文章编号: 1001-0920 (2012) 09-1402-04

基于分段搜索策略的改进蜂群算法

罗 钧, 肖向海, 付 丽, 王 强

(重庆大学 光电技术及系统教育部重点实验室，重庆 400030)

摘 要: 针对基本人工蜂群算法在解决优化问题时收敛速度不够快、易陷入局部最优的缺陷, 提出一种改进蜂群算法. 该算法采用“分段搜索”方式对食物源进行贪婪更新, 以提高食物源更新的成功率; 同时, 招募所有观察蜂选择当前最优食物源, 以实现对最优食物源的充分优化. 对经典测试函数反复实验的结果表明, 改进算法计算结果稳定, 与基本蜂群算法相比, 加速收敛效果非常明显, 全局搜索能力显著提高, 运行时间大大缩短.

关键词: 人工蜂群；改进算法；分段搜索；充分优化

中图分类号: TP301.6 **文献标志码:** A

Modified artificial bee colony algorithm based on segmental-search strategy

LUO Jun, XIAO Xiang-hai, FU Li, WANG Qiang

(Key Laboratory of Optoelectronic Technology and System of Ministry of Education，Chongqing University，Chongqing 400030，China．Correspondent：LUO Jun，E-mail：luojun@cqu.edu.cn)

Abstract: For the problem that when using the basic artificial bee colony(ABC) algorithm to solve the optimal problems, it can not converge so fast and can trap in a local optimal solution easily. Therefore, a modified artificial bee colony algorithm is proposed. To improve the updating rate of food sources, the segmental-search strategy is used, and the way of the onlookers choosing the food sources is modified to optimize the best food source fully. Many experiment results of classic functions show that the modified algorithm owns steady performance, has great advantage of convergence property and global optimizing ability, and also runs faster than ABC algorithm.

Key words: artificial bee colony(ABC)；modified algorithm；segmental-search；optimizing fully

1 引 言

优化问题是实际工程应用中需要经常面对的问题, 为此人们先后提出了多种优化算法. 相关研究[1-2]已经证明, Karaboga[3]于2005 年提出的基于蜂群采蜜过程的人工蜂群算法 (ABC), 相比于遗传算法、差分算法、粒子群算法等其他优化算法, 在收敛速度和计算精度等方面具有更好的性能. 尽管如此, ABC 算法仍然存在着早熟收敛、搜索精度不高、易陷入局部最优等缺点.

针对上述问题, 本文提出一种基于分段搜索策略的改进蜂群算法. 不同于借鉴其他群智能算法的改进ABC 算法[4-6], 所提出的方法是对 ABC 算法本身进行改造, 优化了食物源的更新方式, 并简化了观察蜂对食物源的选择方式, 提高了搜寻过程的寻优效率, 能使蜂群快速找到最优食物源, 即找出目标函数的全局最优解.

2 基本人工蜂群算法

ABC 算法模型[7-8]中包含 3 种角色的蜜蜂, 即雇佣蜂、观察蜂和侦察蜂. 雇佣蜂和观察蜂各占整个蜂群规模的一半, 每只雇佣蜂与食物源是一一对应的. 蜂群搜寻最优食物源的过程就是寻找最优目标值的过程. 算法以随机初始化一定数量的食物源开始, 其后的每一次迭代搜寻过程概括如下: 雇佣蜂在食物源邻域内随机搜索出一个新食物源并检查其适应度, 若大于原来的食物源则用新食物源替代之; 观察蜂通过分享雇佣蜂带回的食物源信息, 按食物源适应度的大小概率采用轮盘赌的方式选择一个食物源, 并按照与雇佣蜂一样的贪婪选择机制更新所选择的食物源; 当

收稿日期: 2011-01-22；**修回日期:** 2011-04-27.

基金项目: 国防科工委国防军工计量“十一五”计划重点项目(B20301118).

作者简介: 罗钧(1963−), 男, 教授, 从事测试计量等研究； 肖向海(1987−), 男, 硕士生, 从事信息获取与处理的研究.

某个食物源在迭代 limit 次后仍没有改进时, 则被强制放弃, 其对应的雇佣蜂转变为侦察蜂, 重新随机搜寻一个新的食物源.

设蜂群规模为 $2S$, 目标函数 $f(X)$ 是一个 D 维的优化问题, 初始化随机产生 S 个食物源 $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD}), i = 1, 2, \cdots, S$. 雇佣蜂和观察蜂按下式对食物源邻域进行随机搜索:

$$x'_{ij} = x_{ij} + \psi(x_{ij} - x_{kj}). \tag{1}$$

其中: $j \in \{1, 2, \cdots, D\}, k \in \{1, 2, \cdots, S\}$ 且 $k \neq i, \psi$ 为 $[-1, 1]$ 内的随机数. 随着迭代过程的进行, 参数 x_{ij} 与 x_{kj} 之间差距缩小, 搜索的步长也随之缩小, 因而算法逐渐收敛.

依据食物源适应度的大小, 观察蜂采用轮盘赌方式进行食物源选择, 其选择概率如下:

$$P_i = \frac{\mathrm{fit}(X_i)}{\sum_{i=1}^{n} \mathrm{fit}(X_i)}, \tag{2}$$

其中 $\mathrm{fit}(X_i)$ 为食物源 X_i 的适应度.

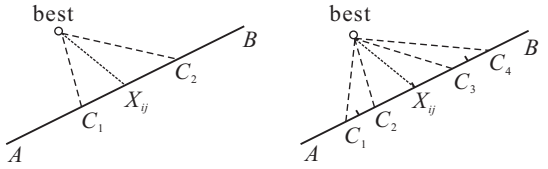
3　基于分段搜索的改进蜂群算法

3.1　食物源更新方式的改进策略

ABC 算法是受自然蜂群的启发而提出的人工蜂群算法. 分析其搜索模式不难发现, 雇佣蜂在对食物源某一维进行更新时, 按式 (1) 搜索到的结果具有较大的偶然性. 如图 1(a) 所示, AB 代表以 x_{ij} 为中心的 $\pm|x_{ij} - x_{kj}|$ 搜索范围, 由于 ψ 为 [−1,1] 内的随机数, 其搜索结果可能取到 AB 内的任意一个值. 若取到比 x_{ij} 更优的 C_1, 则在进行贪婪选择时将会替代 x_{ij} 而更新食物源; 若取到比 x_{ij} 更差的 C_2, 则不会更新. 可见, 正是这种偶然性, 使得食物源的更新成功率较低. 为此, 改进算法采用“分段搜索”的方式对食物源进行贪婪更新, 即在更新某一维时, 将搜索范围等分成若干个区间进行搜索, 从中随机选取的区间代表值如下:

$$x_{ij}^{n} = R_l + (n - \mathrm{rand}(0, 1)) * \frac{R}{N}. \tag{3}$$

其中: 搜索范围下界 $R_l = x_{ij} - |x_{ij} - x_{kj}|$, 搜索范围 $R = 2|x_{ij} - x_{kj}|, N$ 为设置的分段区间数, 分段区间序号 $n \in \{1, 2, \cdots, N\}$. 如图 1(b) 所示, 将 AB 等分成 4 个区间, 从中随机选取 C_1, C_2, C_3, C_4 作为各自所在区间的代表值, 将其代入食物源并计算由此得到的目标函数值, 选出使目标函数值最好的 C_2 作为这一维的最终更新值, 最后再将 C_2 与原值 x_{ij} 进行贪婪选择. 由于区间代表值的好坏从一定程度上代表了其所在区间的好坏, 而更新值 C_2 是若干个区间代表值中最好的, 因此 C_2 在整个搜索范围内都是比较好的更新值. 这样, 食物源更新的成功率便大大提高了, 算法进化能力也随之增强.



(a) 单维随机搜索　　(b) 单维分段搜索

图 1　食物源单维更新示意图

3.2　观察蜂选择机制的改进策略

ABC 算法中的观察蜂按式 (2) 采用轮盘赌方式选择食物源. 这样, 虽然有利于保证多样性, 防止早熟收敛, 但也导致观察蜂分散于多个食物源中, 使每个食物源只能被更新较少的次数. 对于最优食物源而言, 这种更新显然不够充分, 从而最优解进化缓慢. 同时, 轮盘赌选择方式的计算复杂度较高, 使算法的计算量较大.

为了优化 ABC 算法的性能, 可借鉴粒子群算法 (PSO) 的全局寻优思想[9]. PSO 算法在每次迭代过程中不断修正粒子群的速度和位置, 使其靠近距离最优解最近的粒子, 最终找到区域内的最优解. 受 PSO 算法思想的启发, 改进蜂群算法是在每次迭代过程中让观察蜂全部选择雇佣蜂所带回食物源中的最优解, 并按“分段搜索”方式对其进行连续更新, 从而实现对最优解的充分优化, 有效避免由于观察蜂分散所导致的最优解进化缓慢的问题. 因为观察蜂直接选择最优食物源, 所以不需进行食物源适应度计算和轮盘赌选择计算. 同时, “分段搜索”的更新方式以及多次连续更新的充分优化, 保证了当前最优解的进化能力, 使算法更容易跳出局部最优. 其他食物源在各自对应的雇佣蜂的作用下继续更新, 最终向最优食物源靠拢.

3.3　改进蜂群算法流程(求最小值)

改进后的人工蜂群算法称为 SABC 算法, 其实现步骤如下.

Step 1: 初始化随机产生 S 个食物源, 并与雇佣蜂一一对应.

Step 2: 每个雇佣蜂按“分段搜索”方式随机搜索出其所在食物源附近的一个较优解, 并更新食物源. 具体步骤如下:

Step 2.1: 按式 (3) 从食物源单维的搜索范围内随机挑选出 N 个区间代表值 $C_1, C_2, \cdots, C_N$;

Step 2.2: 选出使函数值 $f(X_i)$ 最小的区间代表值, 并将其作为这一维的最终更新值 C_{best}, 即 C_{best} 满足

$$f(X_i)^{\mathrm{best}} = \min\{f(X_i)^1, f(X_i)^2, \cdots, f(X_i)^N\}, \tag{4}$$

其中 $f(X_i)^n$ 为将 C_n 代入食物源所得到的函数值;

Step 2.3: 将 C_{best} 与原值 x_{ij} 进行贪婪选择, 以更新食物源.

Step 3: 雇佣蜂将更新后的食物源信息带回蜂巢, 所有观察蜂选择食物源中的最优解, 即最小解 $f(X)_{\min}$, 并按“分段搜索”方式对 $f(X)_{\min}$ 进行连续更新, 实现其充分优化.

Step 4: 记录当次迭代中充分优化后的最优解, 并替换之前记录的最优解.

Step 5: 若某一食物源在超过 limit 次循环后仍没有更新, 则放弃该食物源, 并使对应的雇佣蜂成为侦查蜂, 按下式重新随机搜寻食物源:

$$x_i^j = x_{\min}^j + \mathrm{rand}(0,1)(x_{\max}^j - x_{\min}^j), \tag{5}$$

其中 $x_{\max}^j$ 和 $x_{\min}^j$ 分别为食物源第 j 维的最大值和最小值.

Step 6: 判断是否满足终止条件. 若不满足, 则转至 Step 2; 否则, 输出最优解.

4 仿真实验及分析

4.1 SABC 与 ABC 的寻优结果比较

在 Matlab 编程环境下, 选取文献 [2] 中的 5 个经典测试函数进行实验, 如表 1 所示. 测试函数的搜索空间维数为 20 和 40, 搜索范围分别为 $[-100,100]$, $[-16,32]$, $[-2.56,5.12]$, $[-10,10]$, $[-5,10]$. 两种算法的参数设置如下: 种群规模均为 100, 最大迭代次数均为 1 500, limit 均为 125, SABC 算法中的分段区间数为 3. 将 ABC 算法与 SABC 算法重复运行 10 轮, 得到的目标函数最优值和平均值如表 2 所示.

表 1 测试函数

函数名称	函数定义
Griewank	$f_1(x)=\frac{1}{4\,000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$
Ackley	$f_2(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$
Rastrigin	$f_3(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$
SumSquares	$f_4(x)=\sum_{i=1}^{n}ix_i^2$
Zakharov	$f_5(x)=\sum_{i=1}^{n}x_i^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^4$

表 2 两种算法的运行结果比较

函数	维数	算法	平均值	最优值	标准差
f_1	20	ABC	1.154 63 e-014	2.775 56 e-015	1.51601 e-014
		SABC	0	0	0
	40	ABC	3.264 63 e-012	3.687 05 e-013	5.308 16 e-012
		SABC	0	0	0
f_2	20	ABC	2.859 93 e-014	2.220 45 e-014	6.355 29 e-015
		SABC	1.794 12 e-014	1.509 9 e-014	3.891 8 e-015
	40	ABC	1.304 83 e-006	9.222 33 e-007	4.876 93 e-007
		SABC	5.259 79 e-013	2.353 67 e-013	2.865 66 e-013
f_3	20	ABC	1.136 87 e-014	0	1.55 672 e-014
		SABC	0	0	0
	40	ABC	6.429 79 e-010	1.961 1 e-011	7.257 2 e-010
		SABC	6.821 21 e-014	5.684 34 e-014	2.542 11 e-014
f_4	20	ABC	3.559 51 e-016	3.074 56 e-016	4.427 08 e-017
		SABC	1.722 73 e-022	1.135 64 e-022	7.043 31 e-023
	40	ABC	1.391 85 e-012	1.652 41 e-013	1.114 2 e-012
		SABC	3.329 76 e-021	1.824 51 e-021	1.426 01 e-021
f_5	20	ABC	3.233 51 e-016	2.424 85 e-016	6.745 88 e-017
		SABC	1.420 64 e-022	9.633 96 e-023	5.996 32 e-023
	40	ABC	1.065 82 e-009	7.345 81 e-010	4.699 13 e-010
		SABC	4.879 57 e-021	4.048 71 e-021	1.067 09 e-021

由表 2 可以看出, SABC 算法能获得更好的最优值, 而且 SABC 算法的平均值和标准差更小, 说明性能更稳定. 两种算法对应的平均进化曲线 (40 维) 如图 2 ~ 图 6 所示.

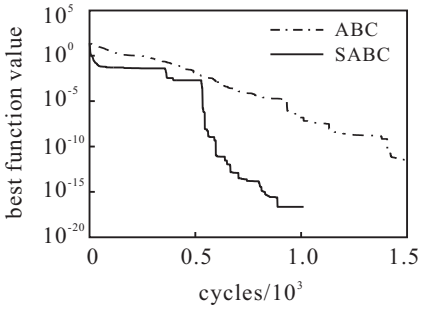


图 2 Griewank 平均进化曲线比较

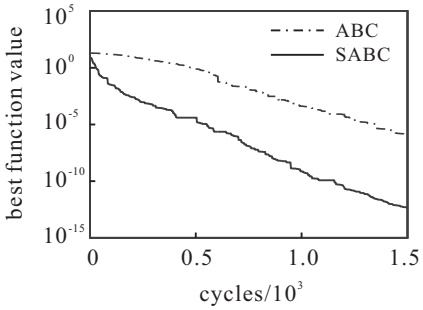


图 3 Ackley 平均进化曲线比较

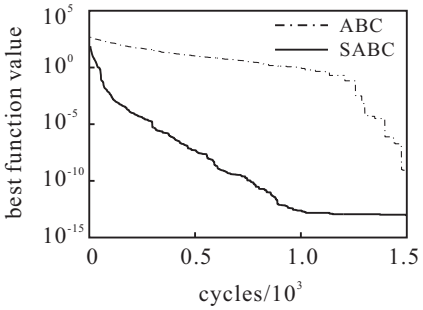


图 4 Rastrigin 平均进化曲线比较

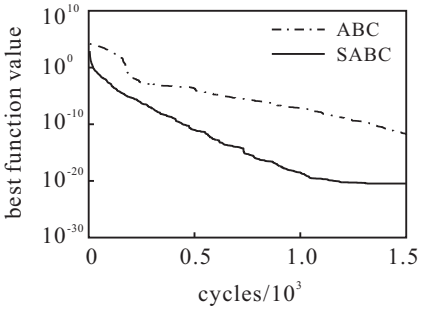


图 5 SumSquares 平均进化曲线比较

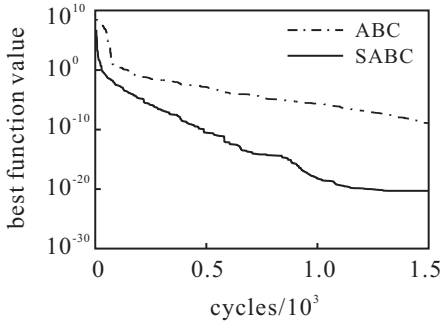


图 6　Zakharov 平均进化曲线比较

4.2　SABC 与几种改进蜂群算法的寻优结果比较

相比于其他一些改进蜂群算法, SABC 算法的寻优能力比较突出. 为了与文献 [10] 中所提到的基于 3 种不同选择机制的改进蜂群算法 (DABC, RABC, TABC) 进行比较, 选用 Sphere 函数和 Rosenbrock 函数作为测试函数, 分别定义如下:

Sphere 函数

$$f_6(x)=\sum_{i=1}^{n}x_i^2; \tag{6}$$

Rosenbrock 函数

$$f_7(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]. \tag{7}$$

SABC 算法参数设置与文献 [10] 相同, 即种群规模为 50, 函数的搜索范围分别为 $[-100,100]$ 和 $[-30,30]$, 维数为 30 时最大迭代次数为 1 000, 维数为 50 时最大迭代次数为 2 000. 各种算法分别运行 30 轮, 两个测试函数的平均进化结果分别如表 3 和表 4 所示.

表 3　Sphere 函数结果对比

算法	维数	平均值	标准差
ABC	30	1.829 458 e-09	2.639 270 e-09
DABC	30	9.798 135 e-12	1.063 763 e-11
RABC	30	1.978 102 e-12	2.033 197 e-12
TABC	30	1.514 169 e-12	1.380 414 e-12
SABC	30	1.730 03 e-20	1.350 08 e-20
ABC	50	9.564 480 e-12	1.212 776 e-11
DABC	50	2.138 416 e-14	2.318 155 e-14
RABC	50	9.066 074 e-16	7.463 074 e-16
TABC	50	4.228 150 e-15	3.779 910 e-15
SABC	50	1.473 12 e-19	2.964 70 e-19

表 4　Rosenbrock 函数结果对比

算法	维数	平均值	标准差
ABC	30	5.220 163	5.278 101
DABC	30	4.002 284	2.780 462
RABC	30	2.355 272	1.769 675
TABC	30	2.093 640	1.698 724
SABC	30	0.655 239	0.509 349
ABC	50	3.283 130	3.986 572
DABC	50	3.185 141	2.264 986
RABC	50	1.331 720	1.301 473
TABC	50	1.460 538	1.301 473
SABC	50	0.368 808	0.530 536

表 3 和表 4 的数据对比结果表明, SABC 算法的寻优效果非常好.

4.3　SABC 与 ABC 的运行时间比较

在上述 7 个测试函数的基础上比较 ABC 算法与 SABC 算法的运行时间. 两种算法在相同硬件环境下得到的平均单次运行时间如表 5 所示.

表 5　两种算法的运行时间比较　s

函数	维数	ABC	SABC
f_1	20	48.19	28.56
	40	52.16	32.65
f_2	20	39.61	29.07
	40	48.43	38.90
f_3	20	39.37	22.36
	40	47.52	22.45
f_4	20	33.91	23.73
	40	36.99	25.58
f_5	20	62.04	32.89
	40	92.70	39.61
f_6	30	11.03	9.60
	50	20.58	18.71
f_7	30	8.21	7.55
	50	16.13	16.02

由表 5 可见, 7 个测试函数的 SABC 算法的运行时间均比 ABC 算法小, 这说明 SABC 算法的计算复杂度降低了, 运算量也有所减少. 其原因是 SABC 算法的观察蜂采用全选最优食物源的选择方式, 省去了适应度计算和轮盘赌概率选择计算. 若增大分段区间数, 则可以得到更为精确的计算结果, 但算法复杂度会增加, 运行时间也会随之变长. 应用 SABC 算法解决实际问题时, 可根据具体情况设置分段区间数.

5　结　论

本文在 ABC 算法的基础上提出了一种改进蜂群算法 SABC. SABC 算法采用“分段搜索”的方式更新食物源, 提高了食物源更新的成功率. 同时, 简化了食物源的选择方式, 让观察蜂全部选择最优食物源以实现其充分优化, 省去了适应度计算和轮盘赌计算, 提高了算法的运行速度. 对经典测试函数的反复实验的结果表明, SABC 算法性能稳定, 加速收敛效果明显, 全局最优搜索能力显著提高, 运行时间大大缩短.

参考文献(References)

[1] Karaboga D, Basturk B. On the performance of artificial bee colony(ABC) algorithm[J]. Applied Soft Computing, 2008, 8(1): 687-697.

[2] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm[J]. Applied Mathematics and Computation, 2009, 214(1): 108-132.

[3] Karaboga D. An idea based on honey bee swarm for numerical optimization[R]. Kayseri: Erciyes University, 2005.

(下转第1410页)