用改进蚁群算法求解函数优化问题*

唐　泳，马永开，唐小我
（电子科技大学 管理学院，四川 成都 610054）

摘　要：提出将蚁群算法用于求解函数优化问题的新方法。使用一定数量的蚂蚁在解空间中首先随机搜索，然后模拟蚂蚁觅食的方式，更新搜索路径上的信息素，按照转移概率来决定搜索方向，即通过信息素来指引搜索，最后搜索收敛于各个全局最优解。给出了基于此思想的具体算法，并通过计算示例仿真说明了该算法的有效性，表明该算法可以同时快速收敛发现多个全局最优解，并保持稳定。

关键词：函数优化；蚁群算法；进化算法；仿生算法
中图法分类号：O22　　**文献标识码**：A　　**文章编号**：1001-3695(2004)09-0089-03

An Improved Ant Colony Algorithm for Function Optimization

TANG Yong, MA Yong-kai, TANG Xiao-wo
(*College of Management, University of Electronic Science & Technology of China, Chengdu Sichuan* 610054, *China*)

Abstract: An improved ant colony algorithm solving function optimization problem is proposed in this paper. The algorithm uses some ants to search in the solution space first in a stochastic way then stimulate the food searching behavior of real ants to guide the search by the pheromone. The new algorithm is explained in details and some simulations show the algorithm is very effective in finding global optimizations.

Key words: Function Optimization; Ant Colony Algorithm; Evolutionary Algorithm; Bionic Algorithm

1　引言

蚁群算法(Ant Colony Algorithm, ACA)是一种崭新的仿生模拟进化算法。ACA 的思想就是模拟蚂蚁觅食行为，即使用大量 Ant 在搜索空间中随机搜索，并且用信息素 Pheromone 来加强搜索路线，引导其他 Ant 的搜索，同时引入信息素的挥发机制 Evaporation。这种正反馈使得该算法能够找到全局的最优解，而不会像其他搜索算法那样容易陷入局部最优解。该算法由 M. Dorigo[1] 提出之后，在旅行商问题 TSP, QAP, JSP 的求解中取得较好的效果[2~5]。现在蚂蚁算法已经在电力网络优化、网络路由分配、函数优化、集成电路布线、聚类分析等领域得到应用。

函数优化问题是在工程、管理、经济、控制、决策中普遍存在的一类优化问题，其优化目标函数可能包含多个在一定范围上的连续变量，传统的优化手段对目标函数要求苛刻，如需要满足可导或可微等。导致有些函数难以优化，容易陷入局部解而难以得到全局最优解，收敛速度较慢。模拟退火算法(Simulated Annealing)、遗传算法(Genetic Algorithm)、神经网络算法(Neural Networks)、禁忌算法(Tabu Search)等算法已经在这个领域得到了研究，也有将蚁群算法 ACA 在这个领域的研究[6~11]。本文将从新的角度来研究将 ACA 算法思想用于求解函数优化问题，其核心是先将各个分量细化，建立搜索空间，用一定数量的蚂蚁在解空间中首先随机搜索，然后再模拟蚂蚁觅食的方式，通过信息素来指引搜索，反复搜索后，得到函数优化结果。通过细化解向量，建立解空间，就将连续问题转换为了离散问题，所以基于蚁群的求解本质上是离散优化求解。

收稿日期：2003-12-13；**修返日期**：2004-05-17
基金项目：电子科技大学青年科技基金资助项目

2　函数优化问题描述

假设目标函数 $F(X)$ 的决策变量是一个 n 维向量 $X=(x_1, x_2, \cdots, x_n)^T$，如果是带约束的优化问题，则可以通过引进惩罚函数等手段将带约束的优化问题转换为不带约束的优化问题，当然对于最大值问题，可以通过简单的变换转为最小值问题，这样优化问题则可以统一为讨论目标函数 $F(X)$ 在各个决策分量区间 $\min_i \leqslant x_i \leqslant \max_i, i=1,2,\cdots,n$ 上的最小值问题。区间 $[\min_i, \max_i], i=1,2,\cdots,n$ 可以通过求解约束条件得出，并确定了目标函数的优化搜索 n 维空间。

3　ACA 原型算法思想

ACA 算法是一种典型的仿生模拟进化算法，在许多领域得到了应用，特别是在 TSP 问题的求解。所以，下面用 ACA 在旅行商问题 TSP 的求解来简要说明 ACA 原型算法思想。

TSP 即旅行商问题 Traveling Salesman Problem 指，对于给定的一组城市 Cities，每个城市之间的距离已知，求出一条总长度最小的封闭路线，该路线通过每一个城市一次，显然 TSP 问题是许多问题的图论上的抽象描述，解决好 TSP 问题有很大的意义。进一步，假设城市数为 n，城市 i 与城市 j 之间的距离为 $d_{ij}(i,j=1,2,\cdots,n)$，蚂蚁数为 m，城市 i 与城市 j 之间路径上在 t 时刻的信息素量为 $\tau_{ij}(t)$，初始时刻每条路径上的信息素为 $\tau_{ij}(0)$。蚂蚁 $k(k=1,2,\cdots,m)$ 在面临路径选择的时候按

照如下的转移概率来决定下一个要去的城市：

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{k\in allowed_k}[\tau_{ik}(t)]^{\alpha}[\eta_{ik}]^{\beta}} & \text{if } j\in allowed_k\\ 0 & \text{otherwise}\end{cases}$$

其中，$P_{ij}^{k}(t)$ 表示在 t 时刻蚂蚁 k 处于城市 i，选择城市 j 为下一个目标的概率，η_{ij} 表示启发式的参量，可以取距离的倒数，即 $\eta_{ij}=1/d_{ij}$，在搜索中此项是固定不变的。假设 $tabu_k$ 表示蚂蚁 k 已经访问过的城市列表，那么 $allowed_k=\{N-tabu_k\}$ 表示蚂蚁 k 还能去的城市集合。每只蚂蚁在两个城市之间移动一步之后，城市之间的路径上的信息素浓度更新，n 步之后更新为 $\tau_{ij}(t+n)=(1-\rho)\tau_{ij}(t)+\Delta\tau_{ij}$，其中 ρ 为挥发系数，$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}$ 为 m 只蚂蚁在路径上留下的信息素的总量。信息的更新过程与基于信息的路径概率选择过程，是一个正反馈的过程，加入信息素的挥发机制后，这样充分搜索之后保证了更优的解能够得到加强的同时又能够使得搜索不至于快速收敛到局部解。

4 进行函数优化的改进ACA 算法

从 ACA 算法对 TSP 问题求解的思想来看，ACA 算法就是用大量的 Ant 在搜索空间中进行局部搜索，反复地对路径上的信息素的高浓度的偏好选择，加强之后得到全局最优解。根据这种思想我们提出如下的函数优化改进 ACA 算法：

(1)通过对约束条件求解，我们可以得到满足约束条件的可行解所构成的搜索空间。这是一个 n 维的凸空间，记为 $\Omega=\{X_1,X_2,\cdots,X_i,\cdots,X_r\}$，其中，$\forall X_i\in\Omega$，$X_i$ 的各个分量满足 $L_{ij}\le x_{ij}\le U_{ij}$，$j=1,2,\cdots,n$。如果把各个分量的范围搜索等分成离散的点，那么空间 Ω 的大小 r 可以由分量 x_i 的划分粗细来确定。假设，x_i 被划分成了 N_i 份，那么 $r=\|\Omega\|=\prod_{i=1}^{n}N_i$，可见如果是高维情况，即决策变量 $X=(x_1,x_2,\cdots,x_n)^T$ 中的 n 较大，而且为了求得精确解将分量 x_i 的区间划分得很细，那么搜索空间将变得十分巨大，这也正是问题求解的困难所在，当然这种划分区间的方法也是许多优化算法所一般采用的方法。

(2)将每个分量的取值区间等分成 N 个点，那么 x_i 就有 N 种选择，n 维决策量 X 就有 N^n 种选择组合。并将每个分量的 N 个取值点看作 N 个城市 City。最开始将 m 个蚂蚁 Ant 随机地放到 x_1 的 N 中的 m 个 City 上。搜索开始后，Ant 按照转移概率 $P_{(i,j)}^{(i+1,k)}=\tau_{(i,j)}^{(i+1,k)}/\sum_{l=1}^{N}\tau_{(i,j)}^{(i+1,l)}$ 从 x_i 的一个 City(i,j) 移动到 x_{i+1} 的一个 City(i+1,k) 上，显然 $\sum_{l=1}^{N}P_{i,j}^{i+1,l}=1$。我们称 x_i 中的 N 个 City 组成一个层 L_i，那么一共有 n 个层。每只 Ant 从 L_1 开始，层层转移，一直到达 L_n，那么 Ant_k 走过的路径 $Tour_k$ 对应一个解 X_k，计算出对应的目标函数值 F^*，所有蚂蚁得出解后，更新 City(i,j) 与 City(i+1,k) 之间路径 $R_{(i,j)}^{(i+1,k)}$ 上的信息素为 $\tau_{(i,j)}^{(i+1,k)}(t+1)=(1-\rho)\tau_{(i,j)}^{(i+1,k)}(t)+\Delta\tau_{(i,j)}^{(i+1,k)}$，其中 ρ 为挥发系数，$\Delta\tau_{(i,j)}^{(i+1,k)}=\sum_{s=1}^{m}\Delta\tau_{(i,j)}^{(i+1,k)}|s=\sum_{s=1}^{m}F^{(s)}$，$Q>0$ 为 m 只蚂蚁在路径 $R_{(i,j)}^{(i+1,k)}$ 上留下的新增信息素的总量。值得一提的是，文献[7]提出在目标函数值上加上一个大正数，来保证函数值为正的，这个方法简单而有效。可见，当 Ant_s 所得出的解 X_s 所对应的目标函数值 $F^{(s)}$ 越小，那么 Ant_s 在 $R_{(i,j)}^{(i+1,k)}$ 上留下的信息素 $\Delta\tau_{(i,j)}^{(i+1,k)}$ 越大。这样就对搜索作出了指导。让 m 只 Ant 做多次搜索之后得出一个当时最优解 $X^{(1)}$，然后将 $X^{(1)}$ 的各个分量细化重新建立搜索空间。重复以上过程，依次得到 $X^{(1)},X^{(2)},\cdots$，当停机条件满足的时候得到最优解 X^* 输出。图 1 是多层搜索示例，每个分量被细分之后，建立一个层 Layer，那么蚂蚁在每个 Layer 上可能选择不同的点，这样通过所有的 Layer 后形成的通路就代表了一个解，当然多个解之间可以通过计算函数值来进行比较。本算法的具体步骤为：

(1)求解约束条件得出各分量的区间；

(2)将分量区间细化，建立搜索空间；

(3)判断精度是否满足要求，若满足，则输出最优解并结束，否则继续(4)；

(4)每个路径上设置信息量初始值；

(5)随机选择路径求解；

(6)每个蚂蚁按照选择概率来选择路径求解；

(7)若每个蚂蚁都找到解，则继续(8)，否则转(4)；

(8)将当前最优解的分量再细化，转(2)。

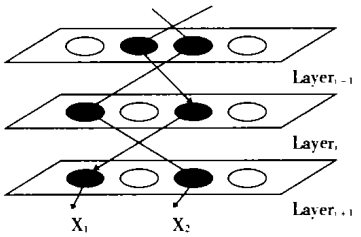

图 1 改进 ACA 算法多层搜索得出解 X_1 和 X_2

5 算法仿真

根据以上所讨论的用于函数优化的蚁群算法思想，进行算法仿真。测试环境为 OS:Windows XP，CPU:P4 1.5GHz，Memory:128MB，编译环境为 C 语言，VC++6.0。

计算实例 1：对于函数优化问题如式(1)

$$\begin{cases}\min f(x_1,x_2)=0.8/(x_1^2+x_2^2+1)+0.2(x_2^1+3x_2^2+1)\\ s.t\ x_1\in[-3,3]\ x_2\in[-5,5]\end{cases}\quad(1)$$

理论最优解为两个 $(F^*,x_1^*,x_2^*)_1=(0.8,1,0)$ 和 $(F^*,x_1^*,x_2^*)_2=(0.8,-1,0)$。

用 ACA 得到的结果为表 1，图 2 是计算结果的表示。

表 1 ACA 算法计算结果

F^*	x_1^*	x_2^*	i 平均时间
0.800013	1.000000	0.005612	<1
0.800001	-1.000000	-0.001530	<1

从表 1 可以看出，ACA 算法在相当短的时间内就得到了最优解，同理论最优解误差也相当小。而且 ACA 准确地得出两个解，而用直接的搜索则不能同时得到两个解。ACA 避免了一般直接搜索算法快速陷入局部最优解而只能得到一个解的缺点。

计算实例 2：对于函数优化问题如式(2)[7]

$$\begin{cases}\min f(x_1,x_2)=(x_1-1)^2+(x_2-2.2)^2+1\\ s.t\ x_1,x_2\in[-4,4]\end{cases}\quad(2)$$

理论最优解为 $(F^*,x_1^*,x_2^2)=(1,1,2.2)$。

表 2 为用 ACA 得到的 1 000 个结果中的最优解、平均解，

图3为计算结果的表示,图4为某次仿真计算中得到的目标函数值F*的结果序列,可见ACA很快就收敛得到最优解1°,并保持稳定。

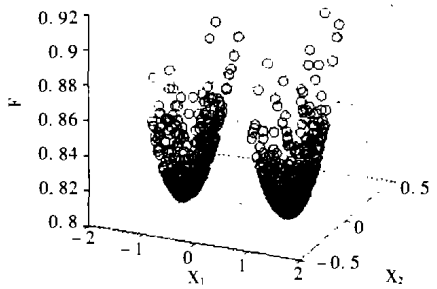

图2　ACA算法得到的两个解(1,0)和(-1,0)

表2　ACA算法计算结果

F*	x_1*	x_2*	$\bar{F}$	$\bar{x_1}$	$\bar{x_2}$	t平均时间
1.000002	0.998674	2.200184	1.048500	1.001200	2.196500	<1

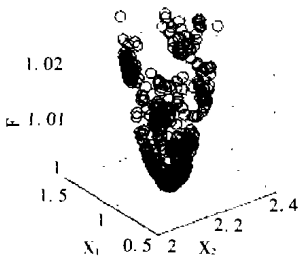

图3　ACA算法得到的解(1,2.2)

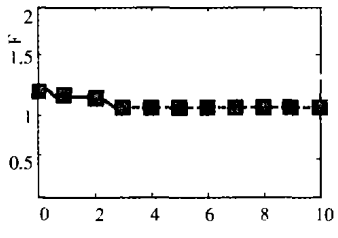

图4　ACA算法得到的目标值快速收敛于2.2,并保持稳定

6　结束语

与其他基于蚁群算法的函数优化算法[6~11]相比较,本算法首先就是让蚁群在解空间中作随机搜索,这其实是更贴近真实的蚁群行为的,即认为蚁群最开始随机搜索,同时对路径进行信息素的更新,让搜索空间充分地等概率地得到信息素的覆盖。这样就对搜索空间的解的分布情况有了一个初步的全局了解,然后按照前一步搜索所得出全局的解的分布情况,根据路径上的信息素进行概率选择,不断加强信息素,从而得到全局的最优解。这样,在求解中本ACA算法的收敛速度很快的同时又保证了多个全局最优解的同时得到,这也是本文算法与其他算法之间的区别所在。

ACA算法这种仿生的算法现在已经在许多应用中取得了很好的效果。本文旨在讨论一种改进的ACA算法思想以及在函数优化问题的最优解求解中的应用与具体算法,计算示例表明这种改进的ACA算法具有快速高效,并且能够得到多个全局最优解的优点,给具有高维度、大尺度解空间的问题求解提供了新的方法。蚁群搜索的并行、动态调整、鲁棒性得到了充分的应用。进一步的工作将着重研究新的信息素更新策略以及将此蚁群算法思想结合其他算法,如模拟退火算法、遗传算法等算法用于函数优化问题。

参考文献:

[1] Dorigo M, Maniezzo V, Colorni A. The Ant System: Optimation by a colony of Cooperating Agents[J]. IEEE Transactions on Systems, Man, and Cybernetics-Part B, 1996, 26(1): 1-13.

[2] Dorigo M, et al. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-66.

[3] Colorni A, Dorigo M, et al. Ant System for Job-Shop Scheduling[J]. Belgian Journal of Operations Research, Statistics and Computer Science, 1997, 1(1): 53-66.

[4] Solnon C. Ants Can Solve Constrained Satisfaction Problems[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(4): 347-357.

[5] Maniezzo V, Colorni A. The Ant System Applied to the Quadratic Assignment Problem[J]. IEEE Transactions on Knowledge and Data Engineering, 1999, 11(5): 769-778.

[6] 马良. 基于蚂蚁算法的函数优化[J]. 控制与决策, 2002, 17(11): 719-722.

[7] 高尚, 钟娟, 莫述军. 连续优化问题的蚁群算法研究[J]. 微机发展, 2003, 13(1): 21-22.

[8] 陈峻, 沈洁, 秦玲. 蚁群算法求解连续优化问题的一种方法[J]. 软件学报, 2002, 13(12): 2317-2323.

[9] 汪镭, 吴启迪. 蚁群算法在连续空间寻优问题求解中的应用[J]. 控制与决策, 2003, 18(1): 45-48.

[10] 熊伟清, 余舜浩, 魏平. 用于求解函数优化的一个蚁群算法设计[J]. 微电子学与计算机, 2003, (1): 23-25.

[11] 魏平, 熊伟清. 用于一般函数优化的蚁群算法[J]. 宁波大学学报, 2001, 14(4): 52-55.

作者简介:

唐泳(1979-),男,硕士,研究方向为智能优化和决策;马永开(1963-),男,教授,硕士,研究方向为预测和决策;唐小我(1955-),男,教授,博士生导师,博士,研究方向为预测和决策。

(上接第87页)点。其中(-0.089842,0.7126)和(0.0898,-0.7126)为全局最小点,最小值为-1.031628;最大值是(-3,-3)=405.9。

求解函数极小值:

BGA得到f(-0.089842,-0.712656)=-1.031628

改进算法得到f(-0.89686,0.712788)=-1.031628

求解函数极大值:

BGA得到f(2.994226,2.999634)=404.106721

改进算法得到f(-3.000000,-3.000000)=405.9

4　结论

从多个数值实验的比较分析得出,改进后的实数编码遗传算法在求解函数的极值时不仅能在短时间内搜索到高精度的全局最优解,而且稳定性也非常好,均未出现陷入局部极值点的现象,比起二进制的遗传算法更是节省了大量的存储空间、缩短了搜索时间、提高了优化精度。表明遗传算法在解决实际问题时,合适的编码以及恰当的变化算子设计对解决问题是非常重要。

参考文献:

[1] Zbigniew Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs[M]. 北京:科学出版社, 2000.

[2] 李敏强, 寇纪松, 林丹, 等. 遗传算法的基本理论与应用[M]. 北京:科学出版社, 2002.

[3] 叶晨洲, 杨杰, 黄欣, 等. 实数编码遗传算法的缺陷分析及其改进[J]. 计算机集成制造系统—CIMS, 7(5): 28-32.

[4] 张文修, 梁怡. 遗传算法的数学基础[M]. 西安:西安交通大学出版社, 2000.

作者简介:

魏平(1965-),女,副教授,硕士,研究方向为进化计算、数据库;熊伟清(1966-),男,副教授,硕士,研究方向为进化计算、软件工程。