第36卷第6期 四川大学学报(工程科学版) Vol.36 No.6
2004年11月 JOURNAL OF SICHUAN UNIVERSITY (ENGINEERING SCIENCE EDITION) Nov. 2004

文章编号:1009-3087(2004)06-0117-04

用于连续函数优化的蚁群算法

陈 烨
(四川大学 电气信息学院,四川 成都 610065)

摘 要:为了用蚁群算法来解决连续优化问题,该算法将函数优化问题中生成解的过程转化为蚁群每前进一步就选择一个十进制数字并以此来生成一个十进制串的过程。与普通蚁群算法相同,蚁群在选择数字的过程中将一定量的信息记录在每条选择的路径上以改变下一次蚁群选择各个数字的概率。实验数据表明,文中的函数优化算法能比遗传算法以及其他用于连续优化的蚁群算法更快地找到更好的解。这种算法为蚁群算法求解连续优化问题提供了一种新的方法。

关键词:蚁群算法;旅行商问题;连续函数优化
中图分类号:TP301.6 **文献标识码**:A

Ant Colony System for Continuous Function Optimization

CHEN Ye
(School of Electrical Eng. and Info., Sichuan Univ., Chengdu 610065, China)

Abstract: Based on Ant Colony System, a new algorithm for continuous function optimization is propose. Each ant makes a selection from ten decimal numbers whenever it takes a step in this algorithm. And in this way a solution for the function optimization problem can be built. The same as general Ant Colony System, the ants will change the information left on their paths, so that the probability that an ant chooses a number in a step next time can be changed to lead the ant to a better path. The experimental result shows that this new algorithm can find a better solution for function optimization problem than genetic algorithms and other ant colony system for continuous optimization. This new algorithm presents a new way to solve continuous optimization problems.

Key words: Ant Colony System; traveling salesman problem; continuous function optimization

蚁群算法(Ant Colony System)已被许多研究证明是一种有效的离散优化算法,目前已用于求解TSP、QAP等各种离散优化问题[1],得到了很好的结果,其中求解许多问题的结果都优于遗传算法、退火算法等启发式随机搜索算法。既然蚁群算法在离散优化方面取得了这么好的结果,人们自然会想到,蚁群算法能不能用于求解连续优化问题并取得同样好的效果。因此,很多学者作了这方面研究。由于蚁群算法最初是针对离散优化问题提出的,其中使用到的信息素分布就是以离散的方式存储的,为了用蚁群算法来解决连续优化问题,总的来说,大致有以下两种途径:一是仍然将信息素离散分布到各条路径上,因此就将解空间划分为几个离散的区间,并用区间中的一个特定的解来表示该区间的优良程度,选出较好区间后,对每个区间再使用优化算法来优化。另一种途径是,蚂蚁直接在连续的解空间上取值,因此就必须使用一定的函数来表示连续的解空间上信息素的分布[2]。

文中将提出一种新的方法,在这种方法中,信息素仍然离散的分布到各条路径上,但是对解空间的划分方式与现有的其他算法不同,下面将具体介绍

收稿日期:2003-06-12
作者简介:陈 烨(1984-),男. 研究方向:演化计算.

这种新算法,后面的实验数据将证明本文算法的优良性质。

1 用于连续函数优化的蚁群算法

1.1 一元连续函数优化

对于任何一个连续函数优化问题,都可以通过一定的变换而成为一个在[0,1]上的函数最小化问题 $minf(x)+C$,其中 $x\in[0,1]$。加上一个常数 C 以使函数值大于0。对于端点值,可以通过直接与除去端点计算出的最小值比较的办法确定是否为最小,因此下面不考虑端点值。

设问题要求自变量精确到小数点后 d 位,则自变量 x 可以用 d 个十进制数来近似表示,就可以构造如下 $d\times 10+2$ 个"城市"。这些城市分为 $d+2$ 层。其中首尾两层分别仅含一个城市:一个为起始城市,一个为终止城市。中间 d 层,从左往右分别表示自变量的十分位、百分位……这些城市中,只有 $k-1$ 与 k 层($k\in[2,d+2]$)之间的各个城市有连接通路。记 $k-1$ 层中代表十进制数 a 的城市与 k 层代表十进制数 b 的城市之间的连接上残留的信息量为 τ_{ab}^{k}。蚂蚁 n 在一次循环中的第 m 步所在的城市用 $T(n,m)$ 表示。设蚂蚁总数为 N_0。

首先用一个较小的值 τ_0 初始化所有的 τ_{ab}^{k}。让每只蚂蚁的第一步为0,即令 $T(n,1)=0(n=1,2,\ldots,N_0)$。然后,就为每一只蚂蚁选择路径。若蚂蚁 n 当前所在的城市为 $T(n,k-1)=a$,根据如下公式选择每只蚂蚁下一步应该到达的城市:

$$T(n,k)=\begin{cases}\arg\max\{\tau_{ab}^{k}\}, & \text{如果 } q<Q_0\\ S_r, & \text{否则}\end{cases}\tag{1}$$

其中,q 为随机数;Q_0 是一个[0,1]上的常数,用于确定伪随机选择的概率;S_r 表示用伪随机选择来确定下一步要走的城市,也就是根据下式计算选择下一层中每一个城市的概率,然后按此概率用遗传算法中的转盘式选择法确定要选择的城市:

$$p(a,b)=\tau_{ab}^{k}/\left(\sum_{x=0}^{9}\tau_{ax}^{k}\right)\tag{2}$$

其中,$p(a,b)$ 表示从当前城市 a 转移到下一层的城市 b 的概率。由于本算法中仅允许蚂蚁有上一层城市向下一层转移,所以这个公式与普通蚁群算法的转移概率计算公式有所不同。

当每只蚂蚁按上面的公式到达了 $d+1$ 层时,都将转移到 $d+2$ 层的唯一的城市0。

蚂蚁在城市上建立路径的过程中,要不断地在经过的路径上按公式(3)减弱上面残留的信息,这样可以减小下一只蚂蚁选择同样路径的概率,除非经过多次循环后已确定一条极优的路径。这个过程叫做残留信息的局部更新。

$$\tau_{T(n,k-1),T(n,k)}^{k}\leftarrow(1-\rho)\times\tau_{T(n,k-1),T(n,k)}^{k}+\rho\tau_0\tag{3}$$

其中,$\rho\in(0,1)$ 为常数,表示路径上残留信息减弱的速度。

当所有蚂蚁都按上面的步骤完成了一次循环。这时就对路径上的信息进行全局更新。首先对蚂蚁选择的路径解码,计算出蚂蚁 n 对应的自变量值:

$$x(n)=\sum_{k=2}^{d+1}T(n,k)\times 10^{1-k}\tag{4}$$

计算每只蚂蚁对应的函数值,并选择出函数值最小的蚂蚁:

$$n_{\min}=\arg\min\{f(x(n))\}\tag{5}$$

对这只最优蚂蚁经过的路径按下式做全局更新:

$$\tau_{ij}^{k}\leftarrow(1-\alpha)\times\tau_{ij}^{k}+\alpha\times f(n_{\min})^{-1}\tag{6}$$

其中,$i=T(n_{\min},k-1),j=T(n_{\min},k),k\in[2,d+2]$,$\alpha$ 为(0,1)上的常数。

至此就完成了一个循环。反复进行上面的步骤直到达到指定的循环次数或得到的解在一定循环次数后没有改进。

文中提出的求解一元连续函数优化问题的蚁群算法具体描述如下:

1) 初始化;

2) 将所有蚂蚁置于初始城市;

3) 对所有的 $k-1$ 到 k 层城市执行步骤4) ~ 8);

4) 对每只蚂蚁执行步骤5) ~ 6);

5) 根据公式(1)和(2)选择蚂蚁在第 k 层应该到达的城市;

6) 每只蚂蚁选择城市后都立即按公式(3)执行局部更新规则;

7) 根据公式(4) ~ (6)评选出最优蚂蚁并执行全局更新规则;

8) 判断是否满足终止条件,满足则输出结果结束计算。

1.2 多元连续函数优化

对于多元连续函数的优化问题,设自变量由 n_x 个分量组成,并要求自变量的每一个分量都精确到小数点后 d 位,则可构造一副由 $n_x\times d+n_x+1$ 层城市组成,且第1、$d+2$、$2d+3$、……$n_x\times d+n_x+1$ 层由1个标号为0的城市组成,其余层都由标号为0

到9的10个城市组成。第$(k-1)\times(d+1)+2$到$k\times(d+1)$层($k=1,2,\ldots,n_x$)表示自变量的第k个分量。其余层都是辅助层。解码时,就对各分量对应的层分别解码。

采用这种方法,每个自变量分量的最后一位与下一个分量的第一位之间都有辅助层隔开,因此前面一个分量的末位就不会影响后面一个分量首位。

除了这一点以外,其余部分都与一元连续函数的优化方法相同,这里就不再详细介绍了。

2　算法有效性分析

作者的算法是以蚁群算法为基础的,和用于其它各种问题的蚁群算法一样,能够在搜索过程中避开局部极值点,且对算法的初始群体不敏感。

在文中的蚁群算法中,每只蚂蚁在选择路径时都没有使用确定性的选择策略,而只是以一定的概率选择当前最好的路径。这样,即使蚁群中的所有蚂蚁都同时进入了局部极值,在下一次选择中,这些蚂蚁也会以一定的概率选择其他的路径继续探索。这就避免了在搜索过程中陷入局部极值。同样也是由于采用了非确定性的选择策略,不管蚁群的初始分布是否对应于较好的解,也不管蚁群是否初始分布于局部极值点上,蚁群都会不断地向全局最优解逼近。这都是与普通蚁群算法类似的。

3　实验结果

为了证明本算法的有效性,用它求解了多个连续函数优化问题,并与基本遗传算法、佳点集遗传算法[3]以及基于浓度模型的遗传算法[4],说明了文中算法能够更好地找到最优解。

下面的所有实验中,采用的参数取值如下:

$\alpha=0.8$, $\rho=0.8$, $Q_0=0.8$, $\tau_0=0.01$, $d=7$, $N_0=20$,运行循环次数:1000。

(虽然这里只取小数点后7位数字,但由于某些函数的定义域不是[0,1],所以在求解该函数最值前必须将自变量进行一些变换,这导致实际计算时使用的自变量有可能包含了更多的位数。因此后面的结果中将保留较多的位数,以确保用这里列出的自变量能计算出给出的结果。)

1)$\max F_1(x)=|(1-x)x^2\sin(200\pi x)|$,其中,$x\in[0,1]$;

2)Branin Function:

$\max F_2(X)=-(x_2-bx_1^2+cx_1-d)^2-e(1-f)\cos x_1-e$,其中,$b=5.1/4\pi^2$, $c=5/\pi$, $d=6$, $e=10$, $f=1/8\pi$, $x_1\in[-5,10]$, $x_2\in[0,15]$。

对以上2个函数计算结果的对比如下。

表1　$F_1(x)$的最好计算结果对比

Tab.1　The comparison between the results of $F_1(x)$ generated by different algorithms

作者算法		文献[4]的改进算法		佳点集遗传算法	
代数	结果	代数	结果	代数	结果
138	0.148147453125				
48	0.148147447778438	8	0.148092	20	0.148148
6	0.148140985544726				

注:佳点集遗传算法的结果0.148148对应的自变量为$x=0.667500$,而本文算法的结果0.148147453125对应的自变量也为$x=0.667500$,这应该是舍入误差

表2　$F_2(X)$的最好计算结果对比

Tab.2　The comparison between the results of $F_2(X)$ generated by different algorithms

本文算法			佳点集遗传算法	
代数	自变量X	结果	代数	结果
4	3.1502125,2.2525095	-0.39849302239805	35	-0.401337
7	3.1513465,2.283861	-0.398614892807521		
71	9.42373,2.4646665	-0.397981926517719		
32	-3.14525,12.279453	-0.397970397512475		

注:文献[5]中给出的$f=-F_2(X)$的最小值为$f(-3.142,2.275)=f(3.142,2.275)=f(9.425,2.425)=0.398$可推知已知的$F_2(X)$的最大值应该为-0.398

以上是用作者提出的连续蚁群算法与其他类型的算法相比较的结果。从以上实验以及没有在这里列出的实验结果可知,文中提出的算法明显优于普通的遗传算法,在某些问题上略优于佳点集遗传算法和基于浓度的遗传算法。

以前也有多种连续优化的蚁群算法被提出,在文献[6]中使用了$\min f(x_1,x_2)=(x_1-1)^2+(x_2-2.2)^2+1$,作为测试函数,仅得到了$f(1.0366,2.1988)=1.13395744$这样的结果,而本文算法多次计算的结果平均值已经达到了1.00002294。文献[7]中也针对连续优化问题提出了"求解一般函数优化问题的蚁群算法"。但是,它的效果不是很好,用于测试它的函数的都比较简单,用本文算法很快就能得到最优解。还有一些与其他算法比较的结果这里就不详细列出了。

本文提出的新算法不仅能够得到很好的解,而且求解速度很快。从得到最优解需要的计算代数上看,新算法似乎与佳点集算法等改进遗传算法的速度差不多,然而,由于新算法是基于基本蚁群算法的,没有增加任何附加算子,因此,实际执行速度比经过改进的很多算法都要快。另一方面,新算法中,

蚂蚁每走一步只需在10个城市中选择一个,对于两个自变量的函数来说,即使要求精确到小数点后10位,也仅需选择10次就完成一个循环。所以,该算法的执行速度很快,在做实验的过程中发现,在设置的蚂蚁数与遗传算法的个体数相等的情况下,它执行速度约为基本遗传算法的2.5倍。而实际计算中,仅取了20只蚂蚁,而遗传算法通常要取50个个体。因此,本文算法的实际计算速度是非常快的。

4 结 语

本文提出了一种有效的求解连续函数优化问题的蚁群算法,实验证明这种新算法继承了蚁群算法在求解离散优化问题上的优点。该算法主要思想是使用离散的点来近似表示连续函数的自变量,然后再将这些离散的点构造成一层一层的城市,最后通过按特定规则运行的蚁群算法来求解。它虽然是建立在普通蚁群算法的基础上,但已取得了接近甚至超过目前很好的改进型遗传算法的结果。若与现有的蚁群算法的改进方案相结合,应该能够取得更好的结果。

参考文献:

[1]Macro Dorigo, Gianni Di Caro, Luca M Gambardella. Ant algorithms for discrete optimization[J]. Artificial Life, 1999, 5(3): 137~172.

[2]Wang Lei, Wu Qidi. Ant system algorithm in continuous space optimization[J]. Control and Decision, 2003, 18(1): 45~48. [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.]

[3]Zhang Ling, Zhang Bo. Good point set based genetic algorithm [J]. Chinese Journal of Computers, 2001, 24(9): 917~922. [张 铃, 张 钹. 佳点集遗传算法[J]. 计算机学报, 2001,24(9):917~922.]

[4]Zhang Ling, Zhang Bo. Research on the mechanism of genetic algorithms[J]. Journal of Software, 2000, 11(7): 945~952. [张 铃, 张 钹. 遗传算法机理研究[J]. 软件学报, 2000,11(7):945~952.]

[5]Wang Ling. Intelligent optimization algorithms with Applications [M]. Tsinghua University Press, 2001. [王 凌. 智能优化算法及其应用[M].北京:清华大学出版社,2001.]

[6]Gao Shang, Zhong Juan. Mo Shujun. Research on ant colony algorithm for continuous optimization problem[J]. Microcomputer Development, 2003, 13(1): 21~22. [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.]

[7]Wei Ping, Xiong Weiqing. Ant colony algorithm for general function optimization problems[J]. Journal of Ningbo University (NSEE), 2001, 14(4): 52~55. [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(理工版), 2001,14(4):52~55.]

(编辑 张 琼)

·快 讯·

通过检索Ei数据库,本刊2004年被Ei数据库已摘录4期,共92篇论文,摘录率81.4%。