基于微粒群策略的自适应觅食算法研究

赵春丽,刘 清

(南京师范大学计算机科学与技术学院,江苏 南京 210023)

[摘要] 为了克服传统觅食算法 BFA(Bacterial Foraging Algorithm)收敛速度慢以及高维优化收敛性差的问题,提出了一种新的基于微粒群优化策略的自适应觅食算法 ABF-PSO(Adaptive Algorithm Bacterial Foraging Oriented by PSO). 该算法采用自适应趋化步长来提高搜索能力,并根据微粒群优化 PSO(Particle Swarm Optimization)策略来控制细菌的运动方向,避免了细菌运动方向因随机性选取而延误全局最优值搜索的问题. 在详细阐述了动态调整细菌的趋化步长和利用微粒群优化策略更新细菌运动方向后,对经典测试函数分别采用 PSO 算法,BFA 算法和 ABF-PSO 算法进行了对比测试. 实验结果表明,ABF-PSO 算法不仅收敛速度得到很大提高,同时对于复杂和高维搜索的问题获得了很好的收敛性.

[关键词] BFA 算法,收敛速度,自适应趋化步长,PSO 算法

[中图分类号]TP301 [文献标志码]A [文章编号]1672-1292(2013)01-0050-05

Self-Adaptive Bacterial Foraging Algorithm Based on Particle Swarm Optimization Strategy

Zhao Chunli, Liu Qing

(School of Computer Science and Technology, Nanjing Normal University, Nanjing 210023, China)

Abstract: To overcome the problems of low convergence rate, and poor convergence characteristics for larger constrained problems in conventional bacterial foraging algorithm(BFA), this paper proposes a new self-adaptive algorithm bacterial foraging based on PSO(ABF-PSO). The new algorithm improves the search ability with self-adaptive chemotactic steps, and controls the bacterial movement directions according to particle swarm optimization strategy, thus avoiding a delay in reaching the global solution because of random selection of the bacterial movement directions. After the detailed illustrations of dynamic adjustment bacterial chemotactic step, and updating bacterial movement directions by the velocity formula of PSO, this paper tests some classical functions with PSO algorithm, BFA algorithm, and ABF-PSO algorithm. The results show that ABF-PSO algorithm not only has greater improvement in convergence rate, but also gets a fruitful achievement in searching complex and high-dimensioned problems.

Key words: BFA algorithm, convergence rate, self-adaptive chemotactic step, PSO algorithm

觅食算法 BFA(Bacterial Foraging Algorithm)是由 Kevin M Passino 于2002 年提出来的一种仿生随机优化算法[1-3],属于求解全局最优化的群体智能算法. 与其他全局优化算法(如微粒群算法)相比较,它能够较好的找到全局最优解并且有较高的精度值,但缺点就是收敛速度慢和早熟收敛问题. 目前解决这一问题的主要方法:其一是对 BFA 算法的趋化步长做出了改进[2];其二是将 BFA 算法与其他优化算法(如遗传算法,微粒群算法等)相结合[3-6]. 虽然改进后的性能在收敛速度和早熟收敛问题方面都有所提高,可是在解决比较复杂和高维的优化问题时,依然存在早熟收敛问题[8,9]. 本文提出一种基于微粒群策略的自适应觅食算法 ABF-PSO(Adaptive Algorithm Bacterial Foraging Oriented by PSO),通过动态调整细菌的趋化步长和利用微粒群全局优化策略,实验结果表明:与标准觅食算法和微粒群算法相比,不仅提高了收敛速度,而且避免了早熟问题.

收稿日期:2013-01-14.

基金项目:国家自然科学基金(61103185)、江苏省高校自然科学基金项目(10KJD520004).

通讯联系人:刘清,博士,教授,研究方向:面向移动物联网环境的搜索关键技术研究、蚁群算法研究. E-mail:njnulq@163.com

1 标准觅食算法及其早熟和收敛速度慢问题

细菌觅食算法 BFA(Bacterial Foraging Algorithm)是基于人类肠道中大肠杆菌在觅食行为过程中而体现出来的智能行为,在搜索过程中通过营养分布函数来判断搜索算法的优劣[1]. BFA 模拟细菌的觅食行为,包括趋化、复制、驱散 3 个步骤. 下面对这 3 种行为的特征和对应抽象出的操作进行介绍.

(1)趋化性操作:该步骤模仿了大肠杆菌的两种运动模式:翻转(tumble)和前进(swim). 细菌向任意方向移动单位步长定义为翻转. 当细菌完成一次翻转后,若适应值得到改善,将沿同一方向继续移动若干步,直至适应值不再改善,或达到预定的移动步数临界值. 此过程定义为前进.

假设 $\theta^i(j,k,l)$ 表示第 i 个细菌在第 j 次趋化,第 k 次复制,第 l 次驱散后的位置. $C(i)$ 表示细菌翻转或者向前移动的步长,则:

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

其中,$\boldsymbol{\Delta}$ 表示任意方向的向量,其元素取值范围是$[-1,1]$.

(2)复制操作:一旦临界趋化次数,细菌将进行繁殖. 细菌的繁殖过程遵循自然界“优胜劣汰,适者生存”原则. 以趋化过程中的各个细菌适应值累加和为标准,较差的半数细菌死亡,较好的半数细菌分裂成两个子细菌. 子细菌将继承母细菌生物特性,具有与母细菌相同的位置及步长. 为简化计算,可以规定复制过程中细菌总数保持不变. 设 $J(i,j,k,l)$ 为第 i 个细菌在第 j 次趋化,第 k 次复制,第 l 次驱散后所在位置的代价函数值,则第 i 个细菌的健壮函数可以表示为:

$$J^i_{\mathrm{health}}=\sum_{j=1}^{N_c+1}J(i,j,k,l), \tag{2}$$

以此作为评价细菌优劣的准则,其中 N_c 是趋化的次数.

(3)驱散操作:趋化过程可确保细菌的局部搜索能力,复制过程能加快细菌的搜索速度,但对于复杂的优化问题,趋化和复制无法避免细菌陷入局部极小现象发生. BFA 引入驱散过程以加强算法全局寻优能力. 细菌在完成一定次数的复制后,将以一定概率 P 被驱散到搜索空间中任意位置.

BFA 算法在解决简单低维优化问题上已经得到了很好的应用[7],但对于复杂和高维搜索的问题存在早熟和收敛速度慢的问题[8,9]. 而 BFA 算法相对 PSO 来说更加复杂,且各个参数的设置对 BFA 的算法性能有较大地影响,特别是趋化步长 $C(i)$ 的选取和细菌方向 Δ 的选取;对于趋化步长如果 $C(i)$ 太大,细菌若已经到了最优值的附近,这样可能会造成细菌跳过最优值;如果 $C(i)$ 太小,算法的收敛速度就会降低,同时可能使细菌陷入局部最优. 对于细菌方向的选取,标准觅食算法中细菌趋化运动的方向是随机产生的,这样会降低算法的收敛速度. 为此,本文提出了一种改进的细菌觅食算法,通过微粒群全局优化策略,使细菌趋化方向一直是沿着适应度函数值小的方向前进;并通过自适应步长来动态控制细菌的步长.

2 改进的觅食算法(ABF-PSO)

2.1 自适应步长

对于全局收敛的觅食算法,细菌步长的选取是影响其收敛的关键因素. 标准觅食算法采用的固定步长,可能在某些简单的优化问题中容易搜索到最优解,但对于复杂的问题,其搜索能力就会大大下降. 从生物学角度来说,细菌会随着觅食时间的增长,其活跃性会降低,这样运动速度就会越来越慢. 所以为了使细菌能够智能的根据自身情况去改变自己的步长,本文选取了非线性动态递减的函数作为细菌的步长. 趋化步长的自适应调整函数如下:

$$C(i,j+1)=\left(\frac{C(i,j)-C(N_c)}{N_c+C(N_c)}\right)(N_c-j), \tag{3}$$

其中,j 代表第 j 次趋化操作,N_c 是细菌的最大趋化次数,并且 $C(N_c)$ 是预先定义好的参数. 在式(3)中,细菌每进行一次趋化,其运动步长呈曲线下降,即细菌的活跃性慢慢降低直到不再前进. 当复制操作不断增大时,$C(i,j)$ 会不断的减小;所以 j 越小,$C(i,j)$ 就会越大,这样细菌就不会在局部范围内耗费过多的搜索时间;同样 j 越大,$C(i,j)$ 就会越小,这样就能保证细菌在全局最优解附近的时候搜索能力增强,从而找到

最优值.

标准觅食算法的程序终止条件是趋化步骤,复制步骤和驱散步骤的长度都达到最大值,这样在某些情况下会增大该算法的计算量. 故本文中采取设置一个终止条件 ε 以跳出所有循环终止程序.

2.2 基于微粒群优化策略的觅食算法

标准觅食算法中细菌趋化操作时的运动方向是随机产生的,若是以微粒群全局优化策略对其改进,细菌每次趋化时所选取的方向都是向历史最优值靠近,避免了细菌最优值搜索的盲目性,从而缩短了细菌搜索的时间,提高了算法的收敛速度和搜索精度. 本文中,利用 PSO 的全局优化策略作为 BFA 的运动方向,如下:

$$\mathrm{Delta}(i)=w*\mathrm{Delta}(i)+C_1*R_1(\mathrm{Plbest}(i,j,k,ell)-P(i,j,k,ell))+C_2*R_2(\mathrm{Pgbest}-P(i,j,k,ell)),\quad(4)$$

其中,w 是惯性权重,用于维护全局最优与局部最优搜索能力的平衡;$\mathrm{Delta}(i)$ 是第 i 个细菌的方向;$\mathrm{Plbest}(i,j,k,ell)$ 是第 i 个细菌在第 j 次趋化,第 k 次复制,第 ell 次驱散后的个体最优位置;$P(i,j,k,ell)$ 是第 i 个细菌在第 j 次趋化,第 k 次复制,第 ell 次驱散后的细菌位置;Pgbest 是细菌找到的全局最优位置;C_1 和 C_2 是加速因子,均为正实数;R_1 和 R_2 是取在[0,1]范围内的随机数.

这样,细菌可以根据群体内已经寻得的最优值向历史最优点靠近,从而加快了收敛速度.

3 ABF-PSO 算法的性能分析

在本文中,选取经典的测试函数来对 ABF-PSO 算法的性能进行测试. 对于每个测试函数分别采用标准 BFA,PSO 算法和本文提出的 ABF-PSO 算法进行测试,然后通过对比来说明 ABF-PSO 算法的优越性.

3.1 检测函数

实验函数选取了两个单峰值函数和两个多峰值函数,包括 Spere 函数,Axis parallel hyper-ellipsoid 函数,Griewank 函数和 Ackley 函数. 它们分别是:

(1)单峰值 Spere 函数 f_1:

$$f_1(x)=\sum_{i=1}^{p}x_i^2,$$

其中,$p=30$,$x_i\in[-100,100]^p$,该函数的全局最优值为 0,其函数图像如图 1(a)所示.

(2)单峰值 Axis parallel hyper-ellipsoid 函数 f_2:

$$f_2(x)=\sum_{i=1}^{p}i*x_i^2,$$

其中,$p=30$,$x_i\in[-100,100]^p$,该函数的全局最优值为 0,其函数图像如图 1(b)所示.

(3)多峰值 Griewank 函数 f_3:

$$f_3(x)=\sum_{i=1}^{n}\frac{x_i^2}{4\,000}-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1,$$

其中,$p=30$,$x_i\in[-600,600]^p$,该函数的全局最优值为 0,其函数图像如图 1(c)所示.

(4)多峰值 Ackley 函数 f_4

$$f_4(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right)+20+\ell,$$

其中,$p=30$,$x_i\in[-32,32]^p$,该函数的全局最优值为 0,其函数如图 1(d)所示.

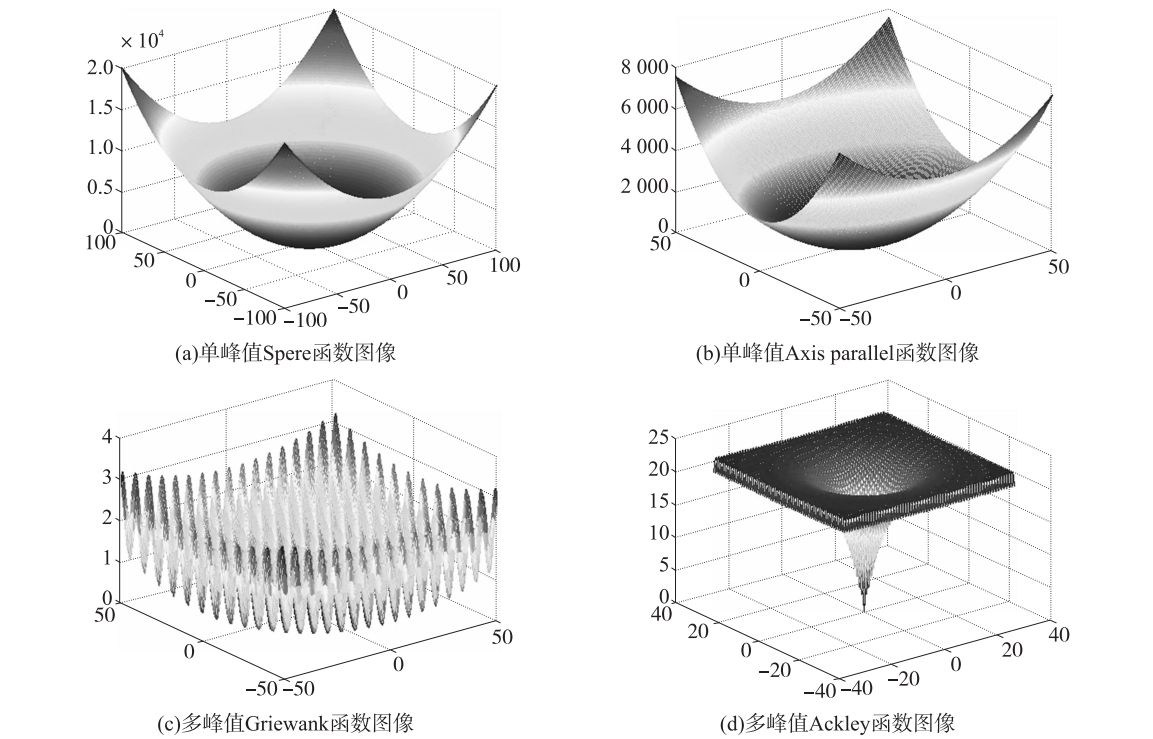
3.2 实验环境设置

算法仿真平台为 Windows XP Professional,处理器为 i3-2120,主频为 3.30 GHz,内存为 1.84 G,软件为 matlab7.1. PSO 参数的设置为 $C_1=1.494$,$C_2=1.494$,$w=0.729$,其余的参数设置,如表 1 所示.

表 1　参数设置

Table 1　Preferences

ABF-PSO								BFA					
S	N_c	N_{re}	N_{ed}	N_s	P_{ed}	C_1,C_2	w	S	N_c	N_{re}	N_{ed}	N_s	P_{ed}
50	50	16	4	12	0.25	1.5,2.2	0.9	100	200	16	4	12	0.25



(a)单峰值Spere函数图像　(b)单峰值Axis parallel函数图像

(c)多峰值Griewank函数图像　(d)多峰值Ackley函数图像

图 1　四个函数图像

Fig. 1　Four function images

3.3　仿真结果

对于每个测试函数,都是进行 50 次迭代运算,将每次运算所寻找到的最好的适应度值保存下来,然后对它们取平均值和标准方差,结果如表 2 所示. 在优化这 4 个函数时,本文对 f_1、f_2、f_3 和 f_4 函数设置的终止条件 ε 为 0.001;ABF-PSO 算法总能够寻找到全局最优值,与其他几种算法相比,体现了 ABF-PSO 算法的优越性,也体现了 ABF-PSO 算法的可行性和有效性.

而对于 f_1 来说 BFA 寻优能力要强于 PSO 算法,而对于其他的函数 BFA 寻优能力没有 PSO 好,这与 BFA 的步长有关,当细菌的步长一直不变,在找到最优值附近的时候,有可能会跳过最优值,这样就会造成反复探索或找不到最优值. 本文采用的标准方差计算公式如下式

$$\mathrm{std}=\sqrt{\left(\frac{1}{n}\sum_{i=1}^{n}\left(\mathrm{fgbest}_i-\mathrm{meanf}\right)\right)}\ , \tag{5}$$

其中,n 是运行算法的次数;fgbest_i 是第 i 次运行算法时的最好适应度值;meanf 是运行 50 次算法所得到的所有最好适应度函数值的均值.

表 2　测试结果

Table 2　The results

f	ABF-PSO		BFA		PSO	
	均值	标准方差	均值	标准方差	均值	标准方差
f_1	$2.100\,7\times10^{-5}$	$6.363\,4\times10^{-5}$	0.528 8	0.064 9	11.816 2	9.642 7
f_2	$7.006\,0\times10^{-5}$	$2.755\,8\times10^{-4}$	7.779 3	1.237 2	3.092 1	4.881 7
f_3	0.018 1	0.0911	15.732 1	7.542 4	1.114 5	0.096 1
f_4	$9.750\,4\times10^{-5}$	$3.486\,1\times10^{-4}$	18.225 7	0.606 6	3.965 3	0.614 8

3.4　收敛特性

在图 2 中给出了 BFA、PSO 和 ABF-PSO 优化这 4 个函数的收敛曲线图. 本文选取计算目标函数的次数来衡量它们的收敛速度. 本文设置最大计算次数为 2×10^5,以防止计算量过大耗费过多时间;另外由于 BFA 和 ABF-PSO 调用目标函数具有不确定性,为了方便本文采取细菌的一次趋化为调用一次目标函数. 图 2 中的横坐标是目标函数的调用次数,纵坐标是最好的适应度值取以 10 为底的 log 值. 从该图中,我们可以看出 BFA-PSO 算法结合了 BFA 和 PSO 的优点,能够较快的找到最优值避免了早熟.

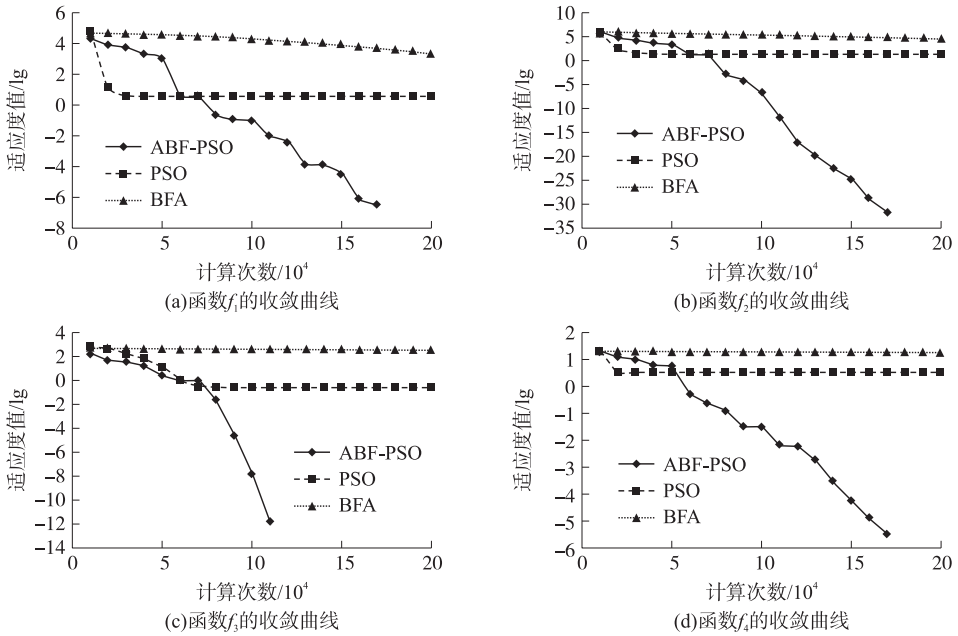


图 2　收敛曲线

Fig. 2　Converging curved lines

4　结语

本文首先介绍了标准觅食算法,指出它所存在的收敛速度慢和早熟收敛问题. 然后在此问题上对其进行了两点改进,其一是产生自适应的步长;其二是利用微粒群全局优化策略改变其运动方向. 最后,通过对经典测试函数的测试,并与 BFA 和 PSO 算法进行了比较,验证了基于微粒群策略的自适应觅食算法(ABF-PSO)的可行性和有效性.

[参考文献](References)

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

[2] Ajith Abraham, Aboul-Ella Hassanien, Patrick Siarry, et al. Foundations of Computational Intelligence Volume 3[M]. Berlin: Springer Berlin Heidelberg,2009:23-55.

[3] 杨尚君,王社伟,陶军,等. 基于混合细菌觅食算法的多目标优化方法[J]. 计算机仿真,2012,29(6):218-222.
Yang Shangjun, Wang Shewei, Tao Jun, et al. Multi-Objective optimization method based on hybrid bacterial foraging algorithm [J]. Computer Simulation,2012,29(6):218-222. (in Chinese)

[4] 储颖,糜华,纪震,等. 基于粒子群优化的快速细菌群游算法[J]. 数据采集与处理,2010,25(4):442-448.
Chu Ying, Mi Hua, Ji Zhen, et al. Fast bacterial swarming algorithm based on particle swarm optimization[J]. Data Acquisition and Processsing,2010,25(4):442-448. (in Chinese)

[5] Prof Emillio Corchado, Prof Juan M Corchado, Prof Ajith Abraham. Innovations in Hybrid Intelligent Systems[M]. Berlin: Springer Berlin Heidelberg,2007:255-263.

[6] Dong Hwa Kim, Ajith Abraham, Jae Hoon Cho. A hybrid genetic algorithm and bacterial foraging approach for global optimization[J]. Information Sciences,2007,177(18):3918-3937.

[7] 扬大炼,李学军,蒋玲莉,等. 一种细菌觅食算法的改进及其应用[J]. 计算机工程与应用,2012,48(13):31-34.
Yang Dalian, Li Xuejun, Jiang Lingli, et al. Improved algorithm of bacterial foraging and its application[J]. Computer Engineering and Applications,2012,48(13):31-34. (in Chinese)

[8] Farhat I A, EI-Hawary M E. Dynamic adaptive bacterial foraging algorithm for optimum economic dispatch with valve-point effects and wind power[J]IET Generation, Transmission and Distribution,2010,4(9):989-999.

[9] Korani W M, dorrah H T, Emara H M. Bacterial foraging oriented by particle swarm optimization strategy for PID tuning[C]// IEEE International Symposium on Computational Intelligence in Robotics and Automation. Korea: Daejeon,2009:445-450.

[责任编辑:刘　健]