$=0，并将出发点置于禁忌表$tabu_k$中，给定α,β,ρ的值，边$(i,j)\in R$；

步骤 2 当while不满足条件算法终止条件时，利用for循环将m只蚂蚁放置于初始城市上。

步骤 3 蚂蚁个体根据状态转移公式(1)计算的概率选择城市j并前进，此时$j\in(C-tabu_k)$。

步骤 4 修改禁忌表中的指针，将城市j写入禁忌表中，让指针指向城市j。

步骤 5 查找是否有未遍历的城市，如果$k<m$，执行k++后跳转到步骤3。//每跳转一次，都要修改禁忌表中的指针，让其指向下一个新的城市。

步骤 6 计算m蚂蚁的路径长度L，记录当前的最优解$L_{\min}$，求出第一次循环中得到最优路径为L_k。

步骤 7 当蚂蚁遍历过所有城市后，就要对路径上的信息素按公式(8)(9)(10)更新。当出现局部最优解时按公式(11)对ρ值进行更新。

步骤 8 当$L_{\min}\leqslant L_k$时，更新τ_{ik}和$\Delta\tau_{ik}$，并替换最优路径表。

步骤 9 如果循环次数大于或等于循环最大次数时，循环结束并输出最优路径长度L_k。否则清空禁忌表，并跳转到步骤2。

4 仿真实验与分析

从通用TSPLIB[14]中选取对称TSP问题进行测试，实验所设置的环境为：赛扬处理器1.6 GHz，512 MB的内存，Window XP，VC++，Matlab6.5作为开发平台。设蚂蚁数量与城市数量相等，参数取值为α=1，β=5，ρ=0.5，Q=100，运行10次，每次运行迭代200次，实验结果如表1所示。

表1 对TSP问题实验的结果

TSP	该文算法最优解	该文算法最差解	实验结果平均值	用TSPLIB所得最优解
Eil51	426.0	432.0	428.1	426.0
Olive30	423.7	431.4	424.3	423.7
ST70	675.0	757.0	702.0	675.0

在以Olive30TSP为例，在不同参数下自适应蚁群算法与基本蚁群算法的仿真实验结果。实验结果如表2所示。

表2 不同参数下的实验结果

	α	β	ρ	最短路径长度	迭代次数
基本蚁群算法	5	2	0.9	429.7	339
	2	2	0.9	427.0	344
	4	1	0.5	428.6	345
该文改进算法	1	4	0.5	423.7	27
	1	5	0.5	425.1	27
	4	1	0.1	424.3	24

图1和图2分别给出了Olive30TSP改进算法最好解与最坏解的进化曲线。综合上述仿真实验可以得到，改进算法使全局性搜索能力增强，加快收敛速度，防止过早陷入局部最优解。

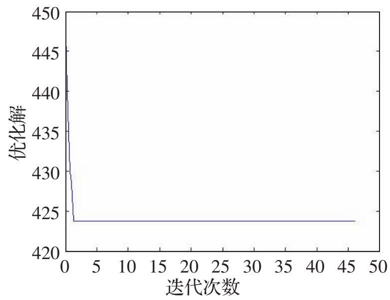

图1 自适应蚁群算法最优解的进化曲线

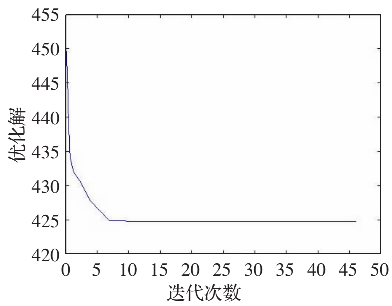

图2 自适应蚁群算法最坏解的进化曲线

由表1和表2所示的实验数据可看出，改进算法减少冗余代码的产生，加快收敛速度，提高了求解的质量，避免了陷入局部最优过程。另一方面，在不同的参数下所得到的最坏结果小于基本蚁群算法的最优结果，进而缩减了迭代次数[9,15]。通过对图1和图2的分析可知自适应蚁群算法最优解的迭代次数和最优路径长度都小于最坏解的次数和长度，从而防止了早熟停滞现象的产生，加快了对全局收敛的速度，进而可以快速地搜索出最优路径。

5 结语

蚁群算法是一种新型的模拟进化算法[16]，结合蚁群算法的优点，提出了自适应组合优化蚁群算法可以有效地克服停滞现象和陷入局部最优解等弊端，同时使得搜索的收敛速度得以加快；通过对路径上信息素的更新，扩大搜索解的空间，增强了算法的全局遍历性。实验结果表明，该算法可以有效地解决旅行商问题。此外，蚁群算法在其他组合优化问题上的应用以及其他研究领域的应用都是值得进一步研究的问题。

（下转37页）