自适应双向菌群优化算法*

胡桂武[1,2a,2b]，陈建超[1]，杜小勇[2a,2b]

(1. 广东商学院 数学与计算科学学院，广州 510320；2. 中国人民大学 a. 教育部数据工程与知识工程重点实验室；b. 信息学院，北京 100872)

摘　要：提出了自适应双向菌群优化算法，应用聚类思想将趋化步长进行自适应调整，提高算法的局部搜索能力，引入双向游动机制，提高了算法的搜索效率和速度。针对 10 个复杂 Benchmark 函数进行了数值优化实验，其结果表明，在所有测试函数中，该算法在搜索能力和稳定性等方面优于其他典型算法的比率达到 60% ~90%，验证了算法的有效性。

关键词：菌群优化算法；趋化步长；聚类；双向

中图分类号：TP301.06　　**文献标志码**：A　　**文章编号**：1001-3695(2012)10-3645-03

doi:10.3969/j.issn.1001-3695.2012.10.011

Adaptive bidirectional bacterial foraging optimization algorithm

HU Gui-wu[1,2a,2b], CHEN Jian-chao[1], DU Xiao-yong[2a,2b]

(1. *School of Mathematics & Computational Science, Guangdong University of Business Studies, Guangzhou* 510320, *China*; 2. *a. Key Laboratory of Data Engineering & Knowledge Engineering for Ministry of Education, b. School of Information, Rennin University of China, Beijing* 100872, *China*)

Abstract: This paper proposed on adaptive bidirectional bacterial foraging optimization(ABBFO), which had self-adaptive chemotactic step size based clustering, could improve algorithm's local search ability. Bidirectional swarm was devised to enhance the efficiency and speed of algorithm, 10 complex Benchmark functions have been tested. The simulation shows that the ABBFO has better search ability and stability than other typical algorithm up to 60% ~90% among test functions. The comparisons also shows ABBFO is an effective optimization.

Key words: bacterial foraging optimization algorithm; chemotactic step size; clustering; bidirection

0　引言

Passino 基于 Ecoli 大肠杆菌在人体肠道内吞噬食物的行为，提出了一种新型群智能优化方法——菌群优化(bacterial foraging optimization, BFO)[1]算法。该算法具有潜在的并行性、随机性、结构简单、易实现等特点，吸引了不少学者对它进行研究，并且取得了较好的效果，已经成为智能计算领域的研究热点[2~5]。然而，BFO 作为一种新的优化算法还不够完善。研究证明，基本 BFO 的局部搜索能力不强，解的精度和稳定性方面也存在一些问题。

为了提高菌群优化算法的性能，通过自适应修改趋化步长提高了算法的局部搜索能力。另外，通过设置一个双向游动机制改善算法的搜索效率和收敛速度，得到了自适应双向菌群优化(adaptive bidirectional bacterial foraging optimization, ABBFO)算法，并将其应用于复杂函数的优化问题求解。仿真验证了 ABBFO 算法求解复杂优化问题的可行性和有效性。

1　基本菌群优化算法

BFO[1,2]算法的基本思路是从初始化一组随机解开始，将细菌的位置表示为问题的潜在解，通过细菌的趋化操作，实现细菌下一步趋向更为有利的生存环境，获得较优的位置；通过复制操作，保留优秀个体淘汰较差个体，从而实现整个细菌群体收敛到局部最优；通过迁移操作，可以在一定程度上避免算法陷入局部最优。简单地说，BFO 可以通过四个优化过程来实现，即趋药性操作、聚集操作(趋药性操作和聚集操作组合成趋化操作)、复制操作和迁徙操作。

1.1　趋药性操作

BFO 中的趋药性操作可描述为如下：

$$\theta^i(j+1,k,l)=\theta^i(j,k,l)+C(i)\frac{\Delta(i)}{\sqrt{\Delta^{\mathrm{T}}(i)\Delta(i)}} \tag{1}$$

其中：$\theta^i(j,k,l)$表示第 i 个体在第 j 代趋药性循环、第 k 代复制循环、第 l 代迁移循环时的位置；$\Delta(i)$表示方向调整后选定的方向向量，向量中的元素属于区间$[-1,1]$；$C(i)$表示前进的步长。

收稿日期：2012-04-06；**修回日期**：2012-05-13　　**基金项目**：国家自然科学基金资助项目(60873017)；国家自然科学基金资助项目(中德合作)(61111130183)

作者简介：胡桂武(1970-)，男，教授，博士(后)，中国计算机学会高级会员，主要研究方向为数据挖掘、优化计算等(guiwuhu@gdcc.edu.cn)；陈建超(1979-)，男，广东潮州人，博士，主要研究方向为人工智能等；杜小勇，男，教授，博导.

1.2 聚集操作

BFO 的聚集操作是通过个体之间的相互作用来实现的。对于最小值问题,设第 i 个细菌个体的信息表示为 $X_i=(x_1,x_2,\cdots,x_p)$,则其聚集操作的数学表达式为

$$J_{cc}(\theta,P(j,k,l))=\sum_{i=1}^{S}J_{cc}(\theta,\theta^i(j,k,l))=\sum_{i=1}^{S}[-d_{\text{attractant}}\exp(-w_{\text{attractant}}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2)]+\sum_{i=1}^{S}[h_{\text{repellant}}\exp(-w_{\text{repellant}}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2)] \tag{2}$$

其中:$d_{\text{attractant}}$、$W_{\text{attractant}}$分别为引力的深度和宽度;$h_{\text{repellant}}$、$W_{\text{repellant}}$分别为斥力的高度和宽度;θ_m^i 为个体 i 的第 m 个分量;θ_m 为整个群体中其他个体的第 m 个分量。此公式代表整个细菌群体在细菌 i 所处位置 θ_m 处所产生的作用力之和。因此,此时第 i 个个体的适应度计算公式变为

$$J(X_i)=J(X_i)+J_{cc}(X_i) \tag{3}$$

因此聚集操作就是通过上述公式来对适应值 $J(X_i)$ 进行修正,使得细菌达到聚集的目的。

1.3 复制操作

在执行完指定步数的趋化操作之后,根据各个细菌的健康度进行排序,健康度定义为

$$\text{heath}(i)=\sum_{i=1}^{\text{num}}J_i \tag{4}$$

其中:heath(i)表示第 i 个细菌的当前健康度;J_i 表示第 i 个细菌的适应度函数值;num 表示到当前位置之时,第 i 个细菌所完成的趋化操作的数目。然后按照式(4)所定义的健康度函数对整个菌群按照健康度进行排序。健康度较高的细菌有权进行繁殖行为,生成子代;健康度不高的细菌没有繁殖权利,它们将死亡。

目前通常的操作是:群体中要被淘汰的个体个数设为 $S_r=S/2$(S 为群体的规模),首先对群体中的个体按其优劣进行排序,而后将排在较靠后的 S_r 个个体删除,剩下的 S_r 个个体进行自身的拷贝。

1.4 迁徙操作

迁移操作是按照给定的概率 P_{ed} 发生的,如果某个体满足迁移操作的条件,那么就将此个体删除,重新生成一个新的个体,相当于将原来的个体移到了一个新的位置。

2 自适应双向菌群优化算法

BFO 的趋化步长 $C(i)$ 是一个关键参数,在算法的进化后期,固定的步长容易漏掉可能的局部最优点。由式(1)(2)可知,BFO 的核心操作趋化是基于距离的,因此引进局部邻居的机制,应用聚类的思想,按照式(5)自适应地修改趋化步长,提高算法的局部搜索能力。另外,为了提高搜索的效率,提供了一种双向游动策略,当个体向一个方向随机搜索后效果没有变优时,则马上反向搜索,从概率角度来说效率高于原来的机制。

$$C(i)=\left(\sum_{n=1}^{n(i)}\sqrt{\sum_{m=1}^{P}(\theta_m^i-\theta_m^{i\_n})^2}/n(i)\right)\times\text{rand}(a,b) \tag{5}$$

其中:$n(i)$是第 i 个个体选择最近的邻居数目;a、b 属于区间(0,1);$C(i)$实际上是以第 i 个个体为中心的局部种群到中心的距离均值乘以一个随机数,能提高算法局部探索或开采能力。

ABBFO 算法的基本实现步骤如下:

a)初始化相关参数和规模为 S 的种群,记个体 θ^i 的健康值为 health_i。

b)令 $l=0$,开始迭代次数为 N_{ed}的迁徙循环,在循环的每次迭代中,l 递增 1,循环体包含了步骤 c)~g)。

c)令 $k=0$,开始迭代次数为 N_{re}的复制循环,在循环的每次迭代中,先让 k 递增 1,health_i 都初始化为 0,循环体包含了步骤 d)~f)。

d)令 $j=0$,开始迭代次数为 N_c 的趋化循环,在循环的每次迭代中,j 递增 1,循环体为步骤 e)。

e)依次对每个粒子 $\theta^i(i=1,2,\cdots,S)$ 执行从步骤(a)~(e)的趋化操作:

(a)计算 θ^i 的当前适应值 $J(i,j,k,l)$、θ^i 与群体的吸引与排斥综合值 $J_{cc}(\theta_i)$。

(b)按照式(5)计算 θ^i 的游动步长。

(c)选择 θ^i 的游动方向,先随机选择并单位化一个向量 Δ^i,执行如下的逻辑判断:

if(θ^i 按 $C(i)$ 和 Δ^i 游动一步能到达一个更好的位置)
then{把 Δ^i 确认为 θ^i 的游动方向,转到步骤(d);}
else{$\Delta^i=-\Delta^i$;
if (θ^i 按 $C(i)$ 和 Δ^i 游动一步能到达一个更好的位置)
then{把 Δ^i 确认为 θ^i 的游动方向,转到步骤(d);}
else{结束 θ^i 的本次趋化操作,跳到步骤(e)。}
}

(d)让 θ^i 按选定的方向和步长进行最多 N_s 次的游动,在每次游动时,先尝试走出一步,得到 θ^i 的临时位置,计算临时适应值 $J'(i,j,k,l)$、临时群体吸引与排斥综合值;如果 $J'(i,j,k,l)<J(i,j,k,l)$,则确认临时位置为 θ^i 的新位置,更新 $J(i,j,k,l)$和 $J_{cc}(\theta_i)$;否则放弃这个临时位置,并结束 θ^i 的本次趋化操作。

(e)把 θ^i 当前的 $J(i,j,k,l)$和 $J_{cc}(\theta_i)$累加给 health_i。

f)进行一次复制操作,把 S 个粒子按 health_i 从小到大排序,把大的一半粒子丢弃,并把小的一半粒子复制一份出来,构成数量为 S 的新的种群。

g)进行一次按概率迁徙,即对每个粒子,在(1,0)内产生一个随机数 r_i,如果 $r_i<P_{\text{ed}}$,则丢弃 θ^i,重新产生一个新的粒子代替 θ^i。

3 实验与分析

3.1 实验函数和相关参数

为了考察 ABBFO 的性能,将它与 BFOA[3]、HPSO-TVAC[6]、EA[7]、BSO[8]和 PSO-BFO 算法进行了比较。选择 10 个复杂的 Benchmark 函数 $f_1\sim f_{10}$[3]进行数值实验,函数的具体描述如表 1 所示。

表 1　Benchmark 函数一览表

function	mathematical representation	range of search	theoretical optima
sphere function(f_1)	$f_1(\vec{x})=\sum_{i=1}^{D}x_i^2$	$(-100,100)^D$	$f_1(\vec{0})=0$
rosenbrock's function (f_2)	$f_2(\vec{x})=\sum_{i=1}^{D-1}(100(x_i^2-x_{i+1})^2+(x_i-1)^2)$	$(-100,100)^D$	$f_2(\vec{1})=0$
rastrigin's function(f_3)	$f_3(\vec{x})=10D+\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i))$	$(-10,10)^D$	$f_3(\vec{0})=0$
griewank's function(f_4)	$f_4(\vec{x})=\sum_{i=1}^{D}\frac{x_i^2}{4000}-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$(-600,600)^D$	$f_4(\vec{0})=0$
ackley's function(f_5)	$f_5(\vec{x})=-20\times\exp[-0.2\sqrt{\frac{1}{D}\times\sum_{i=1}^{D}x_i^2}]-\exp[\frac{1}{D}\times\sum_{i=1}^{D}\cos(2\pi x_i)]+20+e$	$(-32,32)^D$	$f_5(\vec{0})=0$
step function(f_6)	$f_6(\vec{x})=\sum_{i=1}^{D}([x_i+0.5])^2$	$(-100,100)^D$	$f_6(\vec{p})=0,-\frac{1}{2}<p_i<\frac{1}{2}$
schwefel's problem2.22(f)	$f_7(\vec{x})=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	$(-500,500)^D$	$f_7(\vec{0})=0$
shekel's Foxholes(f_8)	$f_8(\vec{x})=\left[\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right]^{-1}$	$(-65.536,65.536)^2$	$f_8(-32,32)=0.998$
six-Hump Camel-Back function(f_9)	$f_9(\vec{x})=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^6$	$(-5,5)$	$f_9(0.08983,-0.7126)=f_9(-0.08983,0.7126)=-1.0316285$
Goldstein-Price function(f_{10})	$f_{10}(\vec{x})=\{1+(x_0+x_1+1)^2(19-14x_0+3x_0^2-14x_1-6x_0x_1+3x_1^2)\}\{30+(2x_0-3x_1)^2\times(18-32x_0+12x_0^2+48x_1-36x_0x_1+27x_1^2)\}$	$(-2,2)^2$	$f_{10}(0,-1)=3$

为保证算法的可比性,各算法取相同的参数。其中,种群规模 $S=50$;变量维数($f_1\sim f_7$) dim = 100,$f_8\sim f_{10}$维数 dim = 2;最大迁徙代数 $N_{ed}=4$;最大复制代数$N_{re}=16$;最大趋化代数 $N_c=100$;最大前进步数 $N_s=12$;驱散概率 $P_{ed}=0.25$;$d_{attractant}=0.1$; $w_{attractant}=0.2$, $w_{repellant}=10$, $h_{repellant}=0.1$, $n(i)=6$, $a=0.07$, $b=0.3$。

终止条件是函数评估次数大于 10^5 或者优化值低于预先设定的($f_1\sim f_7$: 0.001, f_8: 0.998, f_9: -1.0316, f_9: 3.00)。表 2 为 ABBFO、BFOA、HPSO-TVAC、EA 和 BSO 这五种算法对$f_1\sim f_{10}$每个函数连续 50 次运算的统计结果,评价的标准包括平均最优值和标准差。表 2 中的数据除了 ABBFO 之外,其余都引自于文献[3]。表 2 的统计结果表明,所对比的五种算法中,ABBFO 整体性能最好,ABBFO 在所有函数中所得到的最优均值和方差都优于另外四种算法,BFOA、HPSO-TVAC、EA 和 BSO 占所有函数的比例得到 60%。具体如下:

a) ABBFO 对f_1、f_2、$f_4\sim f_{10}$函数优化结果从最优均值和标准差方面来分析,远远优于算 BFOA 占测试函数总数的 90%。

b) ABBFO 对f_1、f_2、f_4、f_6、$f_8\sim f_{10}$函数优化结果从最优均值和标准差方面来分析,远远优于算法 HPSO-TVAC,占测试函数总数的 70%。

c) ABBFO 对f_1、f_4、f_6、$f_8\sim f_{10}$函数优化结果从最优均值和标准差方面来分析,远远优于算 EA,占测试函数总数的 60%。

d) ABBFO 对f_1、f_4、f_6、$f_8\sim f_{10}$函数优化结果从最优均值和标准差方面来分析,远远优于算 BSO,占测试函数总数的 60%。

总体来说,ABBFO 算法在求解精度和稳定性方面都明显优于其他四种算法,表明了 ABBFO 在求解高维和复杂函数优化问题上的可行性和有效性。

表 2　ABBFO、BFOA、HPSO-TVAC、EA 和 BSO 的性能比较

functions	dim	maximum no. of FES	mean best value (standard deviation)				
			ABBFO	BFOA	HPSO-TVAC	EA	BSO
f_1	30	10^5	0.001 (0)	0.084 (0.0025)	0.065 (0.0534)	0.036 (0.001)	0.056 (0.0112)
f_2	30	10^5	43.674 (50.662)	58.216 (14.3254)	706.263 (951.9533)	31.738 (3.6452)	15.471 (2.655)
f_3	30	10^5	134.214 (22.645)	17.5248 (9.8962)	34.837 (10.128)	3.797 (0.8241)	13.7731 (3.9453)
f_4	30	10^5	0.001 (0)	0.3729 (0.0346)	0.2175 (0.1953)	0.2684 (0.3616)	0.2565 (0.1431)
f_5	30	10^5	1.4373 (0.5412)	2.3243 (1.8833)	0.5684 (0.1927)	0.6059 (0.3372)	0.5954 (0.1246)
f_6	30	10^5	0.001 (0)	2.0802 (0.00342)	0.7752 (0.4531)	0.001 (0)	0.4852 (0.28271)
f_7	30	10^5	3.1133 (1.9202)	4.6354 (2.7753)	2.4861 (2.3375)	0.0642 (0.7681)	0.9043 (0.4186)
f_8	2	10^5	0.9980 (0)	1.056433 (0.01217)	0.9998323 (0.00537)	0.9998329 (0.00382)	0.9998017 (0.00825)
f_9	2	10^5	-1.031628 (0)	-0.925837 (0.000827)	-1.029922 (1.382)	-1.031149 (2.527)	-1.031242 (0.00759)
f_{10}	2	10^5	3.0000 (0)	3.656285 (0.109365)	3.1834435 (0.2645)	3.146090 (0.06237)	3.443712 (0.007326)

其中:dim 表示维数;maximum no. of FES 表示函数评估次最大次数;mean best value(standard deviation)表示最优值均值(标准差)

4　结束语

本文提出了新的菌群优化算法——自适应双向菌群优化算法,利用基于空间的聚类思想,自适应地调整趋化步长,提高了算法的局部搜索能力;引进双向搜索策略,提高了算法搜索的效率,测试结果表明了该算法的有效性。同时,这种进化机制也为群体智能优化提供了一个改进的思路,至少对各种不同 BFO 是一种有意义的尝试。下一步的工作是进行理论上的分析和将该算法应用于求解离散的优化问题。(下转第 3668 页)

(上接第 3647 页)

参考文献:

[1] PASSINO K M. Biomimicry of bacterial foraging for distributed optimization and control[J]. IEEE Control Systems, 2002,22(3):52-67.

[2] LIU Y, PASSINO K M. Biomimicry of social foraging bacteria for distributed optimization: Models, principles, and emergent behaviors [J]. Journal of Optimization Theory and Applications,2002,115(3): 603-628.

[3] DASGUPTA S, DAS S, ABRAHAM A, *et al*. Adaptive computational chemotaxis in bacterial foraging optimization: an analysis[J]. IEEE Trans on Evolutionary Computation,2009,13(4):919-941.

[4] SINGH S, GHOSE T, GOSWAMI S K. Optimal feeder routing based on the bacterial foraging technique[J]. IEEE Trans on Power Delivery,2012,27(1):70-78.

[5] MISHRA S, BHENDE C N. Bacterial foraging technique-based optimized active power filter for load compensation[J]. IEEE Trans on Power Delivery,2007,22(1):457-465.

[6] RATNAWEERA A, HALGAMUGE S K, WATSON H C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients[J]. IEEE Trans on Evolutionary Computation, 2004,8(3):240-255.

[7] BÄCK T. Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms[C]//Proc of Genetic Algorithms. London: Oxford University Press, 1996.

[8] BISWAS A, DASGUPTA S, DAS S, *et al*. Synergy of PSO and bacterial foraging optimization: a comparative study on numerical benchmarks[C]//Advances in Soft Computing, vol 44. Berlin: Springer-Verlag,2007: 255-263.

[9] MAI Xiong-fa, LI Ling. Opposition-based bacterial foraging optimization algorithm[C]//Proc of ICCCI. 2010: 145-148.

[10] LI M S, TANG W J, TANG W H, *et al*. Bacterial foraging algorithm with varying population for optimal power flow[C]//Lecture Notes in Computer Science, vol 4448. 2007:32-41.