基于 Levy 飞行的改进菌群觅食算法①

严小飞, 叶东毅

(福州大学 数学与计算机学院, 福州 350108)

摘 要: 针对经典菌群觅食算法因固定趋化步长导致的求解精度不高、收敛性能差等缺陷, 提出一种基于 Levy 飞行的菌群觅食算法, 其特点是利用基于 Levy 分布的趋化步长改善算法的求解精度与收敛性能, 借助 Levy 飞行随机游走策略改善细菌迁徙位置. 多个基准测试函数的实验结果表明, 该算法在求解质量和收敛性能上均取得了较好的改进效果.

关键词: 菌群觅食算法; Levy 飞行; 函数优化

Improved Bacterial Foraging Optimization Algorithm Based on Levy Flight

YAN Xiao-Fei, YE Dong-Yi

(College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China)

Abstract: The classical bacterial foraging optimization algorithm suffers from low solution accuracy and poor convergence performance due to the fixed Chemotaxis step-size. To handle these problems, an improved bacterial foraging optimization algorithm based on Levy flight is put forward. The proposed algorithm uses Levy distribution based Chemotaxis step-size for the improvement of both solution accuracy and convergence performance, and random walk strategy of Levy Flight for the improvement of bacterial migration position. Experimental results on several benchmark test functions show that the proposed algorithm achieves noticeable improvement in terms of solution quality and convergence performance in comparison with existing bacterial foraging optimization algorithms.

Key words: bacterial foraging optimization; Levy flight; function optimization

菌群觅食优化算法(Bacterial Foraging Optimization, BFO)[1]是 Passino Kevin 于 2002 年提出的一种新的群智能优化算法, 该算法以模拟人体肠道内大肠杆菌觅食行为基础, 具有结构简单、易于仿真实现、可并行处理等特点,已成功应用于许多现实中的问题, 例如工程控制[2]、信号处理[3], 神经网络训练[4].

在求解优化问题过程中, 经典 BFO 算法通过趋化算子决定算法搜索方向以及局部搜索详细程度, 通过繁殖算子进行优胜劣汰保留优秀个体保证其收敛性, 最后利用迁徙算子避免陷入局部最优. 虽然算法中细菌通过个体和群体间的协同作用能在一定程度上扩大搜索范围, 增强局部搜索能力, 但是算法依然存在收敛速度慢、求解精度不高等问题, 其原因在于经典 BFO 算法的核心参数——趋化步长是以固定距离设计为准则, 不能很好平衡细菌局部搜索和全局收敛性能之间的关系. 针对经典 BFO 算法趋化步长为固定值的缺陷, 人们已经提出了一些改进的算法, 例如文献[3,5]中提出用 Takagi-Sugeno 型模糊推理机制来选取最优步长的模糊细菌觅食算法(Fuzzy Bacterial Foraging Optimization), 然而该算法的性能完全依赖于隶属函数和模糊规则参数的选择, 需要反复试验获得参数, 算法实用性不强. 另一种常见的方法是利用自适应机制来调整趋化步长, 例如, Chen 等人 2008 年在文献[6]中提出了一种自适应步长菌群觅食算法(Self-Adaptation Bacterial Foraging Optimization, SA-BFO)[6], 其中每个细菌的趋化步长根据当前位置

① 基金项目:国家自然科学基金(71231003)
收稿时间:2014-06-17;收到修改稿时间:2014-09-02

的搜索状态和游动步长的大小进行调整, 遗憾的是文中并未探讨算法对于高维或复杂多峰优化问题求解的有效性. 2009 年, Dasgupta 和 Das 等人对趋化步长进行理论分析后提出了一个结合目标函数值的自适应趋化步长菌群觅食算法(Adaptive Bacterial Chemotaxis Foraging Optimization, ABFO)[7][8], 取得了较好的改进效果. 另一方面, 正是由于该算法趋化步长的取值与目标函数值相关, 对于初始函数值较小的目标函数, 初始阶段趋化步长的取值往往过小, 算法容易陷入局部最优. 因此, 有必要进一步研究更有效的趋化步长自适应调整策略, 提高 BFO 算法的求解质量.

本文受 Levy 飞行觅食策略的启发, 提出一种基于 Levy 飞行的菌群觅食算法, 利用 Levy 步长代替固定趋化步长, 借助 Levy 飞行频繁短距离探索和少数长距离探索的搜索特性, 以平衡 BFO 算法中趋化步长的取值, 达到改善算法的求解精度和收敛速度的目的. 此外, 算法借助 Levy 飞行随机游走策略来产生细菌迁徙后的位置, 以利于算法跳出局部最优. 为了评估本文算法的性能, 本文采用 10 个常用的基准测试函数进行数值实验, 并同经典 BFO 算法、SA-BFO 算法和 ABFO 算法进行比较, 实验结果以及 t-test 统计检验表明, 本文算法具有更好的求解精度和收敛性能, 取得了良好的改进效果.

1 经典菌群觅食算法

生物学研究表明, 大肠杆菌觅的食过程主要包含四个基本步骤: 趋化、聚集、繁殖和迁徙, 它们每一步移动都是根据其自身生理和周围环境约束情况下, 在单位时间内所获得的能量达到最大[1], 经典 BFO 算法正是基于这样的觅食过程而提出的仿生随机优化算法.

1.1 趋化过程

趋化过程模拟了细菌在觅食过程中翻转和游动. 细菌通过翻转找到一个新的游动方向, 利用游动进行相同方向的多次运动. 假设 $C(i)$ 为趋化步长, $q^i(j,k,l)$ 表示的第 i 个细菌在第 j 次趋化, 第 k 次繁殖和第 l 次迁徙所处的位置, 那么细菌 i 的每一步趋化过程可以用如下公式表示:

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

其中 $\mathrm{D}i$ 是位于区间[-1,1]的随机方向向量.

1.2 聚集过程

聚集过程模拟了大肠杆菌在觅食过程中的群体行为, 这种群体行为通过细菌之间相互作用的引力和斥力来实现. 引力使得细菌个体能够一定程度地聚集, 而斥力则让细菌个体都有一定的位置来获取营养维持生存. 细菌在聚集过程中的这种相互作用可以用如下函数表示:

$$\begin{aligned} J_{cc}(q,P(j,k,l)) &= \sum_{i=1}^{S} J_{cc}(q,q^i(j,k,l)) \\ &= \sum_{i=1}^{S}\left[-d_{attractant}\exp\left(-w_{attractant}\sum_{m=1}^{p}(q_m - q_m^i)^2\right)\right] \\ &+ \sum_{i=1}^{S}\left[h_{reprllant}\exp\left(-w_{repellant}\sum_{m=1}^{p}(q_m - q_m^i)^2\right)\right] \end{aligned} \tag{2}$$

其中 $J_{cc}(q,P(j,k,l))$ 为 q 位置的细菌受到细菌群体斥引力值的总和, 它是被附加到实际适应值中的时变函数值. S 为细菌种群个数, p 为问题的维数, $h_{repellant}, w_{repellant}$ $d_{attractant}, w_{attractant}$ 是细菌斥引力参数.

1.3 繁殖过程

根据“适者生存, 优胜劣汰”的生物进化准则, 生存能力较差的细菌个体将在繁殖过程中被淘汰, 细菌的生存能力用健康程度函数表示:

$$J^i{}_{health} = \sum_{j=1}^{Nc+1} J(i,j,k,l) \tag{3}$$

其中, $J(i,j,k,l)$ 为第 i 个细菌所处位置的适应值函数, Nc 为趋化过程的次数. 在繁殖过程中, 对所有细菌的健康程度进行排序, 淘汰健康程度差的半数个体, 复制健康程度好的另一半个体, 复制产生的子代细菌和父代有相同的位置.

1.4 迁徙过程

细菌生活的局部区域环境可能会因为食物消耗殆尽或温度突然升高而发生渐变或突变, 这样有可能导致该区域内细菌群体死亡, 或者集体迁徙到其他新的区域, 迁徙过程正是对这个过程的模拟. 细菌迁徙是按照一定概率 Ped 发生的, 当某个细菌满足迁徙条件时, 该个体将被重新分配到求解空间.

2 改进的菌群觅食算法

2.1 Levy 飞行原理

Levy 飞行[9]特殊的随机游走策略, 它的随机步长服从于一个重尾的概率分布, 这个概率分布被称为 Levy 分布, 满足幂律分布的形式, 可以用公式表示:

$$\mathrm{Levy}(s): \ |s|^{-l}, 1 < l \le 3$$

其中 s 为随机步长, l 为指数参数. 由于 Levy 飞行具有无限且快速增长的方差, 在进行大规模搜索时, 它是一种比布朗随机运动更有效的搜索策略[10][11]. 在搜索过程中, Levy 飞行伴随着频繁短距离局部搜索以及少数长距离全局探索, 因此在搜索到最优值附近时能达到增强局部搜索的效果, 而少数长距离跳跃式的探索有利于扩大搜索范围, 使之更容易跳出局部最优.

近年来人们发现自然界里许多生物的觅食行为同样满足 Levy 飞行模式, 例如信天翁的觅食飞行轨迹[12][13], 果蝇的间歇性飞行觅食轨迹[14], 驯鹿的觅食行为[15], 人类行为中 Ju/'hoansi 狩猎觅食行为[16],甚至在微观世界中热原子蒸汽里光子的运动也存在 Levy 飞行[17]. 目前, 基于 Levy 飞行的搜索策略已被成功应用到一些仿生优化算法[18]中, 并且取得了令人满意的结果.

2.2 Levy 步长的生成

在 Mantegna[19]提出的算法中, 生成 Levy 随机步长的方法可以分成三个部分:

$$s = \frac{u}{|v|^{1/b}} \tag{4}$$

其中参数 u 和 v 由(5)式中正态分布给出, 变量 $b \in [0.3,1.99]$ 在(6)式中也有用到, $\Gamma(z)$ 是 gamma 函数.

$$u : N(0,s_u^2),\ v : N(0,s_v^2) \tag{5}$$

$$s_u = \left(\frac{\Gamma(1+b)\sin(pb/2)}{\Gamma((1+b)/2)\, b 2^{(b-1)/2}}\right)^{1/b},\ s_v = 1 \tag{6}$$

2.3 改进的菌群觅食算法

经典 BFO 算法中的核心参数趋化步长被设计为定值, 而固定的趋化步长不能很好平衡细菌局部搜索和全局收敛性能之间的关系. 一方面, 如果趋化步长设置过大, 则会导致算法在最优值附近扰动, 不利于收敛到最优值; 另一方面, 如果趋化步长设置过小, 则会导致游动总次数剧增, 使算法性能下降. 因此, 选择一个恰当的趋化步长对 BFO 算法是至关重要的. 文献[9][12]的研究表明, Levy 飞行模式是自然中普遍存在的一种高效的觅食策略, 其搜索特性能够很好地平衡全局搜索和局部搜索之间的关系. 为此, 本文借助 Levy 步长来模拟细菌对游动步长的选取. 细菌在搜索过程中, 利用短距离的 Levy 步长来增强局部搜索效果, 借助少数跳跃式的长距离 Levy 步长来扩大游动范围, 避免菌群过早陷入局部最优, 进而从本质上提升算法的性能. 改进的趋化步长公式如下:

$$C(i) = scale * \mathrm{Levy}(l)$$
$$scale = \frac{1}{10p}\sum_{d=1}^{p}\left|q_d^i(j,k,l) - q_d^*(j,k,l)\right| \tag{7}$$

式中 $\mathrm{Levy}(l)$ 是以 l 为参数的 Levy 步长, $scale$ 为尺度系数, $q_d^i(j,k,l)$ 表示第 i 个细菌所处位置的第 d 维坐标值, $q_d^*(j,k,l)$ 为当前最优细菌位置的第 d 维坐标值. 尺度系数 $scale$ 的设计利用到最优细菌和当前细菌的位置关系, 其目的是让细菌 Levy 趋化步长的取值在适当的数量级范围内. 此外, 相关研究还表明, 当多个相互独立的个体对位置随机且稀疏分布的目标进行搜索时, Levy 飞行模式是最理想的搜索策略[12]. 为此, 在细菌迁徙过程中, 本文采用基于 Levy 飞行的随机游走策略来产生细菌迁徙后每个维度的随机位置, 以增强细菌的全局寻优能力, 避免算法过早陷入局部最优. 基于 Levy 飞行的随机游走步长由如下公式给出:

$$stepSize = \frac{\left|X_{\min}^d - X_{\max}^d\right|}{10} * \mathrm{Levy}(l) \tag{8}$$

式中, $\mathrm{Levy}(l)$ 的参数 l 取值与式(7)中的相同, $X_{\min}^d, X_{\max}^d$ 分别为解空间中第 d 维度求解范围的最小值和最大值, $\left|X_{\min}^d - X_{\max}^d\right|/10$ 作为 Levy 飞行随机游走步长的尺度系数, 目的是让 Levy 飞行随机游走在求解空间内, 避免其频繁地超出求解空间的边界.

2.4 LBFO 的算法步骤

综上所述, 基于 Levy 飞行的菌群觅食算法(Levy Flight Based Bacterial Foraging Optimization, LBFO)的流程如图 1 所示, 算法的具体步骤如下:

Step 1. 初始化算法基本参数. 设置菌群规模 S, 趋化次数 Nc, 游动次数 Ns, 繁殖次数 Nre, 迁徙概率 Ped, Levy 步长生成参数 b, 在求解空间里随机初始化细菌的位置, 并计算细菌所处位置的适应值.

Step 2. 执行趋化过程: 产生一个随机游动方向, 根据式(2)计算细菌间的斥引力值并附加到细菌的所处位置的适应值中, 根据式(7)生成趋化步长, 利用式(1)更新细菌游动位置.

Step 3. 若当前游动次数还没达到 Ns 上限或细菌所处位置的适应值有改善, 则转 Step 2, 否则记录当前菌群中最优解个体, 若其最优值比历史最优解更优, 则更新历史最优解, 否则历史最优解不变.

Step 4. 执行繁殖过程: 根据式(3)计算每个细菌在其生命周期内的健康程度, 淘汰健康程度差的半数

细菌, 对健康程度较好的另一半细菌进行复制.

Step 5. 执行迁徙过程: 对每个细菌生成一个随机概率, 并将它与迁徙概率 Ped 进行比较, 如果该值小于 Ped , 那么根据式(8)提供的随机步长在求解空间内进行随机游走, 产生细菌新的候选位置, 否则不做任何处理.

Step 6. 若满足循环结束条件或停止准则, 则停止计算并输出最优值及其相应的参数, 否则转向 Step 2.

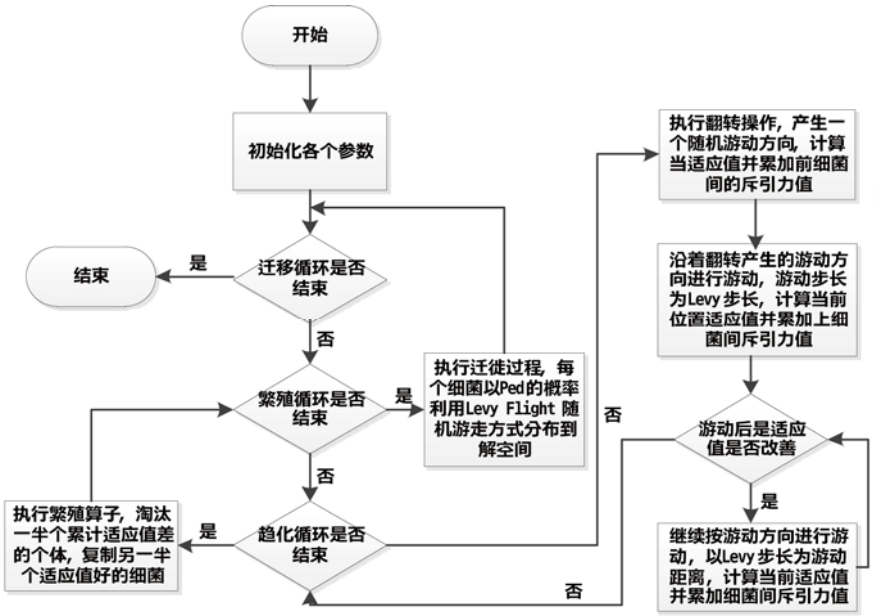

图 1 基于 Levy 飞行的菌群觅食算法流程

3 仿真实验

为了测试 LBFO 算法的优化性能, 本文采用 10 个基准测试函数进行数值实验, 目前这些测试函数已广泛应用于优化算法性能的评价上, 其中 f_1, f_6, f_8 为单峰函数, 其余的是含有大量局部极值的多峰多模态函数, 如表 1 所示. 在算法比较上, 本节着重将 LBFO 算法与经典 BFO 算法、ABFO 算法和 SA-BFO 算法进行比较. 由于菌群觅食算法涉及较多的参数取值, 而各种参数取值目前尚无确切的理论依据, 本文算法所设置的参数值是根据反复实验获得的经验值来确定的. 为了实验的可比性, 四种算法除了各自特定参数和趋化步长的取值不同外, 其余参数均相同. 具体地, 经典 BFO 中的趋化步长 $C(i) = 0.1$, SA-BFO 中算法参数 $C_{initial} = 0.5$, $e_{initial} = 100$, $Ku = 10, a = 10$, $b = 10$, ABFO 中自适应参数 $l = 400$, LBFO 中 Levy 步长生成参数 $b = 1.5$, 其余的公共参数分别为: 种群规模 $S = 50$, 趋化次数 $Ns = 50$, 游动次数 $Ns = 12$, 繁殖次数 $Nre = 16$, 迁徙次数 $Ned = 4$, 迁徙概率 $Ped = 0.25$, 引力深度 $d_{attractant} = 0.1$, 引力宽度 $w_{attractant} = 0.2$, 斥力深度 $h_{repellant} = 0.1$, 斥力宽度 $w_{repellant} = 10$. 在算法优化性能比较上, 本文主要关注最优解的质量和收敛效率. 对每个测试函数, 在相同条件下分别独立运行 20 次, 每次运行以达到预先设定的阈值 ($Epsilon = 1.000E-10$)或最大函数评估次数为停止准则, 并记录实际找到的最优值、平均最优值和方差, 运行结果如表 2 所示. 表 3 是对表 2 里每个测试样例最优算法和次优算法做的 t-test 统计检验, 用于分析这两种算法的优化性能是否有明显差异.

表 1 基准测试函数及其对应的参数

函数名	函数表达式	搜索空间	最优值
Sphere	$F_1(x) = \sum_{i=1}^{d} x_i^2$	[-100,100]	$F_1(\mathbf{0}) = 0$

Griewank	$F_2(x)=\frac{1}{4000}\sum_{i=1}^{d}x_i^2-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600,600]	$F_2(\vec{0})=0$
Ackley	$F_3(x)=-20\exp\left(-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{d}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{d}\cos(2\pi x_i)\right)+20+e$	[-30,30]	$F_3(\vec{0})=0$
Rastrigin	$F_4(x)=\sum_{i=1}^{d}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	[-5.12,5.12]	$F_4(\vec{0})=0$
Levy	$F_5(x)=\sum_{i=1}^{d-1}(w_i-1)^2\left[1+10\sin^2(\pi w_i+1)+(w_d-1)^2[1+\sin^2(2\pi w_d)]\right]+\sin^2(\pi w_1),\quad w_i=1+\frac{x_i-1}{4}$	[-10,10]	$F_5(\vec{0})=0$
Rosenbrock	$f_6(x)=\sum_{i=1}^{d-1}\left[100(x_{i+1}-x_i^2)^2+(x_i-1)^2\right]$	[-30,30]	$f_6(\vec{0})=0$
Colville	$f_7(x)=100(x_1^2-x_2^2)^2+(x_1-1)^2+(x_3-1)^2+90(x_3^2-x_4)^2+10.1((x_2-1)^2+(x_4-1)^2)+19.8(x_2-1)(x_4-1)$	[-10,10]	$f_7(\vec{0})=0$
Beale	$f_8(x)=(1.5-x_1+x_1x_2)^2+(2.25-x_1+x_1x_2^2)^2+(2.625-x_1+x_1x_2^3)^2$	[-4.5,4.5]	$f_8(\vec{0})=0$
Schaffer N.2	$f_9(x)=0.5+\frac{\sin 2(x_1^2-x_2^2)-0.5}{\left[1+0.001(x_1^2+x_2^2)\right]^2}$	[-100,100]	$f_9(\vec{0})=0$
Eggcrate	$f_{10}(x)=x_1^2+x_2^2+25(\sin^2x_1+\sin^2x_2)$	[-6.28 ,6.28]	$f_{10}(\vec{0})=0$

表 2　4 种算法优化实验结果比较

Fn(Dim.)	Max FEs	Index	BFO	ABFO	SA-BFO	LBFO
$f_1(D=20)$	2×10^5	Best	1.320E-001	2.260E-002	3.904E-002	**1.867E-003**
		Mean	1.869E-001	2.374E-002	7.046E-002	**4.050E-003**
		Std	1.382E-003	**6.503E-007**	2.082E-004	2.608E-006
$f_2(D=20)$	2×10^5	Best	8.690E+001	3.056E+001	7.525E+001	**3.033E-002**
		Mean	1.518E+002	3.370E+001	7.804E+001	**6.350E-002**
		Std	6.588E+002	1.806E+000	1.598E+000	**1.263E-003**
$f_3(D=10)$	1×10^5	Best	2.273E+000	1.644E+001	1.467E+001	**1.155E-006**
		Mean	1.420E+001	1.761E+001	1.737E+001	**1.373E-004**
		Std	1.904E+001	3.010E-001	9.511E-001	**2.946E-008**
$f_4(D=10)$	1×10^5	Best	1.082E+001	**3.100E+000**	1.060E+001	1.094E+001
		Mean	1.398E+001	**7.105E+000**	1.985E+001	3.379E+001
		Std	**4.553E+000**	1.068E+001	2.954E+001	2.112E+002
$f_5(D=5)$	5×10^4	Best	1.010E-003	1.940E-004	5.568E-004	**3.671E-006**
		Mean	1.007E-001	6.735E-001	5.511E-001	**2.373E-005**
		Std	3.328E-002	1.184E-001	1.858E-001	**1.372E-010**
$f_6(D=5)$	5×10^4	Best	1.804E-001	**4.545E-006**	1.076E-001	2.909E-004
		Mean	8.170E-001	**2.188E-001**	3.474E-001	2.592E+000
		Std	1.330E-001	1.905E-002	**1.779E-002**	4.093E+001
$f_7(D=4)$	5×10^4	Best	4.606E-003	1.342E-006	2.094E-003	**1.269E-010**
		Mean	5.149E-002	7.478E-002	1.560E-002	**5.315E-008**
		Std	1.151E-003	1.344E-002	8.542E-005	**2.469E-014**

$f_8(D=2)$	5×10^4	Best	1.504E-007	2.792E-010	5.076E-008	**3.034E-012**
		Mean	3.620E-006	7.180E-005	1.366E-006	**5.606E-011**
		Std	5.521E-012	5.548E-008	1.716E-012	**7.249E-022**
$f_9(D=2)$	5×10^4	Best	2.151E-002	3.127E-003	4.072E-006	**2.853E-008**
		Mean	1.275E-001	1.234E-001	1.350E-001	**1.305E-003**
		Std	4.866E-003	1.020E-002	9.715E-003	**8.902E-006**
$f_{10}(D=2)$	5×10^4	Best	7.633E-007	1.201E-010	1.384E-007	**7.914E-012**
		Mean	3.772E-005	8.492E-005	4.290E-006	**6.019E-011**
		Std	1.742E-009	1.693E-008	1.218E-011	**6.107E-022**

本文经典 BFO 算法的实现以 Passino 提供的 Matlab 原始代码[20]为准. 从表 2 的实验结果和表 3 的 t-test 统计检验结果可以得出: LBFO 在 f_4 上优化结果差于 ABFO, 在 f_6 上和 ABFO 无明显差别, 但是对于其他测试函数, LBFO 找到的最优值、平均最优值和方差均优于 BFO、ABFO 和 SA-BFO, 在显著性水平 $a=0.05$ 的条件下, LBFO 找到的平均最优值明显优于其他 3 个算法, 可见在同等的条件下, LBFO 表现出了较高的求解精度和良好的求解稳定性, 说明采用借助 Levy 飞行搜索特性改进的细菌具有良好的探索和开发能力. 表 2 主要针对普通维度情况下 4 种算法优化性能的比较, 表 4 给出了若干测试函数在超高维情况下 20 次独立运行的实验结果. 观察表 4 可知: 在超高维求解空间里, LBFO 在 $f_1(D=50)$, $f_2(D=100)$ $f_4(D=200)$ 上取得了相对较优的结果, 而在其他两个测试函数上优化效果较差, 可以认为 LBFO 在求解超高维优化问题时没有体现出明显的优势, 这是因为在超高维求解空间里, 细菌通过随机翻转很难找一个向全局最优值靠拢的游动方向, 尽管细菌的游动步长采用 Levy 步长, 但由于游动方向的影响, 使得 LBFO 难以发挥其搜索特性. 因此, 在面对超高维的优化问题时, LBFO 还存在一定的缺陷.

在收敛性能方面, 本文比较 4 种算法在达到指定精度下的平均 FEs 迭代次数和平均繁殖代数. 对每个测试函数, 在相同条件下分别独立运行 20 次, 并记录平均 FEs 迭代次数和平均繁殖代数. 当达到预先设定的精度阈值时算法停止运行, 否则, 将在达到最大繁代数($Nre * Ned$)上限时停止运行, 实验结果如表 5 所示. 观察表 5 可以发现: 除 f_4 外, LBFO 的平均 FEs 迭代次数和平均繁殖代数在其他 9 个测试函数上均小于 BFO、ABFO 以及 SA-BFO. 因此, 在达到相同的求解精度情况下, 对大部分测试函数而言, LBFO 迭代的次数最少, 收敛性能更优. 为了形象展示 4 种算法在收敛性能上的差异, 图 3 给出了目标函数适应值随 FEs 迭代次数的变化曲线, 这里选取基准测试函数 f_1 : f_9, 每个测试函数均是用 4 种算法一次独立运行的结果.

算法参数是影响算法性能和效率的重要因素, 菌群觅食算法中种群规模 S 是影响算法优化性能的又一核心参数. 小规模的种群虽然能够提高算法的计算效率, 但是减低了菌群的多样性, 削弱了优化性能; 较大规模的种群虽然能减低细菌陷入局部最优的概率, 但计算量随之递增, 影响算法的总体性能. 为了探究 LBFO 算法性能与种群规模 S 的关系, 本文选取 f_1, f_2, f_3 为测试样例, 分别比较 $S=50,100,200$ 情况下 LBFO 算法的收敛曲线. 观察图 2, 仅管在 f_1, f_2 中 $S=50$ 与 $S=100$ 最终得到的最优值相近, 但是 $S=50$ 的收敛曲线运行在最下方, 有着较好的收敛性能. 随着种群规模的成倍递增, LBFO 算法的求解精度不仅没有提高, 总体收敛性能反而下降了, 其原因在于较大规模的菌群降低了繁殖过程中淘汰劣势细菌的速度, 进而对菌群间的相互作用产生了一定的影响, 不利于算法局部搜索, 而增加种群规模带来的种群多样性的增强并没有弥补上述带来的问题. 因此, LBFO 算法的种群规模不宜设置过大.

综上所述, 虽然 LBFO 算法在处理超高维优化问题时还存在一定缺陷, 但是它在处理其他维度下复杂多峰多模态的函数优化问题时, 表现出良好的全局寻优能力和较高的搜索精度, 所以采用 Levy 飞行模式来模拟菌群的游动和迁徙过程更符合细菌实际觅食的情况, 体现了本文算法的有效性和价值所在.

表 3　关于表 2 数据的 t-test 检验结果

Fn (Dim)	Std.Err	t	95% Confidence interval	Two-tailed P
$f_1(D = 20)$	0.001309	-47.555657	(-0.020531, -0.018854)	1.80857E-35<0.05
$f_2(D = 20)$	0.975417	-109.045598	(-34.259970, -33.011105)	4.68059E-49<0.05
$f_3(D = 10)$	3.165799	-14.187782	(-16.230231, -12.176934)	9.13960E-17<0.05
$f_4(D = 10)$	10.806707	7.808957	(19.768055, 33.604306)	2.03395E-09<0.05
$f_5(D = 5)$	0.312689	-5.573035	(-0.751242, -0.350893)	2.11693E-02<0.05
$f_6(D = 5)$	4.642378	1.616325	(-0.599066, 5.344753)	0.114297 > 0.05
$f_7(D = 4)$	0.006705	-7.356396	(-0.019890, -0.011305)	8.14397E-09<0.05
$f_8(D = 2)$	0.000001	-4.544490	(-0.000002, -0.000001)	5.44689E-05<0.05
$f_9(D = 2)$	0.071539	-5.910173	(-0.179501, -0.087907)	7.56077E-07<0.05
$f_{10}(D = 2)$	0.000003	-5.359083	(-0.000006, -0.000003)	4.30425E-06<0.05

表 4　4 种算法优化在高维情况下的实验结果比较

Fn(Dim.)	Max FEs	Index	BFO	ABFO	SA-BFO	LBFO
$f_1(D = 50)$	2×10^5	Best	1.228E+04	1.767E-01	2.231E-01	**3.042E-02**
		Mean	1.671E+04	2.087E-01	2.841E-01	**4.739E-02**
		Std	4.722E+06	2.534E-04	6.302E-04	**1.217E-04**
$f_2(D = 100)$	2×10^5	Best	9.854E+02	3.603E+02	5.338E+02	**8.493E-01**
		Mean	1.364E+03	4.378E+02	6.235E+02	**1.059E+00**
		Std	2.250E+04	1.562E+03	4.642E+03	**9.138E-03**
$f_3(D = 150)$	2×10^5	Best	1.374E+01	1.303E+01	**1.297E+01**	1.353E+01
		Mean	1.830E+01	1.779E+01	**1.774E+01**	1.898E+01
		Std	**3.618E+00**	5.613E+00	4.187E+00	5.152E+00
$f_4(D = 200)$	2×10^5	Best	8.477E+02	1.242E+03	1.186E+03	**6.612E+02**
		Mean	9.960E+02	1.496E+03	1.520E+03	**7.899E+02**
		Std	6.998E+03	2.513E+04	2.405E+04	**6.591E+03**
$f_5(D = 250)$	2×10^5	Best	**2.450E+02**	3.934E+02	3.806E+02	3.652E+02
		Mean	**3.197E+02**	4.886E+02	4.816E+02	4.386E+02
		Std	**1.449E+03**	4.072E+03	2.545E+03	1.707E+03

表 5　4 种优化算法的收敛性能比较

Fn (Dim)	Epsilon	Reproduction Count				FEs Iteration			
		BFO	ABFO	SA-BFO	LBFO	BFO	ABFO	SA-BFO	LBFO
$f_1(D = 20)$	1×10^{-2}	64.0	64.0	64.0	**57.1**	160000	160000	160000	**141449**
$f_2(D = 20)$	1×10^{-1}	64.0	64.0	64.0	**39.2**	160000	160000	160000	**97136**
$f_3(D = 10)$	1×10^{-1}	64.0	64.0	64.0	**32.2**	160000	160000	160000	**80458**
$f_4(D = 10)$	1×10^{1}	64.0	**27.0**	58.5	64.0	160000	**67482**	146296	160000
$f_5(D = 5)$	1×10^{-1}	62.8	61.0	64.0	**8.5**	156774	152223	160000	**19897**
$f_6(D = 5)$	1×10^{-1}	64.0	30.6	63.8	**19.6**	160000	75276	159391	**47830**

$f_7(D=4)$	1×10^{-1}	64.0	41.5	64.0	**11.5**	160000	102540	160000	**27514**
$f_8(D=2)$	1×10^{-5}	7.3	8.6	3.8	**1.1**	16843	20041	8357	**1728**
$f_9(D=2)$	1×10^{-1}	30.3	50.0	38.1	**1.0**	74119	124314	94359	**704**
$f_{10}(D=2)$	1×10^{-5}	40.3	19.4	10.4	**3.0**	99883	46816	25018	**5912**

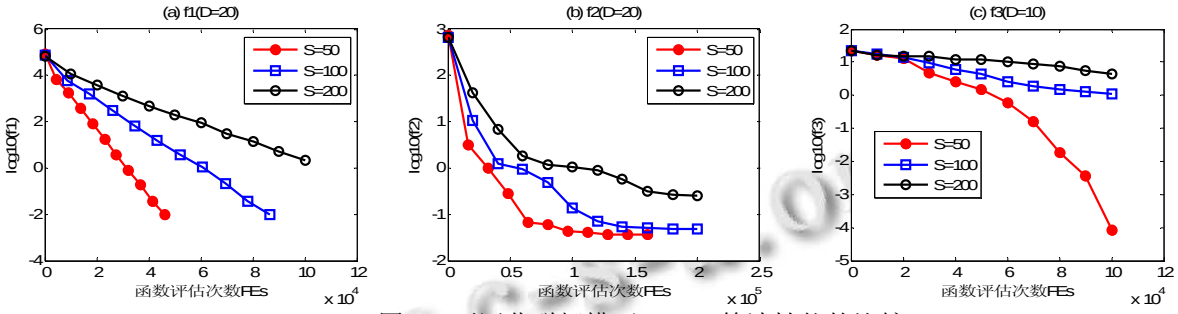

图 2 不同菌群规模下 LBFO 算法性能的比较

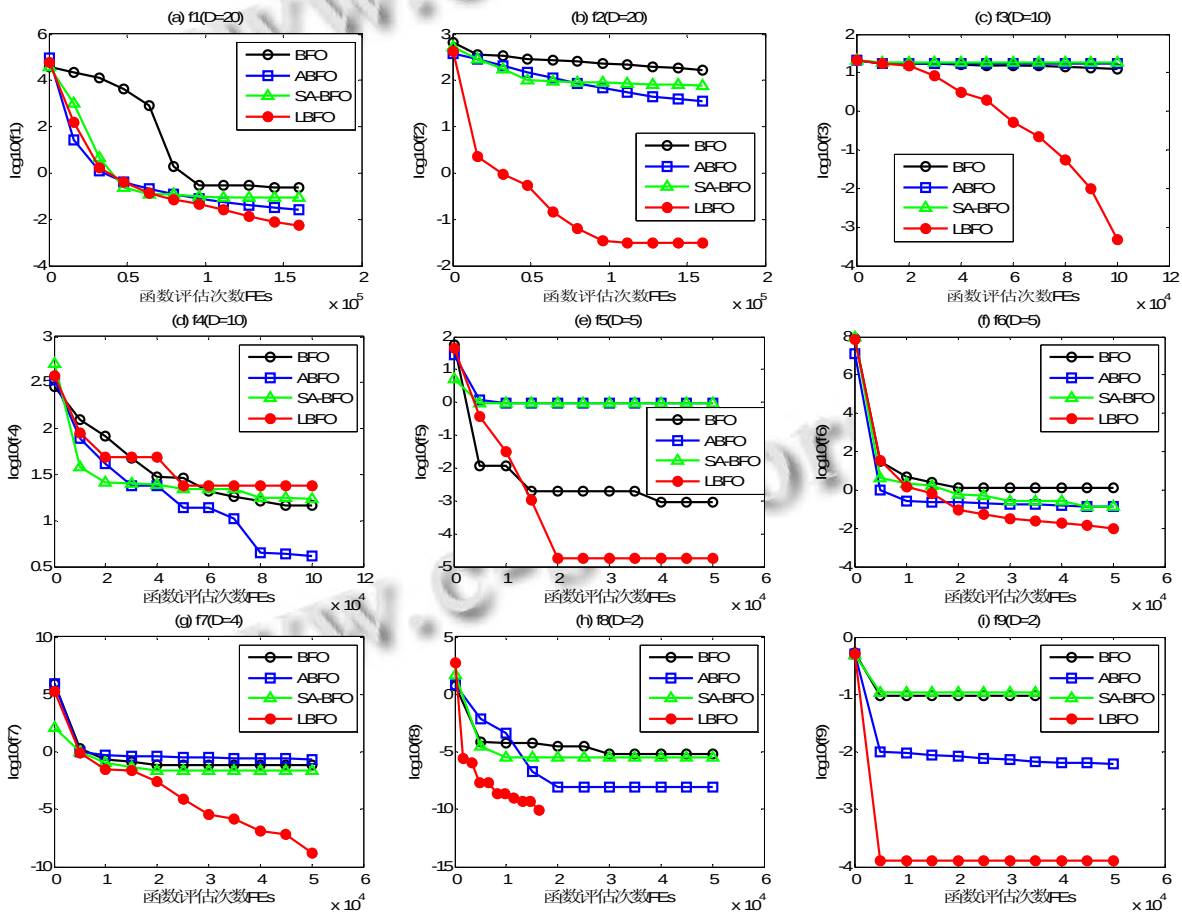

图 3 4 种优化算法在 f1-f9 的适应值收敛曲线

4 结语

Levy 飞行是自然界中普遍存在的高效觅食策略, 受 Levy 飞行的启发, 针对经典菌群觅食算法中固定趋化步长导致的求解精度不高和收敛性能差的缺陷, 本

文借助 Levy 飞行搜索的特性, 引入基于 Levy 分布的趋化步长以达到增强算法局部求精能力, 提高算法收敛性能的效果. 另外, 在迁徙过程中利用 Levy 飞行随机游走策略来重新分配细菌的候选位置, 以利于算法跳出局部最优. 最后, 通过对 10 个常用的基准测试函数进行数值实验, 并进行了 t-test 统计检验, 实验结果表明, 除超高维的函数优化问题以外, 对于绝大部分测试函数而言, LBFO 算法在求解精度以及收敛性能上均有明显的改进, 因此, 本文提出的基于 Levy 飞行的 LBFO 算法是可行和有效的.

参考文献

1 Passino KM. Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Systems, 2002, 22(3): 52–67.

2 Mishra S, Bhende CN. Bacterial foraging technique-based optimized active power filter for load compensation. IEEE Trans. on Power Delivery, 2007, 22(1): 457–465.

3 Mishra S. A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation. IEEE Trans. on Evolutionary Computation, 2005, 9(1): 61–73.

4 Kim DH, Cho CH. Bacteria foraging based neural network fuzzy learning. IICAI. 2005, 2005: 2030–2036.

5 Mishra S. Hybrid least-square adaptive bacterial foraging strategy for harmonic estimation. IEE Proceedings: Generation Transmission and Distribution, IET, 2005, 152(3): 379–389.

6 Chen H, Zhu Y, Hu K. Self-adaptation in bacterial foraging optimization algorithm. Intelligent System and Knowledge Engineering, 2008, 1: 1026–1031.

7 Dasgupta S, Das S, Abraham A, et al. Adaptive computational Chemotaxis in bacterial foraging optimization: an analysis. IEEE Trans. on Evolutionary Computation, 2009, 13(4): 919–941.

8 Das S, Biswas A, Dasgupta S, et al. Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications. Foundations of Computational Intelligence. Springer Berlin Heidelberg, 2009, 3: 23–55.

9 Viswanathan GM, Afanasyev V, Buldyrev SV, et al. Lévy flights search patterns of biological organisms. Physica A: Statistical Mechanics and its Applications, 2001, 295(1): 85–88.

10 Yang XS. Nature-inspired metaheuristic algorithms. Luniver Press, 2010.

11 Gutowski M. Levy Flights as an Underlying Mechanism for Global Optimization Algorithms. arXiv preprint math-ph/0106003, 2001.

12 Viswanathan GM, Afanasyevt V. Lévy flight search patterns of. Nature, 1996, 381: 30.

13 Edwards AM, Phillips RA, Watkins NW, et al. Revisiting Lévy flight search patterns of wandering albatrosses, bumblebees and deer. Nature, 2007, 449(7165): 1044–1048.

14 Reynolds AM, Frye MA. Free-flight odor tracking in drosophila is consistent with an optimal intermittent scale-free search. PloS one, 2007, 2(4): e354.

15 Marell A, Ball JP, Hofgaard A. Foraging and movement paths of female reindeer: insights from fractal analysis, correlated random walks, and Lévy flights. Canadian Journal of Zoology, 2002, 80(5): 854–865.

16 Brown CT, Liebovitch LS, Glendon R. Lévy flights in dobe juhoansi foraging patterns. Human Ecology, 2007, 35(1): 129–138.

17 Mercadier N, Guerin W, Chevrollier M, et al. Lévy flights of photons in hot atomic vapours. Nature Physics, 2009, 5(8): 602–605.

18 Yang XS, Deb S. Cuckoo search via Lévy flights. Nature & Biologically Inspired Computing, 2009: 210–214.

19 Mantegna RN. Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes. Physical Review E, 1994, 49: 4677–4683.

20 Passino KM. Matlab Code for Biomimicry for Optimization Control and Automation. http://www.ece.osu.edu/passino/ICbook/ic index.html.