

加权中心人工蜂群算法

孙 辉^{1,2,3†}, 谢海华¹, 赵 嘉^{1,2,3}, 邓志诚¹

(1. 南昌工程学院 信息工程学院, 南昌 330000; 2. 鄱阳湖流域水工程安全与资源高效利用国家地方联合工程实验室, 南昌 330000; 3. 江西省水信息协同感知与智能处理重点实验室, 南昌 330000)

摘 要: 针对人工蜂群算法收敛速度慢、局部搜索能力差等缺点, 提出一种新的改进人工蜂群算法. 新算法依据蜜源适应值进行排序, 将排序结果作为权值, 构造一个虚拟蜜源, 即加权中心. 若加权中心优于当前最优解, 则取代当前最优解, 以便得到更好的当前最优解. 在加权中心的基础上, 增加全维搜索策略, 以改善算法的局部搜索能力. 两种策略的应用能够加快算法的收敛速度, 增强局部搜索能力. 在经典的 22 个基准测试函数上, 对新算法的有效性进行实验仿真分析, 实验结果表明, 所提出算法在求解精度和速度上均有显著提高, 在给定等同的时间下远高于其他算法.

关键词: 人工蜂群算法; 加权中心; 虚拟蜜源; 收敛速度; 当前最优解; 局部搜索

中图分类号: TP301.6

文献标志码: A

Artificial bee algorithm with weighted center

SUN Hui^{1,2,3†}, XIE Hai-hua¹, ZHAO Jia^{1,2,3}, DENG Zhi-cheng¹

(1. School of Information Engineering, Nanchang Institute of Technology, Nanchang 330000, China; 2. National-Local Engineering Laboratory of Water Engineering Safety and Effective Utilization of Resources in Poyang Lake Area, Nanchang 330000, China; 3. Jiangxi Province Key Laboratory of Water Information Cooperative Sensing and Intelligent Processing, Nanchang 330000, China)

Abstract: In view of the disadvantages of slow convergence speed and weak local search ability of the artificial bee algorithm, this paper proposes a new improved artificial bee algorithm. The honey source is sorted according to the fitness value, and then the sorting result is used as the weight value to construct a virtual honey source, namely the weighted center. If the weighted center is superior to the current optimal solution, the current optimal solution is replaced to obtain a better current optimal solution. On the basis of the weighted center, the full-dimension search strategy is added to improve the local search capability of the algorithm. The application of two strategies speeds up the convergence speed of the algorithm and enhances the local search capability. The validity of the new algorithm on the classical 22 benchmark test function is analyzed. The experimental results show that the proposed algorithm can significantly improve the accuracy and speed. Under a given equivalent time, the solving accuracy and the convergence speed of the proposed algorithm are much higher than that the comparison algorithm.

Keywords: artificial bee algorithm; weighted center; virtual honey source; convergence speed; current optimal solution; local search

0 引 言

2005 年, Karaboga^[1] 为优化代数问题提出了人工蜂群 (Artificial bee colony, ABC) 算法. 类似 ABC 算法的有萤火虫算法 (Firefly algorithm, FA)^[2]、人工鱼群算法 (Artificial fish swarm algorithm, AFSA)^[3] 以及粒子群 (Particle swarm optimization, PSO) 算法^[4] 等. 对比这些类似算法, ABC 算法具有控制参数少、操作简单和易于实现等优点, 广泛应用于路径规划^[5] 和图像

分割^[6] 等研究领域, 且算法对问题的维数不太敏感^[7], 适合对高维问题进行求解. 然而, ABC 算法存在收敛速度较慢、易陷入局部最优以及在后期种群多样性减少, 导致在寻找最优蜜源的问题上停滞不前等缺点.

针对上述缺点, 国内外学者提出了多种改进的 ABC 算法. 王志刚等^[8] 通过符号函数对多个搜索策略进行融合, 充分发挥各个搜索策略的优势, 加快了

收稿日期: 2018-01-28; 修回日期: 2018-03-29.

基金项目: 国家自然科学基金项目 (61261039, 51669014); 江西省教育厅落地计划项目 (KJLD13096).

责任编辑: 林崇.

† 通讯作者. E-mail: sun_hui2006@163.com.

ABC算法的收敛速度,提高了算法的局部搜索能力. Karaboga等^[9]限定了蜂群个体的更新邻域范围,在限定范围内选取最优蜜源进行更新,该策略牺牲了一定的全局搜索能力以提高ABC算法的局部搜索能力. Kiran等^[10]为雇佣蜂、观察蜂的搜索策略引入方向信息,使得雇佣蜂、观察蜂在产生一个新的蜜源的过程中不再盲目搜索. Gao等^[11]受差分进化算法(Differential evolution, DE)的影响更改了蜜源搜索策略,通过引入控制参数,平衡了全局搜索能力和局部搜索能力. Banharnsakun等^[12]更改了3类蜂的搜索策略,增强了算法局部搜索能力,避免侦查蜂浪费先前的搜索资源,提高了搜索效率. Bao等^[13]提出了3种计算适应度公式,改善了蜂群后期种群多样性减小问题. Wang等^[7]基于不同问题的不同搜索策略思想,针对不同目标函数引入不同的搜索策略进行求优. 目前的研究表明,ABC算法的搜索策略是导致局部搜索能力差的主要原因,改善算法的搜索策略可提高算法的性能.

鉴于此,本文提出一种加权中心人工蜂群算法(Artificial bee colony with weighted center, WCABC). 新算法依据蜜源适应值进行排序,将排序结果作为权值,构造一个虚拟蜜源,即加权中心. 若加权中心优于当前最优解,则取代当前最优解,以便得到更好的当前最优解. 在加权中心的基础上,增加全维搜索策略,以改善算法的局部搜索能力. 两种策略的应用能够加快算法的收敛速度,增强局部搜索能力. 在经典的22个基准测试函数上,对新算法的有效性进行实验仿真分析,实验结果表明,所提出算法在求解精度和速度上均有显著提高,在给定的同等时间下远高于其他算法.

1 标准ABC算法

ABC算法通过模拟蜜蜂的觅食行为来求解优化问题,由3类蜜蜂构成,即雇佣蜂、观察蜂和侦查蜂^[1]. 雇佣蜂的数量是蜂群数量的一半且等于观察蜂的数量. 当某蜜源枯竭时,相应的雇佣蜂转换成侦查蜂. 算法中,被蜂群采集的蜜源对应实际优化问题的可行解,每个蜜源对应一只雇佣蜂^[4]. 雇佣蜂发现蜜源后,通过“8字舞”分享蜜源信息给观察蜂,观察蜂通过轮盘赌的方式,以一定的概率选取要跟随的雇佣蜂,蜜源的质量(适应值)越好,被选中的概率越大.

1.1 蜜源初始化

在蜜源初始化阶段,蜜源由下式在搜索区域内随机产生:

$$x_{i,j} = x_{i,j}^{\min} + \text{rand}(0, 1)(x_{i,j}^{\max} - x_{i,j}^{\min}). \quad (1)$$

其中: $i = 1, 2, \dots, \text{SN}$, $j = 1, 2, \dots, D$, SN为种群的大小(可行解的数量), D 为蜜源的维度大小, $x_{i,j}^{\min}$ 为第*i*个蜜源第*j*维的下边界, $x_{i,j}^{\max}$ 为第*i*个蜜源第*j*维的上边界, $x_{i,j}$ 为初始化后的蜜源, $\text{rand}(0, 1)$ 为属于 $[0, 1]$ 的随机数.

初始化完成后,评估初始化蜜源的适应度. 每一次评估蜜源,相应地增加蜜源评估次数. 蜜源的适应度由下式计算:

$$\text{fit}_i = \begin{cases} \frac{1}{1 + f_i}, & f_i \geq 0; \\ 1 + \text{abs}(f_i), & f_i < 0. \end{cases} \quad (2)$$

其中: f_i 为蜜源*i*的适应值, fit_i 为蜜源*i*的适应度.

1.2 雇佣蜂阶段

各雇佣蜂在对应蜜源的邻域内,由下式产生候选蜜源 $v_{i,j}$:

$$v_{i,j} = x_{i,j} + \text{rand}(-1, 1)(x_{i,j} - x_{k,j}), \quad (3)$$

其中 $x_{k,j}$ 为蜜源 $x_{i,j}$ 的邻域蜜源,且 $i \neq k$.

随后通过贪婪选择机制,在 $v_{i,j}$ 与 $x_{i,j}$ 之间选择一个质量较优的蜜源. 若更新失败,则对应蜜源限制次数加1,否则置为0.

1.3 观察蜂阶段

雇佣蜂通过“8字舞”分享蜜源信息给观察蜂后,观察蜂通过轮盘赌策略,选取要跟随的雇佣蜂. 在轮盘赌策略中,蜜源质量越好,相应的雇佣蜂被跟随的概率越大. 选择概率计算为

$$p_i = \frac{\text{fit}(x_i)}{\sum_{i=1}^{\text{SN}} \text{fit}(x_i)}. \quad (4)$$

观察蜂确定所跟随的雇佣蜂后,在相应蜜源邻域内由式(3)产生候选蜜源 $v_{i,j}$. 同样由贪婪选择机制,在 $v_{i,j}$ 与 $x_{i,j}$ 间选择一个质量较优的蜜源. 若更新失败则蜜源限制次数加1,否则置为0.

1.4 侦查蜂阶段

若蜜源在限制次数内持续更新失败,则相应的雇佣蜂转换成侦查蜂,在搜索区域内由式(1)随机产生一个新蜜源取代该蜜源,对应的蜜源限制次数置为0.

2 WCABC算法

受Gbest^[15](当前最优蜜源)的启发,研究者提出了大量改进的ABC算法,如A modified artificial bee colony(MABC) algorithm^[11]、异维学习人工蜂群(DDLABC)算法^[16]、The best-so-far selection in artificial bee colony(best-so-far ABC) algorithm^[12]等. Gbest在蜂群搜索迭代过程中为全局最优值概率

较小,通常为局部最优值,且Gbest进化的成功率低于其他蜜源.又因为ABC算法轮盘赌策略的缘故,质量较好的蜜源将得到更多的计算资源.导致Gbest计算资源而进化的成功率较低,从而Gbest的限制次数增加,使得对应的雇佣蜂过早进入侦查蜂阶段.针对此问题,本文将所有蜜源根据适应值优劣赋予不同的权重值,构建一个新的蜜源,即加权中心.如果加权中心优于Gbest,则用加权中心取代Gbest,并用其对观察蜂进行引导更新,否则不取代.

2.1 加权中心的构建

蜜源代表目标函数在搜索域中的一个可行解,即使当前解的位置较差,也无法否认它往全局最优解进化的可能,故在构建蜜源时,利用所有的蜜源信息构建加权中心.具体过程如下.

在每代蜜源的迭代更新之前,通过上代蜜源信息,依据蜜源的优劣情况赋予不同的权重值,排名第1赋予权重值SN,排名第2赋予权重值SN-1,依次类推,最差蜜源赋予权重值1,计算加权中心公式为

$$WC = \frac{\sum_{i=1}^{SN} X_i(SN - i + 1)}{\sum_{i=1}^{SN} i}. \quad (5)$$

其中:SN、 i 分别为蜜源数与蜜源排序数, X_i 为排名第 i 的蜜源,SN- i +1为蜜源 X_i 的权重值,WC为加权中心.

2.2 加权中心引导观察蜂更新策略

在ABC算法寻找最优解的过程中,分散在搜索域各个位置的雇佣蜂均有机会在邻域附近产生新蜜源,故雇佣蜂阶段的全局搜索能力强于局部搜索能力.在观察蜂阶段,质量较好的蜜源更新机会多于质量较差的蜜源,故在观察蜂阶段,蜜源更新更多地集中在蜜源质量较好的邻域,但选取邻域蜜源亦在搜索域内.因此,ABC算法全局搜索能力强于局部搜索能力^[17-19].为加强ABC算法的局部搜索能力,本文通过加权中心引导观察蜂进行搜索,并由之前的单维更新改为全维更新.观察蜂更新公式如下所示:

$$v_{i,d} = x_{i,j} + \text{rand}(-1, 1)(x_{i,j} - WC_j). \quad (6)$$

其中: $v_{i,d}$ 为第 i 只观察蜂第 d 维的候选蜜源信息, $d = 1, 2, \dots, D$; $x_{i,j}$ 为第 i 只观察蜂第 j 维更新之前的蜜源信息; WC_j 为加权中心的第 j 维蜜源信息.

加权中心与全维更新的结合构成了本文的WCABC算法.

WCABC算法流程如下.

- 1) 设置参数SN、limit、 D //初始化阶段
- 2) 式(1)对蜜源进行初始化($i = 1, 2, \dots, SN$, $j = 1, 2, \dots, D$)
- 3) 计算适应值
- 4) while(算法终止条件) do
- 5) 由式(5)构建WC
- 6) if $f(WC) < f(Gbest)$
- 7) $Gbest = WC$ //雇佣蜂阶段
- 8) for $i = 1$ to SN
- 9) 随机选取一邻域 $k = 1, 2, \dots, SN, k \neq i$
- 10) 随机选取一维度 $j = 1, 2, \dots, D$
- 11) 雇佣蜂由式(3)对蜜源进行更新
- 12) end for
- 13) 计算适应值 f_i 及选择概率 p_i //观察蜂阶段
- 14) 依据式(4)选取观察蜂 i
- 15) 随机选取一维度 $j = 1, 2, \dots, D$
- 16) for $d = 1$ to D
- 17) 观察蜂 i 由式(6)对蜜源进行更新
- 18) end for
- 19) //侦查蜂阶段
- 20) 式(1)对蜜源超过限制次数进行初始化
- 21) 记录Gbest
- 22) end while

在初始化阶段(步骤2)~步骤3),采用式(1)在搜索域内随机产生初始值;随后进入循环迭代过程(步骤4)~步骤22),在雇佣蜂阶段之前需依据式(5)构造加权中心,其序数由快速排序法求得,若加权中心优于Gbest,则用加权中心取代Gbest,否则不取代(步骤6)和步骤7);本文未对雇佣蜂阶段作出改进,同标准ABC算法;观察蜂依据轮盘赌策略对雇佣蜂跟随,该阶段为全维更新故不随机选择一维度,随后根据式(6)更新蜜源(步骤14)~步骤18));若某蜜源的蜜源限制次数超出规定的阈值,则丢弃原蜜源,采用式(1)重新在搜索域内产生一个新蜜源(步骤20)).

3 仿真实验与结果分析

3.1 基准测试函数

为对WCABC算法的有效性进行分析,在22个测试函数上进行测试,各函数表达式见文献[20].文献[12]指出,当适应值小于 $1e-20$,计算机求解的适应度可能会被忽略,故本文实验均采用比较适应值来判断蜜源优劣.在22个测试函数中, $f_1 \sim f_6$ 和 f_8 是连续的单峰函数, f_7 是不连续的单峰阶梯函数, f_9 是噪声4次函数, f_{10} 在2维和3维上是单峰的,在高维上可能是多峰的,且局部最优值的数量随着维度的增加呈指

数增长^[11], $f_{11} \sim f_{22}$ 是多峰函数. 单峰函数用来测试改进算法的收敛速度, 多峰函数测试改进算法跳出局部最优的能力.

3.2 算法的耗时分析

WCABC算法需对蜜源加以排序, 且排序是相对耗时的过程. 针对此问题, 本文对算法耗时问题加以分析. 表1给出了4个算法在 $D = 30$ 的情形下, 独立运行25次的时间(单位: s). WCABC算法在22个基准函数的平均耗时比ABC算法增加约1.32倍, 比MSSABC^[8]算法增加约1.29倍, 比ABCMSSCE^[14]算法增加约1.03倍.

表1 各算法运行时间

函数	ABC	MSSABC	ABCMSSCE	WCABC
f_1	1.551	1.851	3.036	3.683
f_2	9.633	10.006	11.219	10.806
f_3	1.565	1.856	3.066	3.772
f_4	10.760	11.001	12.316	27.081
f_5	3.439	4.066	5.279	4.808
f_6	3.179	3.379	4.614	4.058
f_7	8.075	8.131	9.342	9.077
f_8	1.802	1.973	3.195	2.802
f_9	1.636	1.802	3.026	2.509
f_{10}	1.960	2.156	3.387	2.835
f_{11}	5.890	5.991	7.156	6.776
f_{12}	8.672	8.194	9.588	9.691
f_{13}	7.963	8.096	9.308	8.929
f_{14}	8.365	8.610	9.845	9.536
f_{15}	6.613	6.550	7.824	7.308
f_{16}	10.350	10.388	11.556	11.577
f_{17}	11.528	11.083	12.325	14.069
f_{18}	7.053	7.274	8.452	7.298
f_{19}	9.594	9.762	10.992	10.951
f_{20}	440.932	441.193	442.368	440.978
f_{21}	9.762	9.966	11.266	10.613
f_{22}	23.523	23.650	24.559	22.937

WCABC算法在函数评估次数为终止条件的情况下, 平均耗时相比于比较算法有所增加. 若给定等同的时间, 则WCABC算法能寻到更优解, 表明WCABC在性能方面具备更好的优势. 此实验中, 通过给定等同的时间, 分别测试比较算法所能达到的寻优精度. 若随意给定一个等同的时间, 则各算法独立运行的次数可能不等. 为公平比较, 通过程序得到ABC算法独立运行一次的时间(下文均称总时间), 且分别取总时间的1/2为给定的时间, 得到4个算法所

能达到的精度. 表2为各算法对22个函数在给定时间所达到的精度. 实验参数问题维度 $D = 30$, 其他参数不变. 综合表1和表2可知, WCABC算法通过构建加权中心, 在寻优过程中所花费的时间有所增加. 该算法牺牲了时间, 换来快速收敛的特性, 在相同时间的情况下, WCABC算法所能达到的最终结果仍能保持较高的精度.

表2 各算法在给定的时间所能达到的精度

函数	设定时间	ABC	MSSABC	ABCMSSCE	WCABC
f_1	0.0375	7.18e-08	1.107e-18	9.87e-13	4.66e-172
f_2	0.1985	6.05e-03	1.24e-10	1.86e-15	1.74e-240
f_3	0.0365	4.19e-08	1.06e-18	6.08e-13	6.67e-214
f_4	0.2220	2.95e-18	1.09e-32	2.11e-69	2.85e-75
f_5	0.0735	1.62e-04	1.96e-10	1.48e-10	6.07e-129
f_6	0.0670	4.25e+01	1.17e+01	1.50e+00	2.93e-26
f_7	0.1670	0.0e+00	0.0e+00	0.0e+00	0.0e+00
f_8	0.0415	7.18e-66	7.18e-66	7.18e-66	7.18e-66
f_9	0.0385	2.19e-01	4.15e-02	2.55e-02	2.84e-08
f_{10}	0.0455	1.72e-01	4.89e+00	4.42e+01	2.85e+01
f_{11}	0.1230	2.22e+00	2.13e-15	1.76e-13	0.0e+00
f_{12}	0.1280	5.65e-01	0.0e+00	0.0e+00	0.0e+00
f_{13}	0.1640	7.18e-08	6.82e-10	2.35e-13	0.00e+00
f_{14}	0.1730	1.08e+02	8.36e-01	5.09e-07	0.00e+00
f_{15}	0.1380	1.90e-01	1.77e-03	1.08e+00	5.89e-16
f_{16}	0.2095	4.49e-07	1.99e-18	1.87e-19	1.57e-32
f_{17}	0.2330	3.59e-06	6.59e-16	1.20e-18	1.35e-32
f_{18}	0.1470	1.16e-03	3.87e-05	1.40e-08	1.57e-32
f_{19}	0.2005	9.25e-07	3.83e-15	1.01e-18	1.35e-32
f_{20}	8.8610	1.90e-01	0.0e+00	0.0e+00	0.0e+00
f_{21}	0.2000	-78.332	-78.332	-78.332	-78.332
f_{22}	0.4780	-28.617	-29.203	-29.207	-28.807

3.3 与经典改进算法比较分析

将WCABC与10个改进算法和标准ABC算法在22个测试函数上进行比较分析, 各测试函数独立运行25次, 种群数 $SN = 50$, 问题维度分别为 $D = 30$, $D = 100$, 限制次数 $limit = SN \cdot D$, 函数评估次数为 $FES = 5000D$. 实验数据见表3~表6. 表中: 较优结果加粗, “w”、“e”、“1”分别表示WCABC算法优于、等于、差于相比较算法. 函数 f_{22} 由于原始论文关键参数不一致, 作者根据相应论文算法实现寻优结果, 并用“*”表示.

在单峰函数 $f_1 \sim f_6$ 中, WCABC算法除函数 f_4 弱于ABCMSSCE算法外, 剩余算法在收敛速度、求解的精度均有明显优势. 函数 f_8 都获得了相同的解,

表 3 $D = 30$ WCABC 与各个算法的比较 (一)

函数		ABC	MSSABC ^[8]	CABC ^[20]	ABCMSSCE ^[14]
f_1	Mean(std)	1.04e-17(1.20e-17)(w)	2.01e-41(1.28e-40)(w)	2.41e-50(3.19e-50)(w)	2.94e-54(1.43e-53)(w)
f_2	Mean(std)	4.38e-10(4.72e-10)(w)	4.74e-34(9.09e-33)(w)	1.17e-41(2.83e-41)(w)	1.78e-51(6.60e-51)(w)
f_3	Mean(std)	1.14e-19(9.89e-20)(w)	5.54e-42(4.07e-41)(w)	1.15e-51(1.38e-51)(w)	7.57e-55(7.688e-54)(w)
f_4	Mean(std)	2.02e-31(5.30e-31)(w)	1.96e-64(3.744e-63)(w)	1.60e-60(6.69e-60)(w)	1.34e-152(3.28e-151)(l)
f_5	Mean(std)	7.69e-11(3.04e-11)(w)	2.78e-22(7.70e-22)(w)	1.59e-26(1.04e-26)(w)	1.73e-33(4.94e-33)(w)
f_6	Mean(std)	4.39e+00(1.07e+00)(w)	2.05e+00(1.83e+00)(w)	5.83e-02(1.33e-02)(w)	1.46e-04(4.24e-04)(w)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_8	Mean(std)	7.18e-66(5.92e-71)(e)	7.18e-66(1.05e-80)(e)	7.18e-66(8.92e-80)(e)	7.18e-66(1.05e-80)(e)
f_9	Mean(std)	6.02e-02(1.09e-02)(w)	2.32e-02(2.63e-02)(w)	2.15e-02(4.59e-03)(w)	6.25e-03(7.38e-03)(w)
f_{10}	Mean(std)	5.45e-02(5.86e-02)(w)	4.81e-01(7.16e+00)(w)	1.28e-01(2.03e-01)(w)	4.16e+01(1.55e+02)(w)
f_{11}	Mean(std)	3.50e-14(1.35e-13)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_{12}	Mean(std)	1.70e-12(4.36e-12)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_{13}	Mean(std)	2.36e-14(5.62e-14)(w)	4.21e-16(1.03e-14)(w)	1.05e-14(5.20e-14)(w)	0.00e+00(0.00e+00)(e)
f_{14}	Mean(std)	4.58e-12(1.59e-12)(w)	4.27e-02(1.04e+00)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_{15}	Mean(std)	4.31e-09(1.85e-09)(w)	3.03e-14(1.58e-14)(w)	2.66e-15(0.00e+00)(w)	3.94e-01(3.36e+00)(w)
f_{16}	Mean(std)	1.03e-18(6.90e-19)(w)	1.57e-32(4.10e-47)(e)	1.57e-32(5.59e-48)(e)	1.57e-32(4.10e-47)(e)
f_{17}	Mean(std)	4.88e-18(5.03e-18)(w)	1.35e-32(2.74e-47)(e)	1.50e-33(0.00e+00)(l)	1.35e-32(2.73e-47)(e)
f_{18}	Mean(std)	2.35e-06(1.66e-06)(w)	5.05e-07(7.40e-06)(w)	9.41e-28(1.03e-27)(w)	6.59e-16(4.94e-15)(w)
f_{19}	Mean(std)	4.46e-14(5.39e-14)(w)	1.35e-31(1.09e-46)(e)	1.35e-31(2.23e-47)(e)	1.35e-31(1.09e-46)(e)
f_{20}	Mean(std)	2.06e-02(2.35e-02)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_{21}	Mean(std)	-78.332(0.00e+00)(e)	-78.332(5.85e-14)(e)	-78.332(0.00e+00)(e)	-78.332(4.02e-14)(e)
f_{22*}	Mean(std)	-28.620(2.73e-01)(w)	-27.914(1.04e+00)(w)	-29.602(1.96e-01)(l)	-29.594(1.968 838e-01)(l)
	w / e / l	19 / 3 / 0	18 / 4 / 0	11 / 9 / 2	

函数		MABC ^[11]	DFSABC_elite ^[17]	WCABC
f_1	Mean(std)	2.36e-40(1.85e-40)(w)	4.14e-82(8.76e-82)(w)	0.00e+00(0.00e+00)
f_2	Mean(std)	5.46e-37(4.09e-37)(w)	5.37e-78(8.66e-78)(w)	1.99e-255(0.00e+00)
f_3	Mean(std)	2.93e-41(2.35e-41)(w)	2.84e-83(4.66e-83)(w)	0.00e+00(0.00e+00)
f_4	Mean(std)	6.53e-84(3.26e-83)(w)	2.41e-110(1.19e-109)(w)	9.94e-143(2.44e-141)
f_5	Mean(std)	1.50e-21(6.64e-22)(w)	2.06e-42(2.08e-42)(w)	1.24e-280(0.00e+00)
f_6	Mean(std)	1.93e+00(1.32e+00)(w)	5.08e-07(3.69e-07)(w)	2.13e-049(3.65e-48)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_8	Mean(std)	7.18e-66(9.74e-80)(e)	7.18e-66(3.23e-81)(e)	7.18e-066(1.05e-80)
f_9	Mean(std)	2.77e-02(6.36e-03)(w)	1.20e-02(3.80e-03)(w)	6.50e-05(2.34e-04)
f_{10}	Mean(std)	1.10e+00(2.86e+00)(w)	3.45e+00(1.45e+01)(w)	0.00e+00(0.00e+00)
f_{11}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{12}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{13}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{14}	Mean(std)	3.49e-12(7.28e-13)(w)	4.37e-13(1.09e-12)(w)	0.00e+00(0.00e+00)
f_{15}	Mean(std)	7.07e-15(2.36e-15)(w)	3.80e-15(1.69e-15)(w)	5.89e-016(0.00e+00)
f_{16}	Mean(std)	1.57e-32(5.59e-48)(e)	1.57e-32(5.59e-48)(e)	1.57e-32(1.37e-47)
f_{17}	Mean(std)	1.50e-33(0.00e+00)(l)	1.50e-33(0.00e+00)(l)	1.35e-032(2.74e-47)
f_{18}	Mean(std)	1.43e-21(1.75e-21)(w)	3.10e-40(1.03e-39)(w)	7.89e-299(0.00e+00)
f_{19}	Mean(std)	1.35e-31(2.23e-47)(e)	1.35e-31(2.23e-47)(e)	1.35e-31(0.00e+00)
f_{20}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{21}	Mean(std)	-78.332(5.02e-15)(e)	-78.332(5.02e-15)(e)	-78.332(0.00e+00)
f_{22*}	Mean(std)	-29.617(1.53e-01)(l)	-28.960(1.32e+00)(w)	-29.211 0(1.04e+00)
	9 / 11 / 2	12 / 9 / 1	11 / 9 / 2	

表4 D = 30 WCABC与各个算法的比较(二)

函数		dABC ^[10]	qABC ^[9]	EABC ^[21]
f_1	Mean(std)	4.21e-13(3.92e-13)(w)	3.38e-15(5.42e-15)(w)	5.85e-62(2.90e-61)(w)
f_2	Mean(std)	8.76e-08(1.53e-07)(w)	1.31e-10(2.21e-10)(w)	9.26e-60(1.41e-59)(w)
f_3	Mean(std)	3.75e-14(1.04e-13)(w)	2.47e-16(2.43e-16)(w)	4.50e-65(5.16e-65)(w)
f_4	Mean(std)	4.09e-26(1.12e-25)(w)	2.99e-21(5.31e-21)(w)	9.57e-33(3.42e-32)(w)
f_5	Mean(std)	4.26e-08(1.48e-08)(w)	1.17e-08(3.31e-09)(w)	9.45e-34(8.43e-34)(w)
f_6	Mean(std)	1.02e+00(5.26e-01)(w)	9.87e-02(2.30e-02)(w)	2.43e+01(5.22e+00)(w)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_8	Mean(std)	7.18e-66(5.92e-71)(e)	7.18e-66(3.50e-72)(e)	7.18e-66(7.49e-67)(e)
f_9	Mean(std)	6.19e-02(1.17e-02)(w)	2.73e-02(6.83e-03)(w)	1.65e-02(3.68e-03)(w)
f_{10}	Mean(std)	1.38e-01(1.89e-01)(w)	5.47e-01(5.00e-01)(w)	1.14e+00(2.94e+00)(w)
f_{11}	Mean(std)	8.16e-13(1.26e-12)(w)	1.33e-10(2.15e-10)(w)	3.82e-02(1.91e-01)(w)
f_{12}	Mean(std)	4.74e-11(9.50e-11)(w)	5.28e-10(5.37e-10)(w)	1.20e-01(3.32e-01)(w)
f_{13}	Mean(std)	4.06e-04(2.03e-03)(w)	5.47e-12(2.39e-11)(w)	4.29e-08(2.14e-07)(w)
f_{14}	Mean(std)	1.04e-11(6.27e-12)(w)	4.16e-10(1.05e-09)(w)	3.35e-12(8.60e-13)(w)
f_{15}	Mean(std)	3.83e-07(1.85e-07)(w)	1.67e-06(8.22e-07)(w)	2.73e-05(1.36e-04)(w)
f_{16}	Mean(std)	5.56e-14(6.04e-14)(w)	2.31e-14(9.81e-14)(w)	1.57e-32(5.59e-48)(e)
f_{17}	Mean(std)	1.94e-13(2.27e-13)(w)	1.72e-15(1.86e-15)(w)	1.50e-33(0.00e+00)(l)
f_{18}	Mean(std)	7.38e-06(6.48e-06)(w)	8.87e-06(2.10e-05)(w)	6.00e-17(3.41e-16)(w)
f_{19}	Mean(std)	9.22e-12(1.02e-11)(w)	2.40e-09(3.34e-09)(w)	1.35e-31(2.23e-47)(e)
f_{20}	Mean(std)	3.32e-02(3.58e-02)(w)	1.10e-02(1.03e-02)(w)	6.03e-03(1.30e-02)(w)
f_{21}	Mean(std)	-78.332(2.0e-15)(e)	-78.332(4.10e-15)(e)	-78.332(2.90e-15)(e)
f_{22*}	Mean(std)	-29.387(2.99e-01)(l)	-29.609(9.99e-02)(l)	-29.620(4.65e-02)(l)
	w/e/1	18/3/1	18/3/1	15/5/2

函数		ABCVSS ^[22]	BABC ^[23]	WCABC
f_1	Mean(std)	2.40e-35(8.54e-35)(w)	1.14e-43(1.77e-43)(w)	0.00e+00(0.00e+00)
f_2	Mean(std)	2.29e-27(9.79e-27)(w)	4.18e-30(3.33e-17)(w)	1.99e-255(0.00e+00)
f_3	Mean(std)	9.40e-37(2.54e-36)(w)	7.40e-15(3.70e-14)(w)	0.00e+00(0.00e+00)
f_4	Mean(std)	4.31e-44(1.40e-43)(w)	4.96e-90(1.54e-89)(w)	9.94e-143(2.44e-141)
f_5	Mean(std)	7.03e-19(2.18e-18)(w)	1.61e-24(8.21e-25)(w)	1.24e-280(0.00e+00)
f_6	Mean(std)	2.56e-01(9.19e-02)(w)	1.71e+00(1.15e+00)(w)	2.13e-049(3.65e-48)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{82}	Mean(std)	7.18e-66(9.98e-78)(e)	7.18e-66(2.04e-77)(e)	7.18e-066(1.05e-80)
f_9	Mean(std)	3.25e-02(4.58e-02)(w)	2.70e-02(8.28e-03)(w)	6.50e-05(2.34e-04)
f_{10}	Mean(std)	1.28e-01(2.03e-01)(w)	3.97e-02(4.96e-02)(w)	0.00e+00(0.00e+00)
f_{11}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{12}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{13}	Mean(std)	3.45e-11(1.73e-10)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{14}	Mean(std)	1.60e-12(3.45e-13)(w)	2.18e-13(7.80e-13)(w)	0.00e+00(0.00e+00)
f_{15}	Mean(std)	6.50e-15(2.27e-15)(w)	5.65e-15(1.33e-15)(w)	5.89e-016(0.00e+00)
f_{16}	Mean(std)	1.57e-32(5.59e-48)(e)	8.98e-14(4.49e-13)(w)	1.57e-32(1.37e-47)
f_{17}	Mean(std)	1.50e-33(0.00e+00)(l)	1.50e-33(8.28e-33)(l)	1.35e-032(2.74e-47)
f_{18}	Mean(std)	6.26e-18(2.91e-17)(w)	3.33e-17(1.28e-16)(w)	7.89e-299(0.00e+00)
f_{19}	Mean(std)	1.35e-31(2.23e-47)(e)	1.35e-31(2.23e-47)(e)	1.35e-31(0.00e+00)
f_{20}	Mean(std)	0.00e+00(0.00e+00)(e)	2.63e-05(1.32e-04)(w)	0.00e+00(0.00e+00)
f_{21}	Mean(std)	-78.332(1.05e-14)(e)	-78.332(1.23e-14)(e)	-78.332(0.00e+00)
f_{22*}	Mean(std)	-28.310(1.0e+00)(w)	-29.568(2.42e-01)(l)	-29.211 0(1.04e+00)
	w/e/1	13/8/1	13/7/2	

表 5 $D = 100$ WCABC 与各个算法的比较(一)

函数		ABC	MSSABC ^[8]	CABC ^[20] ABCMSSE ^[14]	
f_1	Mean(std)	4.78e-16(4.79e-16)(w)	1.29e-38(1.28e-37)(w)	9.73e-49(1.05e-48)(w)	9.27e-64(4.17e-63)(l)
f_2	Mean(std)	5.30e-09(4.89e-09)(w)	1.45e-32(1.13e-31)(w)	5.41e-40(8.05e-40)(w)	6.86e-61(6.31e-60)(w)
f_3	Mean(std)	1.03e-16(8.69e-17)(w)	3.00e-38(5.77e-37)(w)	2.83e-49(2.58e-49)(w)	7.06e-64(5.72e-63)(w)
f_4	Mean(std)	1.17e-31(4.89e-31)(w)	2.47e-64(5.26e-63)(w)	2.01e-60(7.13e-60)(w)	5.26e-159(9.46e-158)(l)
f_5	Mean(std)	1.27e-09(4.11e-10)(w)	7.42e-21(2.25e-20)(w)	1.25e-25(4.65e-26)(w)	3.42e-42(1.22e-41)(w)
f_6	Mean(std)	2.85e+01(1.31e+00)(w)	3.14e+01(7.99e+00)(w)	3.71e+00(3.41e-01)(w)	3.00e+00(3.36e+00)(w)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_8	Mean(std)	7.12e-218(0.00e+00)(e)	7.12e-218(0.00e+00)(e)	7.12e-218(0.00e+00)(e)	7.12e-218(0.00e+00)(e)
f_9	Mean(std)	2.89e-01(5.49e-02)(w)	1.29e-01(7.61e-02)(w)	1.05e-01(1.43e-02)(w)	1.64e-02(1.32e-02)(w)
f_{10}	Mean(std)	1.82e-01(2.04e-01)(w)	1.94e+01(1.59e+02)(w)	1.56e-01(2.23e-01)(w)	1.48e+02(2.13e+02)(w)
f_{11}	Mean(std)	2.28e-08(1.14e-07)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	9.15e+00(1.56e+01)(w)
f_{12}	Mean(std)	1.69e-01(3.97e-01)(w)	0.00e+00(0.00e+00)(e)	4.00e-02(2.00e-01)(w)	1.10e+01(1.35e+01)(w)
f_{13}	Mean(std)	9.77e-17(1.90e-16)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_{14}	Mean(std)	5.79e-11(7.42e-12)(l)	4.73e+00(1.16e+02)(w)	7.28e-12(0.00e+00)(l)	3.79e+02(1.05e+03)(w)
f_{15}	Mean(std)	1.42e-08(4.33e-09)(w)	1.40e-11(2.63e-10)(w)	1.33e-14(0.00e+00)(w)	1.99e+01(1.38e-02)(w)
f_{16}	Mean(std)	1.48e-17(1.08e-17)(e)	4.71e-33(0.00e+00)(e)	4.71e-33(6.98e-49)(e)	4.77e-33(1.51e-33)(w)
f_{17}	Mean(std)	3.27e-16(4.05e-16)(w)	1.35e-32(2.74e-47)(e)	1.50e-33(0.00e+00)(l)	4.39e-04(1.0e-02)(w)
f_{18}	Mean(std)	3.96e-04(4.66e-04)(w)	2.59e-05(1.21e-04)(w)	1.05e-26(6.96e-27)(w)	1.52e-14(3.04e-14)(w)
f_{19}	Mean(std)	4.47e-14(5.35e-14)(w)	1.35e-31(1.09e-46)(e)	1.35e-31(2.23e-47)(e)	1.44e-31(8.58e-32)(w)
f_{20}	Mean(std)	4.54e-01(1.24e-01)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_{21}	Mean(std)	-78.332(2.23e-14)(e)	-78.332(1.06e-13)(e)	-78.332(0.00e+00)(e)	-78.332(2.77e-01)(e)
f_{22*}	Mean(std)	-83.191(4.53e+00)(w)	-88.915(3.72e+00)(w)	-91.953(2.95e+00)(w)	-93.917(3.26e+00)(l)
	w/e/1	17/4/1	12/10/0	12/8/2	14/5/3

函数		MABC ^[11]	DFSABC_elite ^[17]	WCABC
f_1	Mean(std)	2.50e-36(1.14e-36)(w)	9.57e-81(9.73e-81)(l)	4.17e-059(1.02e-057)
f_2	Mean(std)	5.49e-33(2.04e-33)(w)	5.50e-77(6.42e-77)(w)	8.67e-142(2.12e-140)
f_3	Mean(std)	1.19e-36(6.34e-37)(w)	3.90e-81(8.94e-81)(w)	6.67e-070(1.49e-068)
f_4	Mean(std)	4.69e-88(2.07e-87)(w)	2.31e-109(7.26e-109)(w)	1.02e-125(2.50e-124)
f_5	Mean(std)	3.08e-19(7.21e-20)(w)	2.13e-41(2.55e-41)(w)	0.00e+00(0.000e+00)
f_6	Mean(std)	3.88e+01(3.70e+00)(w)	3.90e-02(1.50e-02)(w)	5.74e-097(1.40e-095)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_8	Mean(std)	7.12e-218(0.00e+00)(e)	7.12e-218(0.00e+00)(e)	7.12e-218(0.00e+00)
f_9	Mean(std)	1.75e-01(1.67e-02)(w)	5.02e-02(5.94e-03)(w)	1.68e-05(1.19e-004)
f_{10}	Mean(std)	2.31e+00(2.62e+00)(w)	6.28e+00(1.90e-01)(w)	1.53e-05(3.75e-04)
f_{11}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{12}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{13}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{14}	Mean(std)	3.58e-11(2.91e-12)(l)	2.18e-11(3.52e-17)(l)	1.091e-010(0.00e+00)
f_{15}	Mean(std)	2.88e-14(2.02e-15)(w)	1.30e-14(9.84e-16)(w)	5.89e-016(0.00e+00)
f_{16}	Mean(std)	4.71e-33(6.98e-49)(e)	4.71e-33(6.84e-48)(e)	4.71e-033(6.84e-48)
f_{17}	Mean(std)	1.50e-33(0.00e+00)(l)	1.50e-33(0.00e+00)(l)	1.35e-032(2.74e-47)
f_{18}	Mean(std)	3.90e-18(3.06e-18)(w)	2.38e-16(3.55e-16)(w)	0.00e+00(0.00e+00)
f_{19}	Mean(std)	1.35e-31(2.23e-47)(e)	1.35e-31(2.23e-47)(e)	1.35e-31(0.00e+00)
f_{20}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{21}	Mean(std)	-78.332(7.11e-15)(e)	-178.332(0.00e+00)(e)	-78.332(0.00e+00)
f_{22*}	Mean(std)	-89.086(2.19e+00)(w)	-92.875(2.23e+00)(l)	-92.58(6.52e+00)
	w/e/1	11/9/2	9/9/4	

表6 $D = 100$ WCABC与各个算法的比较(二)

函数		dABC ^[10]	qABC ^[9]	EABC ^[21]
f_1	Mean(std)	5.02e-12(3.49e-12)(w)	8.08e-15(4.44e-15)(w)	1.83e-59(1.99e-59)(l)
f_2	Mean(std)	1.41e-07(1.70e-07)(w)	4.15e-10(4.20e-10)(w)	5.92e-36(2.96e-35)(w)
f_3	Mean(std)	1.38e-12(9.92e-13)(w)	3.17e-15(1.98e-15)(w)	8.14e-60(7.86e-60)(w)
f_4	Mean(std)	9.52e-26(3.55e-25)(w)	1.63e-21(3.83e-21)(w)	2.39e-31(1.13e-30)(w)
f_5	Mean(std)	5.21e-07(1.40e-07)(w)	4.26e-08(7.69e-09)(w)	3.11e-31(1.02e-31)(w)
f_6	Mean(std)	9.91e+00(1.98e-13)(w)	6.50e-01(9.51e-02)(w)	4.90e+01(5.45e+00)(w)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)
f_8	Mean(std)	7.13e-218(0.00e+00)(e)	7.12e-218(0.00e+00)(e)	9.36e-218(0.00e+00)(e)
f_9	Mean(std)	2.87e-01(5.16e-02)(w)	7.34e-02(1.29e-02)(w)	9.36e-02(1.01e-02)(w)
f_{10}	Mean(std)	3.40e-01(4.89e-01)(w)	8.81e-01(6.59e-01)(w)	4.92e+00(1.55e+01)(w)
f_{11}	Mean(std)	1.03e-08(4.82e-08)(w)	6.26e-10(1.24e-09)(w)	3.98e-02(1.99e-01)(w)
f_{12}	Mean(std)	1.77e-01(3.73e-01)(w)	1.90e-09(1.56e-09)(w)	2.80e-01(4.58e-01)(w)
f_{13}	Mean(std)	3.89e-12(1.22e-11)(w)	2.15e-13(4.31e-13)(w)	0.00e+00(0.00e+00)(e)
f_{14}	Mean(std)	6.17e-11(7.61e-12)(l)	1.61e-09(6.38e-09)(w)	3.79e+01(6.59e+01)(w)
f_{15}	Mean(std)	1.43e-06(5.22e-07)(w)	2.34e-06(1.01e-06)(w)	2.56e-10(1.28e-09)(w)
f_{16}	Mean(std)	1.98e-13(1.31e-13)(w)	6.62e-16(1.04e-15)(w)	4.71e-33(6.98e-49)(w)
f_{17}	Mean(std)	2.55e-12(1.77e-12)(w)	9.56e-15(8.32e-15)(w)	1.50e-33(0.00e+00)(l)
f_{18}	Mean(std)	7.62e-04(9.02e-04)(w)	4.17e-05(6.72e-05)(w)	1.71e-16(4.37e-16)(w)
f_{19}	Mean(std)	2.51e-11(3.27e-11)(w)	2.23e-09(3.04e-09)(w)	1.35e-31(2.23e-47)(e)
f_{20}	Mean(std)	4.64e-01(1.71e-01)(w)	5.25e-02(3.04e-02)(w)	1.14e-02(2.67e-02)(w)
f_{21}	Mean(std)	78.332(1.88e-14)(e)	-78.332(4.07e-14)(e)	-78.332(2.15e-14)(e)
f_{22*}	Mean(std)	-97.710(8.20e-01)(l)	-96.974(4.22e+00)(l)	-87.988(4.58e+00)(w)
	w/e/1	17/3/2	18/3/1	15/5/2

函数		ABCVSS ^[22]	BABC ^[23]	WCABC
f_1	Mean(std)	3.84e-31(1.39e-30)(w)	2.59e-13(1.30e-12)(w)	4.17e-059(1.02e-057)
f_2	Mean(std)	2.42e-23(8.37e-23)(w)	2.51e-14(1.25e-13)(w)	8.67e-142(2.12e-140)
f_3	Mean(std)	9.03e-29(4.51e-28)(w)	1.38e-14(4.52e-14)(w)	6.67e-070(1.49e-068)
f_4	Mean(std)	4.81e-47(1.72e-46)(w)	2.44e-91(9.04e-91)(w)	1.02e-125(2.50e-124)
f_5	Mean(std)	2.26e-16(4.87e-16)(w)	3.09e-07(1.54e-06)(w)	0.00e+00(0.000e+00)
f_6	Mean(std)	1.72e+01(1.43e+00)(w)	1.80e+01(4.07e+00)(w)	5.74e-097(1.40e-095)
f_7	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_8	Mean(std)	7.12e-218(0.00e+00)(e)	7.12e-218(2.32e-16)(e)	7.12e-218(0.00e+00)
f_9	Mean(std)	1.50e-01(2.64e-02)(w)	1.82e-01(3.92e-02)(w)	1.68e-05(1.19e-004)
f_{10}	Mean(std)	8.46e-02(1.65e-01)(w)	3.76e-02(4.32e-02)(w)	1.53e-05(3.75e-04)
f_{11}	Mean(std)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{12}	Mean(std)	0.00e+00(0.00e+00)(e)	4.00e-02(2.00e-01)(w)	0.00e+00(0.00e+00)
f_{13}	Mean(std)	1.67e-15(8.33e-15)(w)	0.00e+00(0.00e+00)(e)	0.00e+00(0.00e+00)
f_{14}	Mean(std)	2.82e-11(7.07e-12)(l)	2.27e-11(3.20e-12)(l)	1.091e-010(0.00e+00)
f_{15}	Mean(std)	3.58e-14(1.03e-14)(w)	2.09e-14(2.77e-15)(w)	5.89e-016(0.00e+00)
f_{16}	Mean(std)	6.29e-32(2.05e-31)(w)	7.89e-15(3.89e-14)(w)	4.71e-033(6.84e-48)
f_{17}	Mean(std)	7.68e-32(3.58e-31)(w)	1.50e-33(0.00e+00)(l)	1.35e-032(2.74e-47)
f_{18}	Mean(std)	1.32e-13(5.33e-13)(w)	1.09e-16(2.32e-16)(w)	0.00e+00(0.00e+00)
f_{19}	Mean(std)	1.35e-31(2.23e-47)(e)	1.35e-31(2.23e-47)(e)	1.35e-31(0.00e+00)
f_{20}	Mean(std)	1.36e-14(6.82e-14)(w)	1.80e-04(8.90e-04)(w)	0.00e+00(0.00e+00)
f_{21}	Mean(std)	78.332(2.90e-15)(e)	-78.332(0.00e+00)(e)	-78.332(0.00e+00)
f_{22*}	Mean(std)	89.591(2.95e+00)(w)	-87.478(4.88e+00)(w)	-92.58(6.52e+00)
	w/e/1	15/6/1	14/6/2	

差异仅表现在std(标准差). 在多峰函数中, 由于 f_7 为阶梯函数, 极值点较少且不太复杂, 最优解为区间段, 故所有算法都获得了最优值. 函数 f_9 为噪声的4次函数, 其最优值很难求解, 在对比算法中均达到了精度接近的解, WCABC通过其快速的收敛特性, 获得了优于其他算法的最优解. f_{10} 是Rosenbrock函数, 难以获得最优值, WCABC在该函数中对比于其他算法时好时坏. 函数 $f_{11} \sim f_{20}$ 中, WCABC在最优值精度上稍优于其他改进算法. 函数 f_{21} 类似于函数 f_8 , 所有算法都获得了相同解, 差异亦表现在std中.

非参数Friedman检验能够综合衡量算法的优劣. 表7给出了全部算法的Friedman检验效果排名, 秩均值越小表明算法整体性能越好. 在30维与100维两种情况下, WCABC算法均排名第一, 综合性能最好. 这也表明, WCABC算法在高维问题上, 同样有很好的优势.

表7 $D = 30, 100$ 所有算法的Friedman检验

$D = 30$		$D = 100$	
算法	秩均值	算法	秩均值
WCABC	3.43	WCABC	3.41
DFSABC_elite	4.39	DFSABC_elite	3.80
CABC	4.82	CABC	4.66
ABCMSSCE	5.27	MABC	5.70
MABC	5.98	ABCMSSCE	6.41
BABC	6.05	ABCVSS	6.44
ABCVSS	6.73	MSSABC	6.68
EABC	6.91	BABC	6.80
MSSABC	6.95	EABC	7.05
ABC	8.34	qABC	8.32
qABC	9.27	ABC	9.02
dABC	9.56	dABC	9.52

3.4 30维下算法适应值进化曲线

为直观看出各算法的收敛趋势, 针对维度 $D = 30$ 且其他参数不变的情况下, 分别给出各算法在部分基准测试函数的收敛曲线如图1所示, 部分函数分别为单峰函数 f_1, f_4 , 多峰函数 f_{12}, f_{16}, f_{22} . 由图1可见, WCABC算法具有较快的收敛速度和较高的收敛精度. 在单峰函数 f_1, f_4 上相比其他算法收敛速度具有显著优势. 在多峰函数 f_{12} 中, WCABC算法前期收敛速度较快, 2万次评估次数即达到函数理论最优值. 在函数 f_{16} 中, 2万次评估次数陷入局部最优, 但约4万次收敛速度明显加强, 表明算法跳出局部最优能力较强. 在函数 f_{22} 中各算法均达到最优值.

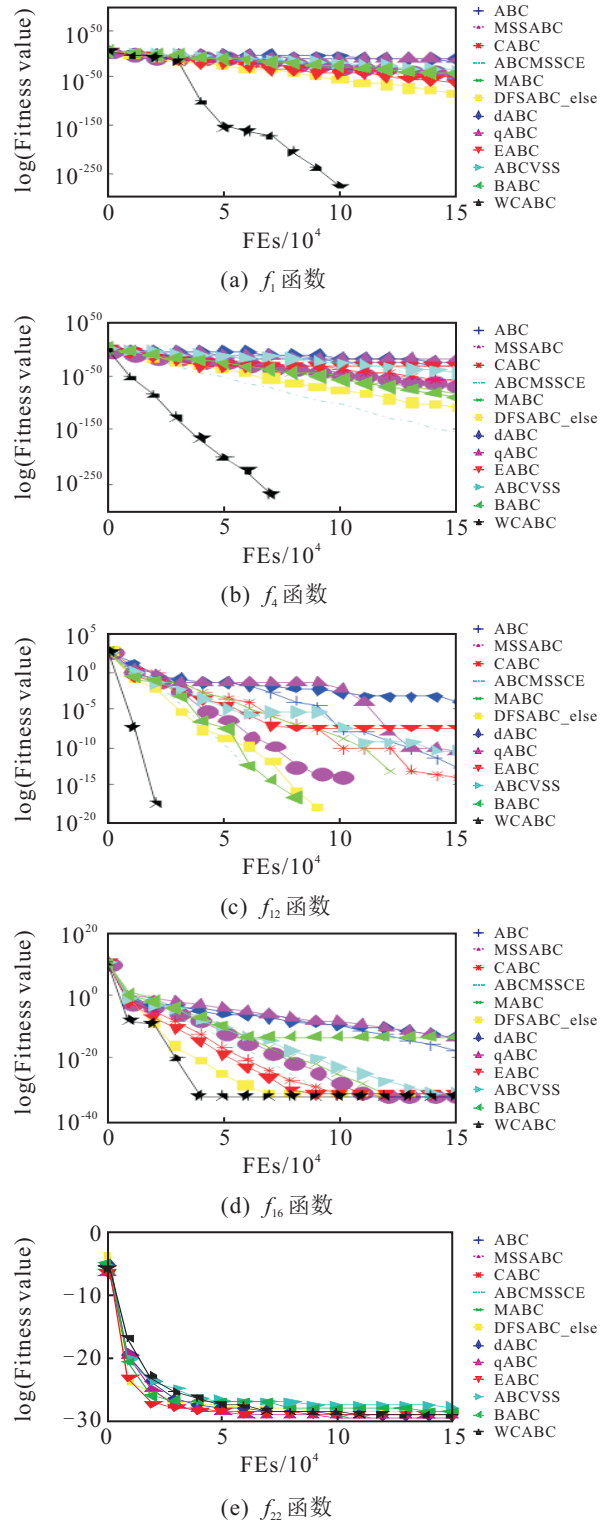


图1 函数收敛曲线

4 结论

本文针对ABC算法在求解高维函数时, 易出现收敛速度慢、易陷入局部最优等缺点, 提出一种新的改进ABC算法, 即WCABC算法. 该算法根据蜜源适应值的优劣赋予各蜜源不同的权重值, 构建一个新的蜜源, 即加权中心. 若加权中心优于当前全局最优, 则取代全局最优, 并通过其引导观察蜂更新, 使算法在收敛速度及跳出局部最优等方面作出有效的改

善. 在22个测试函数以及非参数Friedman检验上对比类似改进算法, 且对WCABC算法耗时进行分析, 实验结果表明, WCABC的寻优性能以及收敛速度具有更好的效果.

参考文献(References)

- [1] Karaboga D. An idea based on honey bee swarm for numerical optimization[R]. Kayseri: Erciyes University, 2005.
- [2] Yang X S. Firefly algorithms for multimodal optimization[C]. Int Symposium on Stochastic Algorithms. Berlin: Springer, 2009, 5792: 169-178.
- [3] 李晓磊, 邵之江, 钱积新. 一种基于动物自治体的寻优模式: 鱼群算法[J]. 系统工程理论与实践, 2002, 22(11): 32-38.
(Li X L, Shao Z J, Qian J X. An optimizing method based on autonomous animats: Fish-swarm algorithm[J]. Systems Engineering — Theory & Practice, 2002, 22(11): 32-38.)
- [4] Kennedy J, Eberhart R. Particle swarm optimization[C]. Proc of IEEE Int Conf on Neural Networks. Piscataway: IEEE, 1995: 1942-1948.
- [5] 王郑拓, 冯振礼, 叶国云, 等. 基于人工蜂群算法的双机器人路径规划分析[J]. 焊接学报, 2015, 36(2): 97-100.
(Wang Z T, Feng Z L, Ye G Y, et al. Analysis of dual-robot path planning based on artificial bee colony algorithm[J]. Trans of the China Welding Institution, 2015, 36(2): 97-100.)
- [6] Bose A, Mali K. Fuzzy-based artificial bee colony optimization for gray image segmentation[J]. Signal, Image and Video Processing, 2016, 10(6): 1-8.
- [7] Wang H, Wu Z, Rahnamayan S, et al. Multi-strategy ensemble artificial bee colony algorithm[J]. Information Sciences, 2014, 279: 587-603.
- [8] 王志刚, 王明刚. 基于符号函数的多搜索策略人工蜂群算法[J]. 控制与决策, 2016, 31(11): 2037-2044.
(Wang Z G, Wang M G. Multi-search strategy of artificial bee colony algorithm based on symbolic function[J]. Control and Decision, 2016, 31(11): 2037-2044.)
- [9] Karaboga D, Gorkemli B. A quick artificial bee colony (qABC) algorithm and its performance on optimization problems[J]. Applied Soft Computing, 2014, 23: 227-238.
- [10] Kiran M S, Findik O. A directed artificial bee colony algorithm[J]. Applied Soft Computing, 2015, 26: 454-462.
- [11] Gao W F, Liu S Y. A modified artificial bee colony algorithm[J]. Computers & Operations Research, 2012, 39(3): 687-697.
- [12] 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.
- [13] Bao L, Zeng J. Comparison and analysis of the selection mechanism in the artificial bee colony algorithm[C]. Proc of IEEE Int Conf on Hybrid Intelligent Systems. Piscataway: IEEE, 2009: 411-416.
- [14] 王志刚, 尚旭东, 夏慧明, 等. 多搜索策略协同进化的人工蜂群算法[J]. 控制与决策, 2018, 33(2): 235-241.
(Wang Z G, Shang X D, Xia H M, et al. Artificial bee colony algorithm with multi-search strategy cooperative evolutionary[J]. Control and Decision, 2018, 33(2): 235-241.)
- [15] Zhu G, Kwong S. Gbest-guided artificial bee colony algorithm for numerical function optimization[J]. Applied Mathematics & Computation, 2010, 217(7): 3166-3173.
- [16] 李冰, 孙辉, 赵嘉, 等. 异维学习人工蜂群算法[J]. 计算机应用研究, 2016, 33(4): 1028-1033.
(Li B, Sun H, Zhao J, et al. Artificial bee colony algorithm with different dimensional learning[J]. Application Research of Computers, 2016, 33(4): 1028-1033.)
- [17] Huo Y, Zhuang Y, Gu J, et al. Elite-guided multi-objective artificial bee colony algorithm[J]. Applied Soft Computing, 2015, 32: 199-210.
- [18] Gao W F, Huang L L, Wang J, et al. Enhanced artificial bee colony algorithm through differential evolution[J]. Applied Soft Computing, 2016, 48: 137-150.
- [19] 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.
- [20] Cui L Z, Li G H, Lin Q Z, et al. A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation[J]. Information Sciences, 2016(367/368): 1012-1044.
- [21] Gao W F, Liu S Y, Huang L L. Enhancing artificial bee colony algorithm using more information based search equations[J]. Information Sciences, 2014, 270(1): 112-133.
- [22] 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: 140-157.
- [23] Gao W F, Chan F T S, Huang L, et al. Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood[J]. Information Sciences, 2015, 316: 180-200.

作者简介

孙辉(1959—), 男, 教授, 从事群智能算法、Rough集与粒计算等研究, E-mail: sun_hui2006@163.com;

谢海华(1994—), 男, 硕士生, 从事群智能优化算法的研究, E-mail: 1332729941@qq.com;

赵嘉(1981—), 男, 教授, 博士, 从事智能信息处理等研究, E-mail: zhao925@163.com;

邓志诚(1995—), 男, 硕士生, 从事群智能优化算法的研究, E-mail: 1125566982@.com.

(责任编辑: 郑晓蕾)