

改进的菌群算法研究

何 丰, 周 鹏

(重庆邮电大学 通信与信息工程学院, 重庆 400065)

摘 要: 为了提高菌群寻优算法(Bacterial Foraging Optimization, BFO)的搜索能力和解决多峰值复杂适应度函数模型避免过早收敛的问题,文中对原始菌群算法进行改进,提出多峰值菌群算法。将寻优过程分成两个时期,前期和原始菌群算法相同,在菌群收敛的后期,加入峰值数目和区间的判断,将区间编号,保证区间内部单峰值;然后在区间内部迭代运行菌群搜索,独立寻优,在多峰值和较复杂模型的情况下进行研究和评估。实验表明,在收敛速度、收敛稳定性和寻找全局最优方面均优于原始菌群算法。

关键词: 菌群算法;多峰值;多区间;寻优;多峰值菌群算法

中图分类号: TP301.6

文献标识码: A

文章编号: 1673-629X(2013)11-0054-05

doi: 10.3969/j.issn.1673-629X.2013.11.014

Research on Optimized Bacterial Foraging Algorithm

HE Feng, ZHOU Peng

(College of Communication & Information Engineering, Chongqing University of Posts and Telecommunications,
Chongqing 400065, China)

Abstract: To improve the search capabilities of bacterial foraging optimization and solve the problem that the multiple peak complex adaptive function avoids premature convergence, a new type of multiple peak bacterial foraging optimization algorithm (MBFO) is proposed in this paper by improving bacterial foraging optimization. Divide the process into two parts, in the beginning of the algorithm, the operating is set as the same as the bacterial foraging optimization. In the second period, add a judgement of the number and the location of peak. Make sure only one peak in each area. And in the area do the iteration of the bacterial foraging optimization, looking for the optimal independently, studying and estimating at the case of multiple peak and complex model. The experiment results indicate that the new type of multiple peak bacterial foraging optimization shows better results than bacterial foraging optimization in convergence rate, stability and looking for the global optimal.

Key words: bacterial foraging optimization; multiple peak; multiple interval; optimal search; multiple peak bacterial foraging optimization

0 引 言

菌群觅食算法是2002年由Passino和muller等提出的,原始的细菌觅食算法^[1]是根据大肠杆菌的觅食行为而提出的一种仿生学算法^[2]。这种算法在解决实数优化问题有明显的优势。与之相近的还有鱼群算法、粒子群算法、蚁群算法等智能群体算法。这些算法都引起了学者的广泛关注,以及较为广泛的应用。

Abraham, Bisawas, Kim等经过研究注明了BFO算法的繁殖操作有利于算法收敛。改进了其收敛方式。并应用于找寻全局最优。实现了对PID控制参数的优化。此外,BFO算法还有较多应用:设计模糊PID控制器参数,矩形微带天线响应频率计算,金融期权参

数估计,股市预测,自适应信道均衡器设计,PI控制器参数优化设计,电力系统无功电力调度等。

在国内,也开始有大量的关注:李威武^[3]等人根据muller等人提出的单个细菌趋化模型,借鉴PSO算法提出了一种细菌群体趋化性算法。刘文霞^[4]等人利用微分进化和混沌迁移的方法对BC算法进行改进,提高了全局搜索能力。陈翰宁等改进了个体交流方式,提高了BFO算法的寻优能力。储颖^[5]等结合了PSO算法,实现了个体对种群最优位置的感知。

对算法的研究主要是对细菌行为的改进。文中将分析菌群个体的行为,在比较BFO和BCHO的基础上,对菌群个体的行为进行改进,来克服多峰值等极端

收稿日期: 2013-01-22

修回日期: 2013-04-27

网络出版时间: 2013-08-28

基金项目: 重庆市科技计划项目(CDY2011120001); 工信部科研计划项目([2011]353)

作者简介: 何 丰(1962-),男,重庆人,教授,硕士生导师,研究方向为电子新技术。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20130828.0829.014.html>

环境,提出对菌群算法的改进。

1 菌群算法简介

细菌是一种单细胞生物,自身带有鞭毛,用于其个体移动。细菌个体移动分为两种有效情况,向前游动以及向后翻转。细菌个体本身有趋利避害的特性,使得细菌个体本身向着富营养值的位置移动,在宏观的菌群视角上就是菌群有趋化的特性。同时,在菌群移动若干位置之后,菌群还有优胜劣汰的自然规律。即在较优位置的细菌个体营养充足就会等到繁殖的机会,而位置较差的细菌个体,由于营养不足将会死亡被菌群淘汰,这样细菌种群就会不断地聚集到位置最优的位置上。而且为了避免细菌个体寻优时遇到局部最优和过早收敛的情况,还引入了驱散机制来解决这些问题。

1.1 菌群算法实现

菌群优化算法^[6]实现大体分为三个步骤:趋化,繁殖和驱散。

1) 趋化:细菌向富营养区聚集的行为称为趋化,自然界中细菌个体对不同的营养值区域有感知,如果它移动向一个位置,新的位置营养值高就会继续往这个方向前进,否则就会向相反的方向运动,算法实现的时候,引入适应度函数模型,得到适应度函数值,比较不同位置的函数值来模拟细菌对环境的感知。

设 S 是一个细菌种群中细菌的数量,初始化细菌位置:

$$P = \min + \text{rand} * (\max - \min) \quad (1)$$

其中, \min 和 \max 是细菌所在空间的上下界; rand 是 $[0, 1]$ 区间的随机数。

用 $P(i, j, k, l)$ 来表示细菌 i 在第 j 次趋向性操作、第 k 次繁殖操作和第 l 次迁徙操作后的位置。每一次趋化操作后细菌的位置更新公式为:

$$P(i, j+1, k, l) = P(i, j, k, l) + C(i) \frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}} \quad (2)$$

其中, $C(i)$ 表示按选定的方向前进的步长; $\Delta(i)$ 为任意方向的向量; $\frac{\Delta(i)}{\sqrt{\Delta^T(i)\Delta(i)}}$ 表示单位步长向量。

在菌群的移动过程中,还要考虑菌群的个体对群体的其他个体的影响^[7-8],综合考虑到细菌种群内部的感应机制,所以适应度采用 J_{cc} 表示:

$$J_{cc}(i, j, k, l) = \sum_{i=1}^s [-d_{\text{attract}} \exp(-w_{\text{attract}} \sum_{n=1}^p (P(i, j, k, l) - P(1:s, j, k, l))^2)] + \sum_{i=1}^s [h_{\text{repellant}} \exp(-$$

$$w_{\text{repellant}} \sum_{n=1}^p (P(i, j, k, l) - P(1:s, j, k, l))^2)] \quad (3)$$

其中, d_{attract} 表示吸引剂的数量; w_{attract} 表示吸引剂释放速度; $h_{\text{repellant}}$ 表示排斥剂数量; $w_{\text{repellant}}$ 表示排斥剂释放速度。

那么个体在一个位置的适应值应当是营养值加上群体中其他个体对它的影响^[9]的和也就是:

$$\text{fitness}(i, j, k, l) = \text{cost}(P(i, j, k, l)) + J_{cc} \quad (4)$$

2) 繁殖:当细菌个体移动一定步数之后,细菌群体就会对所处的环境有所感知,自然界中的细菌(假设营养是有限的)有一个优胜劣汰的规律,处于富营养的细菌就会有繁殖的机会,由于细菌是二分裂,也称为繁殖,由于在算法实现的时候,选择适应度函数值高的个体,同时繁殖保留父代所有信息,所以一般称之为繁殖。而处于较差位置的细菌将会直接淘汰^[10]。适应度值累加和可表示为:

$$J_{\text{health}}(i) = \sum_{j=1}^{N_c} J(i, j, k, l) \quad (5)$$

其中 $J_{\text{health}}(i)$ 表示第 i 个细菌在一个生命周期内的健康度,也即 N_c 次趋化的适应度值累加和; N_c 表示细菌趋化的最大次数。

3) 驱散:经过以上两步,菌群个体会被聚集在一个较高的函数峰值附近,可是这个峰值可能是一个伪最优值,这样菌群陷入过早收敛。为了避免这种情况,在算法中,每繁殖一定的代数,所有细菌将会按照一定的概率进行判定,一定数目的细菌个体将会被突然杀死。同时也会有相同数目的个体的细菌突然产生在环境中,并从随机的位置,重新开始寻优^[11]。

1.2 菌群优化算法基本实现流程

Step1:设置相关参数。 S 设为细菌规模数(为了繁殖的需要,取偶数), N_c 设为趋化次数、 N_s 设为前进次数、 N_{re} 设为繁殖次数、 N_e 设为迁移次数; P_{ed} 设为驱散次数, p 设为维度, $C(i)$ ($i = 1, 2, \dots, S$) 设为步长。

Step2:驱散循环: $l = l + 1$ 。

Step3:繁殖循环: $k = k + 1$ 。

Step4:趋化循环: $j = j + 1$ 。

Substep4.1:趋化操作。对 $i = 1, 2, \dots, S$, 随机找个方向移动一步。

Substep4.2:计算适应度值 $\text{fitness}(i, j, k, l)$ 。

Substep4.3:令 $\text{Bestfit} = \text{fitness}(i, j, k, l)$, 保存当前最优值,直至找到更优的值替代。

Substep4.4:若发现翻转后适应度值比翻转以前更好,即 $\text{fitness}(i, j, k, l) < \text{fitness}(i, j+1, k, l)$, 则继续沿着此方向前进特定步长。进行游动循环: $m = m + 1$, 如果 $\text{fitness}(i, j+1, k, l) > \text{Bestfit}$ 则令 $\text{Bestfit} = \text{fitness}(i, j+1, k, l)$, 继续游动;如果 $\text{fitness}(i, j+1, k,$

1) < Bestfit, 则终止游动。

Substep4.5: $i = i + 1$, 进行下一个细菌的趋化运动。

Step5: 如果 $j < N_c$, 继续执行 Step4。

Step6: 繁殖实现: 细菌个体通过淘汰位置较差的个体实现菌群的种群优化, 加速寻优过程: 在趋化完成后, 比较趋化过程中各细菌适应值累加和, 较差的一半个体被淘汰。较优的一半个体继承父代所有信息, 并重新编号, 进行下一轮的趋化行为。

Step7: 如果 $k < N_{re}$ 继续执行 Step3。

Step8: 驱散实现。趋化繁殖若干代之后, 让每一个个体都进行一次驱散判定, 如果判定值小于给定概率 P_{ed} 则该个体被驱散, 重新进行初始化如(1)式所示。驱散结束之后由于个体的随机分布, 继续进行趋化和繁殖。

Step9: 若 $l < N_{ed}$, 继续执行 Step2。否则算法结束。

2 多峰值菌群算法

菌群优化算法在数值寻优方面有较好的性能, 在适应值模型较为简单, 而且有较为明显、平滑的单峰值情况下, 性能得到最优^[12-13]。但是在实际应用的时候, 发现在多峰值、多级值以及多皱褶等复杂环境下, 菌群优化算法极易陷入局部最优, 或者在寻优效率上表现不是很理想。

为了解决在复杂环境下的寻优不理想的问题, 文中做出一些改进如下: 将菌群优化算法分为两个阶段, 前期和后期。在前期算法中利用菌群本身较好的趋化性能使其收敛于明显的峰值附近, 算法实现和菌群优化算法相同。在后期, 算法将判断菌群是否收敛, 也即是否开始后期算法, 然后判断模型中发现的峰值的主峰的个数, 之后收集信息对主峰的位置进行确定, 在不同的区域中分别在这样的单峰值区域迭代执行菌群优化算法, 使得其寻优速度得到优化。

2.1 多峰值算法实现

多峰值算法在菌群优化算法的基础上将其进行改进, 将其分为两个时期。首先在前期, 还是采用菌群优化算法的策略, 可以使得菌群较快地收敛于若干区域。然后加上判断条件, 如果菌群已经聚集在若干区域, 则开始进行第二个阶段的寻优操作。在算法中如果有超过五分之一种群总数的个体聚集在小于寻优范围长度十分之一的区域, 判断这个个体已经找到寻优位置, 同时如果这样的个体总数达到种群总数的 80% 以上, 整个种群开始进入第二个阶段的寻优。在后期寻优过程中, 采集个体坐标位置, 然后判断其周边个体的数目, 当数目达到总数的五分之一以上则确认其为一个单峰

值区域, 记录第一个个体位置, 确定新的搜索区域范围。直到满足收敛数目。确定环境中的峰值数目, 为每一个峰值确定搜索区间, 再在每一个区间独立进行菌群优化算法。

2.2 多峰值菌群优化算法基本实现流程

Step 1: 初始化参数。这里设置和菌群优化算法相同的参数, 以做比较。初始化细菌位置。

Step 2: 趋化循环: $k = k + 1$ 。

Step 3: 繁殖循环: $j = j + 1$ 。

Step 4: 趋化: 和菌群优化算法相同, 不做重复介绍。

Step 5: 收敛判断: 将标号为 1 的个体记录为圆心 1, 以一定的半径做区域划分 (取搜索区域边长的 1/20), 给每个个体增加一个标志位, 对所有个体进行判断, 如果个体处于这个区域那么将这个标志位设置为 1, 增加一个计数器 t_1++ ; 如果 t_1 大于阈值, 设置为 5 那么这个圆心将会被保留, 将下一个标志位为 0 的个体设置为圆心, 重复操作, 记录新的区域, 设置标志位 2, 增加计数器 t_2 ; 如果 t_1 小于阈值, 将舍弃这个圆心, 将标号变为当前标号加一, 计数器清零。当判断到编号为 S 的个体的时候, 进行一次收敛结束判断: 如果 $t_1 + t_2 + \dots > 4/5S$ 则算法判断结束, 进行到算法后期。否则, 继续进行趋化循环。

Step 6: 分别在各个区域进行菌群优化算法, 各个菌群的搜索区域为各个区域的圆心和以上的划分区域的半径。所有标号为 0 的个体编为一组, 全局搜索。

Step 7: 将各个分组独立寻优的分组最优值记录, 并将各个记录张贴记录板。比较记录板上各个分组的最优值, 此最优值就是当前的全局最优值。

Step 8: 驱散实现。让每一个个体都进行一次驱散判定, 如果判定值小于给定概率 P_{ed} 则该个体被驱散, 重新进行初始化如(1)式所示。驱散结束之后由于个体的随机分布, 继续进行趋化和繁殖, 重新开始前期寻优。

3 性能测试与结果分析

多峰值菌群算法和原始的菌群算法相比, 在不同的测试函数下是不同的, 文中将比较在简单平滑环境下以及在复杂多峰值环境下两种算法的寻优效果。所以选择了比较典型的测试函数: (1) Sphere 函数, 这个函数不仅是个单峰函数, 而且是个搜索环境极好的, 有明显的可趋化性能的函数。(2) 多峰函数。增加两个复杂函数: (3) De Jong 函数, 病态函数。(4) Coldstein-Price 函数, 容易陷入局部极小值。

F1: Sphere 函数

$$f(x, y) = x^2 + y^2 \quad (6)$$

其中, $-10 \leq x, y \leq 10$, 此函数全局极小值点为 $(0, 0)$, 极小值为 0。

F2: Needle-in-haystack 函数

$$f(x, y) = (\frac{3}{0.05 + x^2 + y^2})^2 + (x^2 + y^2)^2 \tag{7}$$

其中, $-5.12 \leq x, y \leq 5.12$, 优化目标为求取函数极大值, 全局极大值为 3 600, 极大值点为 $(0, 0)$, 有四个局部极大值点对称分布于 $(+5.12, +5.12)$, $(-5.12, -5.12)$, $(+5.12, -5.12)$, $(-5.12, +5.12)$, 当优化结果大于 3 599 时认为算法收敛。

算法参数: BFO 中, 维度 $p = 2$, 种群规模 $s = 40$, 趋化代数 $N_c = 50$, 繁殖代数 $N_{re} = 5$, 驱散代数 $N_{ed} = 2$, 驱散概率 $P_{ed} = 0.25$, 初始步长 $c = 0.01$; GA 中, 初始种群规模大小为 40, 染色体长度为 44, 交叉概率为 0.7, 变异概率为 0.1, 最大迭代步数为 $\text{maxgen} = 500$ 。

F3: De Jong 函数

$$f(x, y) = 100 \times (x^2 - y)^2 + (1 - x)^2 \tag{8}$$

这是一个二维函数, 其中 $-2.048 \leq x, y \leq 2.048$, 它在整个解析域中只有一个全局极小点 $f(1, 1) = 0$, 虽然是单峰值函数, 但它是病态的难以进行全局优化。当优化结果小于 0.005 时认为算法收敛。

F4: Goldstein-Price 函数

$$f(x_1, y) = [1 + (x + y + 1)^2(19 - 14x + 3x^2 - 14y + 6xy + 3y^2)] \cdot [30 + (2x - 3y)^2 \cdot (18 - 32x + 12x^2 + 48y - 36xy + 27y^2)] \tag{9}$$

其中 $-2 \leq x, y \leq 2$, 该函数有 4 个极小值点 $(1.2, 0.8)$, $(1.2, 0.2)$, $(-0.6, -0.4)$, $(0, -1)$, 只有一个全局极小值点为 $(0, -1)$, 极小值为 3。当优化结果小于 3.005 时, 认为算法收敛。

算法参数: 在多峰值算法中参数设置与菌群优化算法相同, 维度 $p = 2$, 种群规模 $s = 40$, 趋化代数 $N_c =$

10, 繁殖代数 $N_{re} = 2$, 初始步长 $c = 0.01$, 峰值区域设置为 4, 取消驱散操作。

结果如表 1, 图 1 ~ 4 所示。

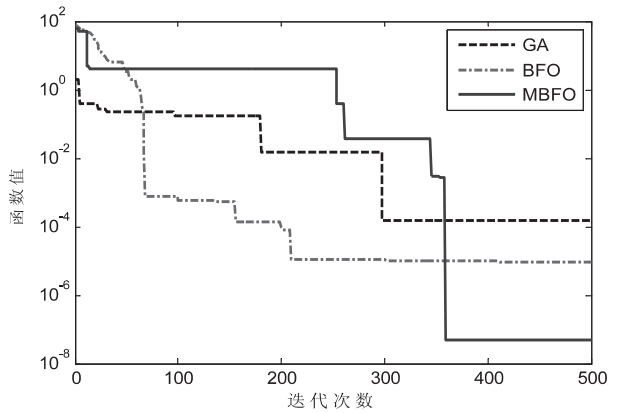


图 1 Sphere 函数三种算法比较图

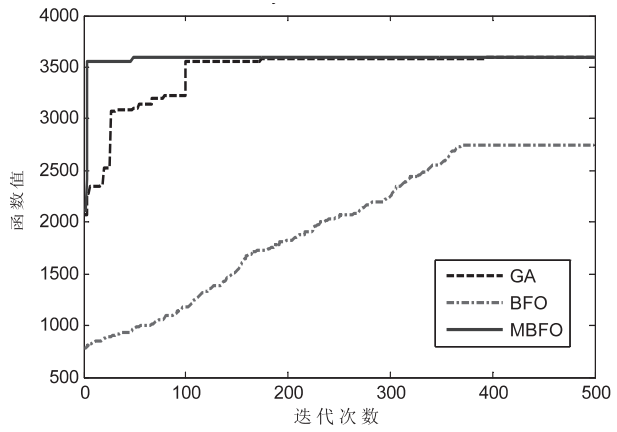


图 2 Needle-in-haystack 函数三种算法比较图

由图 1 至图 4 和表 1 可以看出各种算法性能的优劣: 普通菌群算法对于平滑函数收敛效果不错, 对于复杂函数收敛比较慢, 而且收敛效果不够好, 对于多峰函数, 则完全不能找到全局最优值, 运行 20 次均失效。而改进的菌群算法无论对于简单平滑函数还是多峰及

表 1 三种不同算法的性能比较

函数	算法	最优值	平均值	最少迭代次数	平均迭代次数	成功次数
Sphere	GA	3.998 9e-05	1.657 83e-04	184	321.35	13
	BFO	1.293 4e-07	5.437 5e-05	25	152.35	20
	多峰值	1.245 8e-10	5.715 8e-06	16	229.5	20
Needle-in-haystack	GA	3 600.0	3 592.3	385	398.3	6
	BFO	3 600.0	2 961.6	382	460.5	5
	多峰值	3 600.0	3 599.6	278	301.2	18
DeJong 函数	GA	3.517 3e-03	1.362 7e-02	384	473.35	4
	BFO	5.682 6e-04	1.148 3e-2	253	350.5	9
	多峰值	3.260 2e-08	8.997 6e-5	28	230.9	16
Goldstein-Price 函数	GA	3.010 4	3.036 5	500	500	0
	BFO	3.000 1	3.022 8	208	378.52	9
	多峰值	3.000 0	3.008 4	19	284.81	18

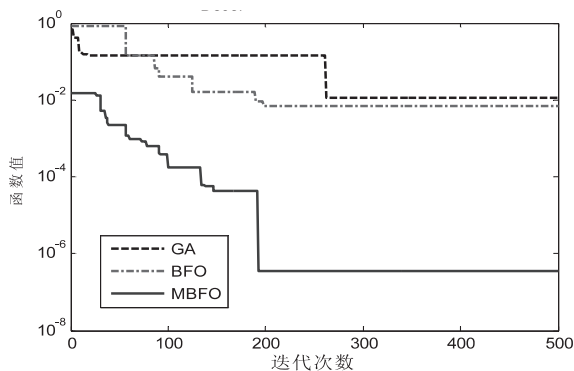


图 3 De Jong 函数三种算法比较图

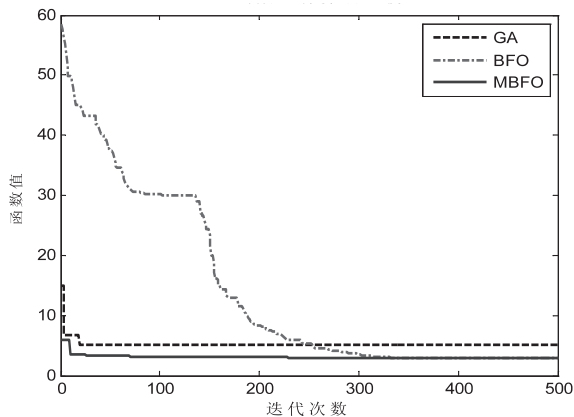


图 4 GP 函数三种算法比较图

复杂函数,比普通遗传算法和普通菌群算法都要好,收敛速度、算法性能方面都有明显的优势,运用于无显著峰谷的非线性优化问题的求解上表现出了优越性。

4 结束语

文中对原始菌群寻优算法进行改进,针对多峰值以及超多极值环境,提出一种新型的多峰值菌群算法,即将算法分为前期和后期两个阶段,前期操作和菌群优化算法相同,使得菌群较快聚集在多个峰值周围,后期阶段将已经分割区域的菌群,在各自所在的小单峰值区域内迭代执行菌群优化算法,独立寻优。通过基准函数的测试实验表明,该算法在收敛性和寻找全局最优解方面有非常好的效果。但是多峰值算法的各阶段参数设置及对收敛性能影响尚未明确,还需要进一步研究和探讨该算法能否应用在 0-1 背包问题上等一系列问题。

(上接第 53 页)

[7] 陈伟海,徐鲤鸿,刘敬猛,等. 网格简化中基于特征矩阵的二次误差测度算法[J]. 北京航空航天大学学报,2009,35(5):572-575.

[8] 杜晓晖,尹宝才,孔德慧. 基于加权二次误差测度的边折叠简化算法[J]. 北京工业大学学报,2007,33(7):731-736.

[9] 陈忆群,曹瑾音,林淑金. 一种均衡代价的网格简化算法[J]. 计算机工程与应用,2011,47(15):75-79.

参考文献:

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

[2] Muller S, Marchetto J, Airaghi S, et al. Optimization based on bacterial chemotaxis[J]. IEEE trans on evolutionary computation,2002,6(1):16-29.

[3] 李威武,王 慧,邹志君,等. 基于细菌群体趋药性的函数优化方法[J]. 电路与系统学报,2005,10(1):58-63.

[4] 刘文霞,刘晓茹,张建华,等. 基于微分进化和混沌迁移的细菌群体趋药性算法[J]. 控制理论与应用,2009,26(4):353-357.

[5] 储 颖,邵子博,糜 华,等. 细菌觅食算法在图像压缩中的应用[J]. 深圳大学学报(理工版),2008,25(2):153-157.

[6] Chen Hanning, Zhu Yunlong, Hu Kunyuan. Cooperative bacterial foraging algorithm for global optimization[C]//2009 中国控制与决策会议论文集. [s. l.]:[s. n.], 2009:3896-3897.

[7] 李 明,杨成梧. 细菌菌落优化算法[J]. 控制理论与应用,2011,28(2):223-228.

[8] 黄伟锋,林卫星,范怀科,等. 细菌觅食优化的智能 PID 控制[J]. 计算机工程与应用,2011,47(21):82-85.

[9] 杨俊安,庄镇泉,史 亮. 多宇宙并行量子遗传算法[J]. 电子学报,2004,32(6):923-928.

[10] Kim H D, Cho H J. Adaptive tuning of PID controller for multi-variable system using bacterial foraging based optimization [C]//Proceedings of 3rd international Atlantic Web intelligence conference on advances. New York: IEEE, 2005:231-235.

[11] Chen H C. Bacterial foraging based optimization design of fuzzy PID controllers[C]//Proceedings of 4th international conference on intelligent computing. New York: IEEE, 2008:841-849.

[12] Golipudi S V R S, Pattnaik S S, Bajpai O P, et al. Bacterial foraging optimization technique to calculate resonant frequency of rectangular microstrip antenna[J]. International journal of RF and microwave computer-aided engineering, 2008,18(4):383-388.

[13] 张代远. 一类新型改进的广义蚁群优化算法[J]. 计算机技术与发展,2012,22(6):39-44.

[10] 李现民,李桂清,张小玲,等. 基于子分规则的边折叠简化方法[J]. 计算机辅助设计与图形学学报,2002,14(1):8-13.

[11] 侯宝明,崔红霞,刘雪娜. 三维网格模型的快速拓扑重建算法[J]. 计算机应用,2010,30(11):3002-3004.

[12] Garland M. Quadric-based polygonal surface simplification [D]. Carnegie: Carnegie Mellon University, 1999.

改进的菌群算法研究

作者：[何丰](#)，[周鹏](#)，[HE Feng](#)，[ZHOU Peng](#)
作者单位：[重庆邮电大学 通信与信息工程学院, 重庆, 400065](#)
刊名：[计算机技术与发展](#)

英文刊名：

ISTIC

[Computer Technology and Development](#)

年，卷(期)：

[2013\(11\)](#)

本文链接：http://d.wanfangdata.com.cn/Periodical_wjfz201311015.aspx