

基于符号函数的多搜索策略人工蜂群算法

王志刚¹, 王明刚^{1,2}

(1. 南京师范大学泰州学院 数学科学与应用学院, 江苏 泰州 225300; 2. 南京师范大学 数学科学学院, 南京 210023)

摘要: 针对人工蜂群算法传统搜索策略在求解高维复杂函数时存在收敛速度较慢、容易陷入局部最优的缺陷, 提出一种基于符号函数的多搜索策略人工蜂群算法. 该算法将几种不同的搜索策略借助符号函数进行融合, 在进化过程中充分发挥各搜索策略的优势, 可以较好地平衡算法的局部搜索能力和全局搜索能力, 同时基于目标函数值进行选择寻优. 通过对16个基准函数进行的仿真实验以及与其他改进算法的比较, 表明了所提出的算法具有较快的收敛速度和较高的求解精度.

关键词: 人工蜂群算法; 符号函数; 搜索策略

中图分类号: TP301.6

文献标志码: A

Multi-search strategy of artificial bee colony algorithm based on symbolic function

WANG Zhi-gang¹, WANG Ming-gang^{1,2}

(1. School of Mathematics, Nanjing Normal University Taizhou College, Taizhou 225300, China; 2. School of Mathematical Sciences, Nanjing Normal University, Nanjing 210023, China. Correspondent: WANG Zhi-gang, E-mail: wzg19.scut@163.com)

Abstract: The traditional search strategy of the artificial bee colony(ABC) algorithm exists some disadvantages when solving complex functions with high dimensions, such as that the convergence speed is not fast enough, easy to fall into local optimum. In order to solve these issues, the multi-search strategy of the artificial bee colony(MSSABC) algorithm based on the symbolic function is presented. The new algorithm uses the symbolic function to fuse several different search strategies, makes full use of the advantages of the different search strategies during evolution to balance the local search ability and the global search ability, and selects the best solution based on the objective function value. Experiments are conducted on a set of 16 benchmark functions, and the results show that the proposed algorithm has fast convergence and high accuracy than several other ABC-based algorithms.

Keywords: artificial bee colony algorithm; symbolic function; search strategy

0 引言

人工蜂群^[1](ABC)算法是一种较为新颖的仿生优化算法. 算法具有控制参数少、鲁棒性强、易于实现等优点, 已被应用于解决多种不同的实际优化问题^[2-5], 并取得了较好的实验结果. 但在ABC算法中, 引领蜂和跟随蜂所采用的搜索策略在求解高维复杂函数时存在着过早收敛、容易陷入局部最优等问题. 为此, 专家们提出许多不同的搜索策略^[6-15]. 例如, 文献[6]通过引入Rosenbrock旋转方向的办法对引领蜂的搜索策略进行改进, 提高了算法的收敛速度; 文献

[7]受粒子群算法的启发, 在原有搜索策略的基础上融入全局最优个体的信息来提高算法的局部搜索能力; 文献[8]首先采用混沌映射和反向学习理论初始化种群, 然后引入差分变异搜索机制和一个用来平衡选择两种搜索机制的概率, 以便提高算法的全局进化性能; 文献[9]在ABC算法的搜索策略中引入一个扰动概率参数和自适应尺度因子对ABC算法进行改进; 文献[10]在差分进化思想的启发下, 提出了ABC/best算法; 文献[11]中引领蜂和跟随蜂在执行搜索策略时按照一定的选择概率从5种不同的搜

收稿日期: 2015-08-18; 修回日期: 2016-01-12.

基金项目: 国家自然科学基金项目(71503132); 江苏省高校自然科学基金项目(14KJD110005, 14KJB110017).

作者简介: 王志刚(1978—), 男, 讲师, 从事组合优化、智能优化算法等研究; 王明刚(1982—), 男, 副教授, 博士生, 从事智能优化算法、复杂系统建模与控制等研究.

索策略中选取其中一种进行搜索;文献[12]提出了一种具有自适应全局最优引导快速搜索策略的人工蜂群算法;文献[13]受分治策略的启发,提出一种基于分治策略的改进人工蜂群算法;文献[14]采用邻域搜索机制来改进人工蜂群算法的搜索策略;文献[15]提出一种基于动态搜索策略的人工蜂群算法。

在以上研究的基础上,本文提出一种基于符号函数的多搜索策略人工蜂群算法(MSSABC)。该算法在进化过程中将几种不同的搜索策略借助符号函数进行融合,并在引领蜂、跟随蜂和侦察蜂对食物源的选择过程中直接采用目标函数值来代替原算法中的适应值。为验证本文算法的性能,在16个典型的benchmark函数上进行了仿真,并与6种不同类型的ABC算法进行了对比。

1 人工蜂群算法

ABC算法中,人工蜂群包含引领蜂、跟随蜂和侦察蜂3种蜜蜂。求解优化问题时,食物源代表优化问题的一个可能解,引领蜂和跟随蜂各占蜂群总数的一半,且与食物源数目相等。用 $x_i = (x_{i1}, x_{i2}, \dots, x_{iD})$ 表示第 i ($i = 1, 2, \dots, SN$)个食物源, D 为搜索空间的维数。

引领蜂首先根据下式在食物源的附近进行邻域搜索,生成一个候选食物源

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}). \quad (1)$$

其中: v_{ij} 为生成的候选食物源, φ_{ij} 为 $[-1, 1]$ 上均匀分布的随机数, $k \in \{1, 2, \dots, SN\}$, $j \in \{1, 2, \dots, D\}$,这两个数都是随机选取的,但 $k \neq i$ 。式(1)称为ABC算法的搜索策略。

引领蜂产生候选食物源之后,通过贪婪选择较优的食物源,跟随蜂通过引领蜂传达的食物源信息按照如下概率来选择食物源:

$$P_i = \frac{\text{fit}_i}{\sum_{n=1}^{SN} \text{fit}_n}, \quad (2)$$

其中 fit_i 为第 i 个食物源的适应值。在最小化问题中, fit_i 与优化问题目标函数值 f_i 的对应关系为

$$\text{fit}_i = \begin{cases} \frac{1}{1 + f_i}, & f_i \geq 0; \\ 1 + |f_i|, & f_i < 0. \end{cases} \quad (3)$$

选择完食物源之后,跟随蜂也根据式(1)在食物源的附近进行邻域搜索,产生候选食物源,并通过贪婪选择较优的食物源。

在ABC算法中,如果一个食物源连续经过limit次循环之后仍没有得到改善,则该食物源处的引领蜂

将成为侦察蜂,并按下式随机产生一个新食物源:

$$x_{ij} = L_j + \text{rand}(U_j - L_j). \quad (4)$$

其中: x_{ij} 为新食物源的第 j 维分量, $j \in \{1, 2, \dots, D\}$,rand为 $(0, 1)$ 之间均匀分布的随机数, L_j 、 U_j 分别为第 j 维分量的下界和上界。

2 基于符号函数的多搜索策略人工蜂群算法

在ABC算法中,搜索策略在很大程度上影响着算法性能的好坏。目前,专家们提出了许多不同的搜索策略,其中常见的搜索策略有

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}), \quad (5)$$

$$v_{ij} = x_{rj} + \varphi_{ij}(x_{rj} - x_{kj}), \quad (6)$$

$$v_{ij} = x_{rj} + \varphi_{ij}(x_{ij} - x_{kj}), \quad (7)$$

$$v_{ij} = x_{\text{best},j} + \varphi_{ij}(x_{rj} - x_{kj}), \quad (8)$$

$$v_{ij} = x_{\text{best},j} + \varphi_{ij}(x_{ij} - x_{kj}). \quad (9)$$

其中: $r, k \in \{1, 2, \dots, SN\}$, $j \in \{1, 2, \dots, D\}$, $r \neq k \neq i$, x_{best} 为种群的全局最优个体。

式(5)为ABC算法的搜索策略,它具有较好的全局搜索能力,但局部搜索能力不足,导致算法收敛速度较慢;式(6)^[16]和(7)^[16]在随机选择的个体附近产生候选食物源,增加了种群的多样性,但收敛速度较慢;式(8)^[8]和(9)^[17]在种群全局最优个体附近产生候选食物源,增强了算法的局部搜索能力,加快了收敛速度,却容易陷入局部最优。众所周知,全局搜索能力和局部搜索能力对于智能优化算法而言是非常重要的,ABC算法中传统的单一搜索策略很难平衡算法的全局搜索能力和局部搜索能力。通过对上述几种搜索策略的分析,结合不同搜索策略的特点,本文提出一种基于符号函数的多搜索策略

$$v_{ij} = \omega x_{ij} + (1 - \omega)[s_1 x_{rj} + (1 - s_1)x_{\text{best},j}] + \varphi_{ij}[s_2(x_{rj} - x_{kj}) + (1 - s_2)(x_{ij} - x_{kj})]. \quad (10)$$

其中: ω 为 $[0, 1]$ 之间的实数, s_1 和 s_2 均为符号函数,定义为

$$s_1 = \begin{cases} 0, & r < 0.5; \\ 1, & \text{其他}; \end{cases} \quad (11)$$

$$s_2 = \begin{cases} 0, & r < 0.5; \\ 1, & \text{其他}. \end{cases} \quad (12)$$

式(10)采用算术交叉方式^[18],将当前个体、随机选择的个体和种群全局最优个体进行组合,结合差向量的扰动,产生一个候选食物源。与ABC算法传统的搜索策略相比,式(10)中加入的随机选择的个体有利

于增加种群的多样性和算法跳出局部最优的可能性, 而种群全局最优个体能够引导种群的搜索轨迹, 有利于加快算法的收敛速度. 基于符号函数的多搜索策略的具体过程如下:

当 $s_1 = s_2 = 1$ 时, 式 (10) 可表示为

$$v_{ij} = \omega x_{ij} + (1 - \omega)x_{rj} + \varphi_{ij}(x_{rj} - x_{kj}).$$

当前个体与随机选择的个体进行算术交叉, 外加差向量扰动, 产生候选食物源. 该候选食物源同时向当前个体与随机选择的个体靠近, 有利于增加种群多样性和提高算法的局部搜索能力. 当 $\omega = 0$ 时, 式 (10) 等价于式 (6).

当 $s_1 = 1, s_2 = 0$ 时, 式 (10) 可表示为

$$v_{ij} = \omega x_{ij} + (1 - \omega)x_{rj} + \varphi_{ij}(x_{ij} - x_{kj}).$$

此时生成的候选食物源仍然向当前个体与随机选择的个体靠近. 当 $\omega = 0$ 时, 式 (10) 等价于式 (7); 当 $\omega = 1$ 时, 式 (10) 等价于式 (5).

当 $s_1 = 0, s_2 = 1$ 时, 式 (10) 可表示为

$$v_{ij} = \omega x_{ij} + (1 - \omega)x_{\text{best},j} + \varphi_{ij}(x_{rj} - x_{kj}).$$

当前个体与种群的全局最优个体进行算术交叉, 外加差向量扰动, 产生候选食物源. 通过全局最优个体的引导, 加快算法收敛. 当 $\omega = 0$ 时, 式 (10) 等价于式 (8).

当 $s_1 = s_2 = 0$ 时, 式 (10) 可表示为

$$v_{ij} = \omega x_{ij} + (1 - \omega)x_{\text{best},j} + \varphi_{ij}(x_{ij} - x_{kj}),$$

此时生成的候选食物源仍然向当前个体与种群的全局最优个体靠近. 当 $\omega = 0$ 时, 式 (10) 等价于式 (9); 当 $\omega = 1$ 时, 式 (10) 等价于式 (5).

由以上分析可知, 式 (10) 包含了目前常见的几种搜索策略的特点, 可以弥补单一搜索策略的不足, 有利于增加种群的多样性, 较好地平衡全局搜索能力和局部搜索能力, 且基于符号函数的多搜索策略原理简单, 易于实现, 不会增加算法的复杂度.

此外, 在 ABC 算法中, 引领蜂、跟随蜂和侦察蜂在对食物源的选择过程中都是基于适应值 fit_i , 如果候选食物源的适应值优于之前食物源的适应值, 则将其取代. 但从式 (3) 可以看出, 对于函数优化问题其目标函数值大于 0 且无限接近 0 时, 对应的适应值 fit_i 不具有区分度. 为解决这个问题, 本文算法在引领蜂、跟随蜂和侦察蜂对食物源的选择过程中直接利用目标函数值来代替适应值^[19].

新算法的具体步骤如下:

1) 设置算法的各个参数;

2) 初始化种群, 计算每个引领蜂对应的食物源的函数值并记录全局最优值;

3) While 算法中止条件不满足;

4) for 引领蜂;

5) 根据式 (10) 对食物源进行更新并计算其函数值;

6) 通过贪婪选择较优的食物源;

7) 若食物源得到更新, 则 $\text{trail}(i) = 0$, 否则 $\text{trail}(i) = \text{trail}(i) + 1$;

8) end for;

9) 计算更新后的食物源的函数值, 并按式 (2) 计算选择概率;

10) for 跟随蜂;

11) if $\text{rand} < P_i$;

12) 根据式 (10) 对食物源进行更新并计算其函数值;

13) 通过贪婪选择较优的食物源;

14) 若食物源得到更新, 则 $\text{trail}(i) = 0$, 否则 $\text{trail}(i) = \text{trail}(i) + 1$;

15) end if;

16) end for;

17) if $\max(\text{trail}(i)) > \text{limit}$;

18) 根据式 (4) 产生新食物源;

19) end if;

20) end While;

21) 输出最优解及最优值.

3 仿真实验

3.1 MSSABC 算法与 ABC 算法的收敛性能比较

为验证 MSSABC 算法的性能, 选取 16 个常用的基准测试函数用于仿真实验, 并与 ABC 算法的测试结果进行比较. 实验时, 种群规模均为 $\text{SN} = 20$, $\text{limit} = \text{SN} \times D$, 最大评价次数均为 $\text{MaxFEs} = 5000D$, 通过对测试函数仿真实验发现, $\omega = 0.05$ 能使测试函数取得较优的实验结果, 因此实验中取 $\omega = 0.05$. 两种算法在每个函数上独立运行 10 次, 记录结果的最优值 (Best)、最差值 (Worst)、平均值 (Mean) 和标准差 (Std).

基准测试函数的表达式、搜索范围和理论最优值如表 1 所示, 其中 $f_1(x) \sim f_7(x)$ 和 $f_{15}(x)$ 为单峰函数; $f_8(x) \sim f_{14}(x)$ 和 $f_{16}(x)$ 为多峰函数. 表 2~表 4 分别给出了 ABC 算法和 MSSABC 算法在 $D = 30$ 、 $D = 60$ 和 $D = 100$ 时的测试结果.

由表 2 可以看出, 对于单峰函数, 两种算法都能求得 $f_5(x)$ 的理论最优值, 对于 $f_7(x)$, MSSABC 算法的求解效果不如 ABC 算法, 而对于其他单峰函数,

表 1 基准测试函数

函数名称	表达式	搜索范围	最优值
Sphere	$f_1(x) = \sum_{i=1}^D x_i^2$	$[-100, 100]$	0
Schwefel 2.22	$f_2(x) = \sum_{i=1}^D x_i + \prod_{i=1}^D x_i $	$[-10, 10]$	0
Schwefel 2.21	$f_3(x) = \max_i \{ x_i , 1 \leq i \leq D\}$	$[-100, 100]$	0
SumSquares	$f_4(x) = \sum_{i=1}^D ix_i^2$	$[-10, 10]$	0
Step	$f_5(x) = \sum_{i=1}^D (\lfloor x_i + 0.5 \rfloor)^2$	$[-100, 100]$	0
Quartic	$f_6(x) = \sum_{i=1}^D ix_i^4 + \text{random}[0, 1)$	$[-1.28, 1.28]$	0
Rosenbrock	$f_7(x) = \sum_{i=1}^{D-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$	$[-10, 10]$	0
Rastrigin	$f_8(x) = \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i) + 10)$	$[-5.12, 5.12]$	0
Non-continuous rastrigin	$f_9(x) = \sum_{i=1}^D (y_i^2 - 10 \cos(2\pi y_i) + 10)y_i = \begin{cases} x_i, & x_i < \frac{1}{2}; \\ \frac{\text{round}(2x_i)}{2}, & x_i \geq \frac{1}{2} \end{cases}$	$[-5.12, 5.12]$	0
Griewank	$f_{10}(x) = \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$	0
Ackley	$f_{11}(x) = 20 + e - 20e^{-0.2} \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2} - e^{1/D} \sum_{i=1}^D \cos(2\pi x_i)$	$[-32, 32]$	0
Schaffer	$f_{12}(x) = 0.5 + \frac{\sin^2\left(\sqrt{\sum_{i=1}^D x_i^2}\right) - 0.5}{\left(1 + 0.001 \sum_{i=1}^D x_i^2\right)^2}$	$[-100, 100]$	0
Penalized1	$f_{13}(x) = \frac{\pi}{D} \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_D - 1)^2 \right\} + \sum_{i=1}^D u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{1}{4}(x_i + 1), u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a; \\ 0, & -a \leq x_i \leq a; \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	$[-50, 50]$	0
Penalized2	$f_{14}(x) = \frac{1}{10} \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{D-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + (x_D - 1)^2 [1 + \sin^2(2\pi x_D)] \right\} + \sum_{i=1}^D u(x_i, 5, 100, 4)$	$[-50, 50]$	0
Shifted sphere	$f_{15}(x) = \sum_{i=1}^D z_i^2, z = x - o$	$[-100, 100]$	0
Shifted rastrigin	$f_{16}(x) = \sum_{i=1}^D (z_i^2 - 10 \cos(2\pi z_i) + 10), z = x - o$	$[-5.12, 5.12]$	0

MSSABC 算法在解的精度和稳定性上都优于 ABC 算法; 对于多峰函数, 两种算法都能得到 $f_8(x)$ 、 $f_9(x)$ 和 $f_{16}(x)$ 的理论最优值, 对于其他的复杂多峰函数, MSSABC 算法的求解结果要优于 ABC 算法. 为了直

观地反映 MSSABC 算法的收敛性能, 图 1 给出了两种算法对于部分测试函数的收敛曲线. 可以看出, MSSABC 算法的收敛曲线下降速度较快, 能够收敛到较高精度的解.

表 2 ABC 和 MSSABC 在函数维数为 30 时的测试结果

函数	算法	Best	Worst	Mean	Std	函数	算法	Best	Worst	Mean	Std
f_1	ABC	3.84e-16	6.52e-16	4.96e-16	7.08e-17	f_9	ABC	0	0	0	0
	MSSABC	8.15e-83	1.80e-76	1.84e-77	5.40e-77		MSSABC	0	0	0	0
f_2	ABC	1.20e-15	1.61e-15	1.37e-15	1.22e-16	f_{10}	ABC	0	3.98e-09	3.98e-10	1.19e-09
	MSSABC	7.62e-53	3.21e-46	4.39e-47	9.51e-47		MSSABC	0	3.89e-15	4.22e-16	1.16e-15
f_3	ABC	4.43e-01	1.47e+00	8.79e-01	3.08e-01	f_{11}	ABC	2.55e-14	3.97e-14	3.40e-14	4.55e-15
	MSSABC	1.14e-02	3.31e-02	2.19e-02	7.60e-03		MSSABC	2.55e-14	2.90e-14	2.79e-14	1.63e-15
f_4	ABC	2.77e-16	5.50e-16	4.70e-16	9.26e-17	f_{12}	ABC	2.73e-01	3.96e-01	3.22e-01	3.89e-02
	MSSABC	1.98e-82	9.30e-79	2.90e-79	3.42e-79		MSSABC	1.27e-01	3.12e-01	2.29e-01	5.31e-02
f_5	ABC	0	0	0	0	f_{13}	ABC	3.28e-16	5.36e-16	4.71e-16	7.27e-17
	MSSABC	0	0	0	0		MSSABC	1.57e-32	1.57e-32	1.57e-32	2.74e-48
f_6	ABC	3.55e-02	7.35e-02	4.96e-02	1.03e-02	f_{14}	ABC	3.15e-16	5.51e-16	4.84e-16	6.80e-17
	MSSABC	1.05e-02	2.51e-02	1.55e-02	3.76e-03		MSSABC	1.35e-32	1.35e-32	1.35e-32	2.74e-48
f_7	ABC	1.42e-04	1.31e-01	3.84e-02	4.20e-02	f_{15}	ABC	3.22e-16	6.84e-16	4.95e-16	9.01e-17
	MSSABC	5.57e-04	7.50e-01	1.41e-01	2.18e-01		MSSABC	0	0	0	0
f_8	ABC	0	0	0	0	f_{16}	ABC	0	0	0	0
	MSSABC	0	0	0	0		MSSABC	0	0	0	0

表 3 ABC 和 MSSABC 在函数维数为 60 时的测试结果

函数	算法	Best	Worst	Mean	Std	函数	算法	Best	Worst	Mean	Std
f_1	ABC	9.76e-16	1.63e-15	1.25e-15	1.88e-16	f_9	ABC	0	0	0	0
	MSSABC	1.45e-80	2.17e-76	4.63e-77	7.97e-77		MSSABC	0	0	0	0
f_2	ABC	2.74e-15	3.41e-15	3.02e-15	1.87e-16	f_{10}	ABC	1.11e-16	9.99e-16	3.44e-16	2.78e-16
	MSSABC	1.25e-50	1.65e-47	3.30e-48	5.22e-48		MSSABC	0	0	0	0
f_3	ABC	8.36e+00	1.71e+01	1.25e+01	2.68e+00	f_{11}	ABC	7.16e-14	8.59e-14	7.91e-14	5.14e-15
	MSSABC	1.24e+00	1.94e+00	1.59e+00	2.25e-01		MSSABC	6.10e-14	7.16e-14	6.67e-14	2.84e-15
f_4	ABC	8.86e-16	1.41e-15	1.20e-15	1.51e-16	f_{12}	ABC	4.59e-01	4.83e-01	4.76e-01	7.16e-03
	MSSABC	3.80e-82	1.08e-77	2.51e-78	3.70e-78		MSSABC	4.15e-01	4.76e-01	4.39e-01	1.75e-02
f_5	ABC	0	0	0	0	f_{13}	ABC	9.04e-16	1.41e-15	1.20e-15	1.71e-16
	MSSABC	0	0	0	0		MSSABC	7.85e-33	7.85e-33	7.85e-33	1.37e-48
f_6	ABC	6.56e-02	1.13e-01	9.72e-02	1.34e-02	f_{14}	ABC	9.50e-16	1.42e-15	1.17e-15	1.58e-16
	MSSABC	2.85e-02	5.42e-02	4.15e-02	7.36e-03		MSSABC	1.35e-32	1.35e-32	1.35e-32	2.74e-48
f_7	ABC	4.58e-03	2.04e-01	7.09e-02	7.17e-02	f_{15}	ABC	7.62e-16	1.44e-15	1.23e-15	1.85e-16
	MSSABC	1.34e-03	1.10e+00	2.48e-01	3.27e-01		MSSABC	0	0	0	0
f_8	ABC	0	0	0	0	f_{16}	ABC	0	0	0	0
	MSSABC	0	0	0	0		MSSABC	0	0	0	0

表 4 ABC 和 MSSABC 在函数维数为 100 时的测试结果

函数	算法	Best	Worst	Mean	Std	函数	算法	Best	Worst	Mean	Std
f_1	ABC	1.65e-15	2.54e-15	2.21e-15	2.68e-16	f_9	ABC	0	0	0	0
	MSSABC	1.36e-80	2.69e-75	3.552e-76	7.86e-76		MSSABC	0	0	0	0
f_2	ABC	4.49e-15	5.61e-15	4.99e-15	3.42e-16	f_{10}	ABC	1.11e-16	9.99e-16	4.88e-16	3.55e-16
	MSSABC	5.12e-49	9.17e-45	1.55e-45	3.00e-45		MSSABC	0	0	0	0
f_3	ABC	2.35e+01	3.29e+01	2.82e+01	2.87e+00	f_{11}	ABC	1.28e-13	1.50e-13	1.40e-13	8.82e-15
	MSSABC	7.41e+00	8.92e+00	7.83e+00	6.46e-01		MSSABC	1.07e-13	1.21e-13	1.16e-13	5.07e-15
f_4	ABC	1.66e-15	2.53e-15	2.23e-15	2.57e-16	f_{12}	ABC	4.96e-01	4.98e-01	4.97e-01	7.54e-04
	MSSABC	1.45e-81	4.79e-77	1.50e-77	1.67e-77		MSSABC	4.87e-01	4.94e-01	4.91e-01	2.40e-03
f_5	ABC	0	0	0	0	f_{13}	ABC	1.87e-15	2.52e-15	2.18e-15	1.76e-16
	MSSABC	0	0	0	0		MSSABC	4.71e-33	4.71e-33	4.71e-33	0
f_6	ABC	1.27e-01	1.91e-01	1.63e-01	1.82e-02	f_{14}	ABC	1.62e-15	2.54e-15	2.20e-15	2.83e-16
	MSSABC	4.92e-02	6.94e-02	6.04e-02	7.38e-03		MSSABC	1.35e-32	1.35e-32	1.35e-32	2.74e-48
f_7	ABC	5.10e-03	1.86e+00	2.89e-01	5.29e-01	f_{15}	ABC	1.66e-15	2.72e-15	2.27e-15	3.10e-16
	MSSABC	9.85e-04	7.20e+01	7.57e+00	2.15e+01		MSSABC	0	0	0	0
f_8	ABC	0	0	0	0	f_{16}	ABC	0	0	0	0
	MSSABC	0	0	0	0		MSSABC	0	0	0	0

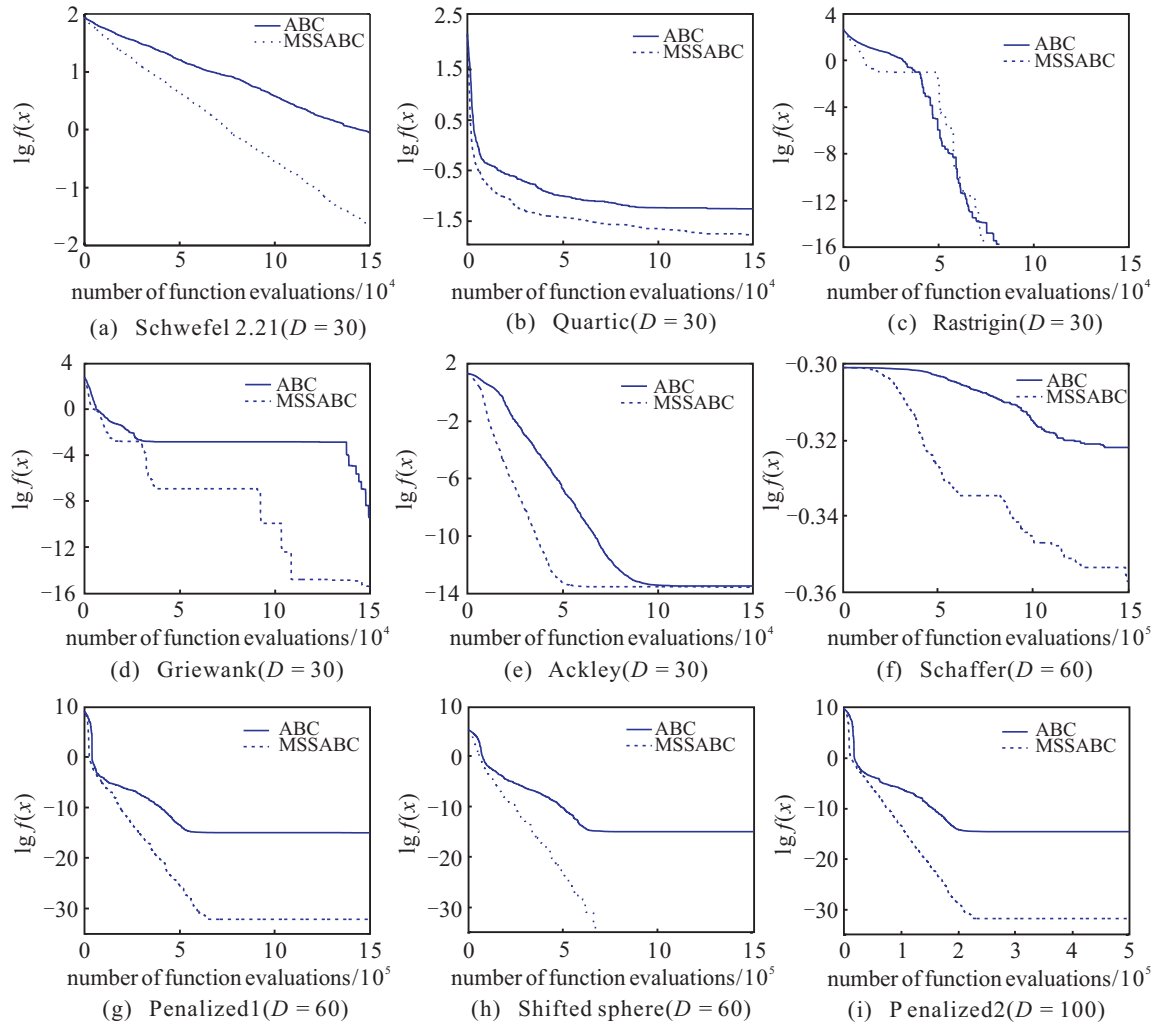


图 1 不同基准测试函数的收敛曲线

表 5 6 种算法在函数维数为 30 时的测试结果

函数	GABC		MABC		ABCBest1		ABCBest2		ABCVSS		MSSABC	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
f_1	4.62e-16	7.12e-17	9.43e-32	6.67e-32	3.11e-47	3.44e-47	5.96e-35	3.61e-35	1.53e-81	8.37e-81	1.84e-77	5.40e-77
f_2	1.35e-15	1.36e-16	2.40e-17	9.02e-18	2.10e-25	9.08e-26	1.36e-18	4.27e-19	7.89e-43	4.32e-42	4.39e-47	9.51e-47
f_3	2.18e-01	4.01e-02	1.02e+01	1.49e+00	2.18e+00	3.27e-01	3.55e+00	4.79e-01	4.08e-02	2.20e-02	2.19e-02	7.60e-03
f_4	4.55e-16	7.00e-17	2.10e-32	1.56e-32	6.50e-48	6.04e-48	5.55e-36	3.36e-36	3.19e-89	1.48e-88	2.90e-79	3.42e-79
f_5	0	0	0	0	0	0	0	0	0	0	0	0
f_6	2.03e-02	5.74e-03	3.71e-02	8.53e-03	2.06e-02	4.75e-03	2.53e-02	4.67e-03	1.81e-02	5.27e-03	1.55e-02	3.76e-03
f_7	3.21e-01	8.21e-01	6.11e-01	4.55e-01	1.49e+01	2.87e+01	5.45e+00	8.40e+00	3.87e-01	1.54e+00	1.41e-01	2.18e-01
f_8	0	0	0	0	0	0	0	0	0	0	0	0
f_9	0	0	0	0	0	0	0	0	0	0	0	0
f_{10}	3.70e-17	5.32e-17	0	0	0	0	1.81e-08	6.29e-08	0	0	4.22e-16	1.16e-15
f_{11}	3.20e-14	3.36e-15	4.13e-14	2.17e-15	3.01e-14	2.91e-15	3.07e-14	3.43e-15	2.45e-14	4.00e-15	2.79e-14	1.63e-15
f_{12}	2.66e-01	4.39e-02	2.95e-01	3.17e-02	2.39e-01	6.13e-02	2.81e-01	3.92e-02	2.84e-01	5.69e-02	2.29e-01	5.31e-02
f_{13}	4.12e-16	8.36e-17	1.90e-32	3.70e-33	1.57e-32	5.57e-48	1.57e-32	5.57e-48	1.57e-32	5.57e-48	1.57e-32	2.74e-48
f_{14}	4.01e-16	8.19e-17	2.23e-31	1.46e-31	1.35e-32	5.57e-48	1.35e-32	5.57e-48	1.35e-32	5.57e-48	1.35e-32	2.74e-48
f_{15}	4.38e-16	8.43e-17	0	0	0	0	0	0	0	0	0	0
f_{16}	0	0	0	0	0	0	0	0	0	0	0	0

由表 3 和表 4 可以看出, 当 $D=60$ 和 $D=100$ 时, MSSABC 算法也取得了与 $D=30$ 时一样好的优化结果, 即对于 16 个测试函数, 除 $f_7(x)$ 外, MSSABC 算法

的求解结果要优于 ABC 算法, 或与 ABC 算法一样, 都能得到理论最优值 ($f_5(x)$ 、 $f_8(x)$ 、 $f_9(x)$ 和 $f_{16}(x)$).

3.2 本文算法与几种改进的 ABC 算法比较

为测试 MSSABC 算法的性能, 将其与 GABC^[7]、MABC^[8]、ABCBest1^[10]、ABCBest2^[10]、ABCVSS^[11] 等较新的改进算法在 $D=30$ 、 $D=60$ 和 $D=100$ 时进行比较, 算法的最大评价次数均为 $\text{MaxFEs}=5\,000D$, 对于 MSSABC 算法, $\text{SN}=20$, $\text{limit}=\text{SN}\times D$, $\omega=0.05$;

对于其余 5 种算法, 实验数据直接取自文献 [11]. 表 5~表 7 给出了 6 种算法的测试结果. 由表 5~表 7 可以看出, 在 16 个基准测试函数中, 对于 $f_5(x)$ 、 $f_8(x)$ 、 $f_9(x)$ 和 $f_{16}(x)$, 6 种算法都能求得理论最优值, 对于其他函数, MSSABC 算法在解的精度和稳定性两方面明显优于 GABC、MABC、ABCBest1、ABCBest2, 在大多数函数中优于最新提出的 ABCVSS.

表 6 6 种算法在函数维数为 60 时的测试结果

函数	GABC		MABC		ABCBest1		ABCBest2		ABCVSS		MSSABC	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
f_1	1.06e-15	1.21e-16	6.03e-29	4.31e-29	3.92e-44	2.64e-44	4.82e-33	2.59e-33	1.09e-83	5.01e-83	4.63e-77	7.97e-77
f_2	2.96e-15	1.85e-16	6.96e-16	1.20e-16	8.48e-24	2.31e-24	1.58e-17	3.32e-18	1.47e-45	7.00e-45	3.30e-48	5.22e-48
f_3	4.47e+00	6.09e-01	3.77e+01	3.14e+00	2.10e+01	1.68e+00	2.40e+01	2.16e+00	1.68e+00	4.05e-01	1.59e+00	2.25e-01
f_4	1.04e-15	1.27e-16	1.39e-29	8.84e-30	2.06e-44	1.83e-44	9.10e-34	3.87e-34	8.17e-86	4.47e-85	2.51e-78	3.70e-78
f_5	0	0	0	0	0	0	0	0	0	0	0	0
f_6	5.43e-02	7.03e-03	1.14e-01	1.16e-02	6.11e-02	8.89e-03	6.79e-02	9.38e-03	4.35e-02	7.69e-03	4.15e-02	7.36e-03
f_7	3.30e+00	1.28e+01	1.51e+00	1.34e+00	5.04e+01	5.46e+01	5.10e+01	3.77e+01	5.27e-01	1.18e+00	2.48e-01	3.27e-01
f_8	0	0	0	0	0	0	0	0	0	0	0	0
f_9	0	0	0	0	0	0	0	0	0	0	0	0
f_{10}	2.47e-04	1.35e-03	0	0	0	0	3.96e-09	2.04e-08	0	0	0	0
f_{11}	7.31e-14	5.57e-15	1.37e-13	1.24e-14	6.93e-14	5.00e-15	7.47e-14	4.12e-15	5.93e-14	7.65e-15	6.67e-14	2.84e-15
f_{12}	4.62e-01	1.79e-02	4.84e-01	3.62e-03	4.61e-01	1.15e-02	4.68e-01	9.17e-03	4.72e-01	1.42e-02	4.39e-01	1.75e-02
f_{13}	1.05e-15	1.21e-16	6.19e-31	3.62e-31	7.85e-33	2.78e-48	7.85e-33	2.78e-48	7.85e-33	2.78e-48	7.85e-33	1.37e-48
f_{14}	1.01e-15	1.28e-16	3.80e-29	1.87e-29	1.35e-32	5.57e-48	1.35e-32	5.57e-48	1.35e-32	5.57e-48	1.35e-32	2.74e-48
f_{15}	1.01e-15	1.23e-16	5.61e-29	4.18e-29	0	0	0	0	0	0	0	0
f_{16}	0	0	0	0	0	0	0	0	0	0	0	0

表 7 6 种算法在函数维数为 100 时的测试结果

函数	GABC		MABC		ABCBest1		ABCBest2		ABCVSS		MSSABC	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
f_1	1.84e-15	1.72e-16	1.43e-27	8.12e-28	1.54e-42	8.93e-43	5.09e-32	2.03e-32	1.01e-83	5.52e-83	3.552e-76	7.86e-76
f_2	5.17e-15	2.24e-16	4.41e-15	1.50e-15	6.27e-23	1.09e-23	7.21e-17	1.36e-17	3.94e-40	2.12e-39	1.55e-45	3.00e-45
f_3	1.59e+01	1.55e+00	5.98e+01	1.60e+00	4.72e+01	2.29e+00	5.06e+01	2.67e+00	7.91e+00	1.32e+00	7.83e+00	6.46e-01
f_4	1.85e-15	2.00e-16	4.46e-28	2.08e-28	8.94e-43	7.34e-43	2.15e-32	1.24e-32	5.31e-85	2.68e-84	1.50e-77	1.67e-77
f_5	0	0	0	0	0	0	0	0	0	0	0	0
f_6	9.47e-02	1.26e-02	2.31e-01	2.79e-02	1.30e-01	1.12e-02	1.42e-01	1.58e-02	7.87e-02	1.16e-02	6.04e-02	7.38e-03
f_7	2.40e+01	3.39e+01	1.98e+00	1.30e+00	5.81e+01	6.80e+01	1.18e+02	5.76e+01	5.14e-01	1.05e+00	7.57e+00	2.15e+01
f_8	0	0	0	0	0	0	0	0	0	0	0	0
f_9	0	0	0	0	0	0	0	0	0	0	0	0
f_{10}	1.26e-16	1.56e-16	0	0	0	0	2.94e-10	1.16e-09	0	0	0	0
f_{11}	1.32e-13	9.74e-15	3.56e-13	2.29e-14	1.27e-13	7.15e-15	1.39e-13	5.75e-15	1.11e-13	9.93e-15	1.16e-13	5.07e-15
f_{12}	4.95e-01	1.87e-03	4.99e-01	1.75e-04	4.96e-01	8.28e-04	4.97e-01	6.58e-04	4.98e-01	8.15e-04	4.91e-01	2.40e-03
f_{13}	1.90e-15	1.95e-16	1.89e-30	8.42e-31	4.71e-33	1.39e-48	4.71e-33	1.39e-48	4.71e-33	1.39e-48	4.71e-33	0
f_{14}	1.85e-15	1.68e-16	1.81e-28	6.44e-29	1.35e-32	5.57e-48	2.18e-32	6.62e-33	1.35e-32	5.57e-48	1.35e-32	2.74e-48
f_{15}	1.90e-15	1.96e-16	1.44e-27	9.16e-28	0	0	0	0	0	0	0	0
f_{16}	0	0	0	0	0	0	0	0	0	0	0	0

从上述实验结果可以看出, MSSABC 算法在计算精度上有了明显提高, 能有效克服 ABC 算法收敛速度慢和易陷入局部最优的缺陷, 并且随着目标函数维数的增加, 仍能保持较好的有效性和鲁棒性.

4 结 论

本文针对 ABC 算法搜索策略容易导致算法陷入早熟、搜索效率较低的缺点, 提出一种基于符号函数的多搜索策略人工蜂群算法. 该算法在进化过程中借

助符号函数将几种常见的搜索策略进行融合,实现优势互补,并基于目标函数值进行选择操作. 16个基准测试函数的仿真实验结果表明,本文算法在优化性能和鲁棒性等方面较基本人工蜂群算法及改进的人工蜂群算法有了较大的改善,但其也有局限性,例如对于很难求解的 Rosenbrock 函数的求解效果并不理想. 如何使算法能够在更多测试函数上表现更好的性能以及将所提出的算法应用到约束优化、多目标优化等领域将是下一步的研究工作.

参考文献(References)

- [1] Karaboga D. An idea based on honey bee swarm for numerical optimization[R]. Kayseri: Erciyes University, 2005.
- [2] Karaboga N. A new design method based on artificial bee colony algorithm for digital IIR filters[J]. J of the Franklin Institute, 2009, 346(4): 328-348.
- [3] Singh A. An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem[J]. Applied Soft Computing, 2009, 9(2): 625-631.
- [4] Tasgetiren M F, Pan Q K, Suganthan P N, et al. A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops[J]. Information Sciences, 2011, 181(16): 3459-3475.
- [5] Szeto W, Wu Y, Ho S C. An artificial bee colony algorithms for the capacitated vehicle routing problem[J]. European J of Operational Research, 2011, 215(1): 126-135.
- [6] Kang F, Li J L, Ma Z Y. Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions[J]. Information Sciences, 2011, 181(16): 3508-3531.
- [7] Zhu G P, Kwong S. Gbest-guided artificial bee colony algorithm for numerical function optimization[J]. Applied Mathematics and Computation, 2010, 217(7): 3166-3173.
- [8] Gao W F, Liu S Y. A modified artificial bee colony algorithm[J]. Computer & Operations Research, 2012, 39(3): 687-697.
- [9] Akay B, Karaboga D. A modified artificial bee colony algorithm for real-parameter optimization[J]. Information Sciences, 2012, 192(1): 120-142.
- [10] Gao W F, Liu S Y, Huang L L. A global best artificial bee colony algorithm for global optimization[J]. J of Computational and Applied Mathematics, 2012, 236(11): 2741-2753.
- [11] Kiran M S, Hakli H, Gunduz M, et al. Artificial bee colony algorithm with variable search strategy for continuous optimization[J]. Information Sciences, 2015, 300(1): 140-157.
- [12] 赵辉, 李牧东, 翁兴伟. 具有自适应全局最优引导快速搜索策略的人工蜂群算法[J]. 控制与决策, 2014, 29(11): 2041-2047.
(Zhao H, Li M D, Weng X W. Improved artificial bee colony algorithm with self-adaptive global best-guided quick searching strategy[J]. Control and Decision, 2014, 29(11): 2041-2047.)
- [13] 李田来, 刘方爱, 王新华. 基于分治策略的改进人工蜂群算法[J]. 控制与决策, 2015, 30(2): 316-320.
(Li T L, Liu F A, Wang X H. Modified artificial bee colony algorithm based on divide-and-conquer strategy[J]. Control and Decision, 2015, 30(2): 316-320.)
- [14] 周新宇, 吴志健, 邓长寿, 等. 一种邻域搜索的人工蜂群算法[J]. 中南大学学报: 自然科学版, 2015, 46(2): 534-546.
(Zhou X Y, Wu Z J, Deng C S, et al. Neighborhood search-based artificial bee colony algorithm[J]. J of Central South University: Science and Technology, 2015, 46(2): 534-546.)
- [15] 王志刚. 基于动态搜索策略的人工蜂群算法[J]. 计算机工程与科学, 2015, 37(4): 734-739.
(Wang Z G. A novel artificial bee colony algorithm based on dynamic search strategy[J]. Computer Engineering & Science, 2015, 37(4): 734-739.)
- [16] Gao W F, Liu S Y, Huang L L. A novel artificial bee colony algorithm based on modified search equation and orthogonal learning [J]. IEEE Trans on Cybernetics, 2013, 43(3): 1011-1024.
- [17] Gao W F, Liu S Y. Improved artificial bee colony algorithm for global optimization[J]. Information Processing Letters, 2011, 111(17): 687-697.
- [18] 孔祥勇, 高立群, 欧阳海滨, 等. 双向随机多策略变异的自适应差分进化算法[J]. 计算机集成制造系统, 2014, 20(8): 1948-1958.
(Kong X Y, Gao L Q, Ouyang H B, et al. Adaptive differential evolution algorithm with bidirectional randomly multi-mutation strategy[J]. Computer Integrated Manufacturing Systems, 2014, 20(8): 1948-1958.)
- [19] Banharnsakun A, Achalakul T, Sirinaovakul B. The best-so-far selection in artificial bee colony algorithm[J]. Applied Soft Computing, 2011, 11(2): 2888-2901.

(责任编辑: 孙艺红)