Applied Soft Computing 48 (2016) 230–239

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

On the convergence of multivariant optimization algorithm

Baolei Li[a,b], Jianhua Chen[b], Xinling Shi[b,*], Yufeng Zhang[b], Danjv Lv[b], Lanjuan Liu[b], Qinhu Zhang[b]

[a] *Oil Equipment Intelligent Control Engineering Laboratory of Henan Province, Physics and Electronic Engineering College, Nanyang Normal University, Nanyang 473061, China*

[b] *Department of Electronic Engineering, Information School, Yunnan University, Kunming 650091, Yunnan, China*

ARTICLE INFO

Article history:
Received 19 April 2014
Received in revised form
18 December 2015
Accepted 1 July 2016
Available online 20 July 2016

Keywords:
Multivariant optimization algorithm
Convergence analysis
Multivariant search groups
Global optimization

ABSTRACT

In this paper, we introduce a new global optimization method and study its global convergence property through theoretical and experimental approaches. The proposed method is named as multivariant optimization algorithm (MOA) because the intelligent searchers, which are called as atoms, not only are divided into multiple subgroups but also are variant in responsibility. That is, global atoms explore the whole solution space in the hope of finding potential areas where local atoms start the local exploitation. The proposed method is characterized by two important features. On one hand, global atoms do the global exploration in each loop to jump out from local traps. On the other hand, global and local atoms conduct the global exploration and the local exploitation according to their own responsibility, respectively. These features contribute to increasing the chance of converging to the global best. To study the convergence property of MOA, we carried out the convergence analysis, numerical optimization experiments and the shortest path planning experiments. And the results demonstrate that MOA is globally convergent and superior to the compared methods in the global convergence accuracy and probability in solving complex challenging problems which have one or more features such as deceptiveness, randomly located optimum, asymmetry or multiple traps.

1. Introduction

Global optimization algorithms are designed to deal with optimization problems with multiple local extremal solutions [1]. In this study, we address the solution of the global optimization problem $\min f(\vec{x})$ where $\vec{x} \in \mathbb{R}^N, f : \mathbb{R}^N \to \mathbb{R}$ is a given objective function, N is the dimension of a problem. A global optimization algorithm is utilised to seek a solution $\vec{x}^* \in \mathbf{S} \subseteq \mathbb{R}^N$ such that $f(\vec{x}^*) \leq f(\vec{x}), \forall \vec{x} \in \mathbf{S}$, where the solution space $\mathbf{S}$ is some region of $\mathbb{R}^N$. Such a solution $\vec{x}^*$ is called a global minimum or a global optimum. A solution $\vec{x}'$ is a local minimum (trap) in a local neighborhood $\mathbf{S}_0 \subset \mathbf{S}$ if $f(\vec{x}') \leq f(\vec{x})$, $\forall \vec{x} \in \mathbf{S}_0$.

The "basic" versions of particle swarm optimization (PSO), genetic algorithm (GA), differential evolution (DE) and firefly algorithm (FA) [2] are widely regarded as classical global optimization algorithms, since they have many advantages such as briefness, compact and fast convergence speed [3,4]. A compact differential evolution (cDE) algorithm ne-cDE was proposed to efficiently perform an optimization process despite a limited memory requirement in [5]. However, a high likelihood of being trapped into local optima limits their performance [6]. The proper control of the global exploration and the local exploitation is critical to the global convergence probability of a global optimization algorithm [7]. However, there exists a contradiction between the global exploration and the local exploitation because a single swarm is responsible for both the global exploration and the local exploitation, in these "basic" algorithms [8]. As a result, a user has to balance the contradiction carefully to achieve a good performance [9].

Focusing on increasing the probability of locating the global optimum among numbers of local traps, different strategies were proposed and discussed. Most studies address the performance improvement of PSO, GA or DE through endowing with auxiliary local search, parameter adaptation or scalability strategies, which are briefly reviewed below. *(1) Auxiliary local search:* Dynamic multi-swarm PSO with harmony search (DMS-PSO-HS) is developed through combining the exploration capabilities of the dynamic multi-swarm particle swarm optimizer (DMS-PSO) with the stochastic exploitation of the harmony search (HS) algorithm [10]. A probabilistic memetic framework (APrMF) is a probabilistic memetic algorithm, which is able to analyze the probability of evolution or individual learning [11]. Global and local real-coded GA (HRCGA) is proposed by means of the parent-centric crossover

[*] Corresponding author.
E-mail address: xlshi@ynu.edu.cn (X. Shi).

http://dx.doi.org/10.1016/j.asoc.2016.07.001
1568-4946/© 2016 Elsevier B.V. All rights reserved.

operators [12]. *(2) Parameter adaptation:* A new PSO version with adaptive ω, η_1, and η_2, called adaptive PSO(APSO), was proposed by Zhan et al., recently [13]. In APSO, four evolutionary states, including exploitation, exploration, convergence, and jumping out, are defined. Self-learning PSO (SLPSO) is an improved version of APSO. In SLPSO, each particle has a set of four strategies to cope with different situations in the search space [14]. By considering a time-varying population topology, the velocity update mechanism in fully informed PSO [15], and a decreasing inertia weight, Frankenstein's PSO (FPSO) was proposed in [16]. The covariance matrix adaptation evolution strategy algorithm (G-CMA-ES) employs a restart mechanism and an increasing population size strategy [17]. J-adaptive differential evolution (JaDE) is proposed by implementing a new mutation strategy "DE/current-to-pbest" with optional external archive and updating control parameters in an adaptive manner [18]. By means of learning from their previous experiences in generating promising solutions, a self-adaptive differential evolution (SaDE) algorithm, in which both trial vector generation strategies and their associated control parameter values are gradually self-adapted, was proposed [19]. *(3) Scalability:* comprehensive learning particle swarm algorithm (CLPSO) was proposed by using a novel learning strategy whereby all other particles' historical best information is used to update a particle's velocity [20].

These improved methods, have shown a superiority over the "basic" algorithms in solving problems with lots of local traps [21–23,10]. However, the attractiveness in these improved methods may mislead intelligent searchers to move into a local optimum when solving deceptive functions. The reason is that the global minimum lies in a very narrow basin of attraction and at the same time there exists a strong local minimum with a wide basin of attraction in deceptive functions such as the Damavandi function [24]. So, these improved methods have difficulty in locating the global minimum of deceptive functions.

In this paper, a new "basic" heuristic search framework named multivariant optimization algorithm (MOA) is introduced to increase the global convergence probability in solving complex challenging problems. In MOA, the intelligent searchers (named as atoms) are divided into two categories according to their different responsibilities. They collaborate to search the solution space based on the historical searching information which is obtained by atoms and managed by a data structure. The data structure remembers and shares the historical useful searching information selectively. In each loop of MOA, a group of global atoms explore the whole solution space to locate areas which are more potential than those remembered by the data structure. Then, a group of local atoms exploit each potential area remembered by the data structure for a local refinement. After enough numbers of iteration, the global optimum is recorded in the data structure. From the brief description of MOA, it can be seen that a feature of this proposed method is that the global exploration is executed in each loop, which lessens the probability of being trapped into the local optima. Another is that multivariant search groups carry out the global exploration and the local exploitation respectively, which settles the contradiction between the global exploration and the local exploitation. These features make it a well-suited approach for solving global optimization problems with multiple local optima.

The purpose of this work is twofold. First, a new "basic" stochastic heuristic global optimization method was proposed. Further, we study the global convergence property of the proposed method through the convergence analysis, numerical optimization experiments and the shortest path planning experiments.

In the following sections, we describe the MOA method and prove that MOA is globally convergent. Then, MOA is compared with several state-of-the-art algorithms on twenty-three complex benchmark functions. The results suggest that MOA has a better performance in the global convergence accuracy and probability than the compared methods in solving complex challenging problems which are characterised by deceptiveness, randomly located optimum, asymmetry or multiple traps. Finally, MOA is used to solve the shortest path planning problems to assess the convergence property of MOA in application oriented problems.

2. Multivariant optimization algorithm

In this section, we present the MOA method for solving global optimization problems, introduce the data structure used to manage the communication and cooperation among multivariant search groups and describe the search strategy of MOA.

Without loss of generality, we take the solution space $\mathbf{S}$ as the hyperrectangle $\mathbf{S} = \{\vec{x} = (x_1, \ldots, x_N) | \underset{i}{\min} \le x_i \le \underset{i}{\max}, (i = 1, \ldots, N)\}$ where $\underset{i}{\min}$ and $\underset{i}{\max}$ are the lower and upper bounds of the ith dimension of the solution space, respectively. The global optimization problem considered in this paper is: find $\vec{x}^* = argmin\{f(\vec{x}) | \vec{x} \in \mathbf{S}\}$, where f is a given objective function.

In MOA, intelligent searchers called atoms search the solution space through cooperating with each other based on a data structure illustrated in Fig. 1. To simplify the description, we name the horizontal and vertical sorted doubly linked list as the queue and stack, respectively.

MOA is a stochastic heuristic optimization algorithm, where each iteration consists of two phases: a global exploration phase and a local exploitation phase. In the global exploration phase, global atoms explore the whole solution space randomly to produce a diverse set of potential areas. In the local exploitation phase, local atoms exploit each potential areas gained in the previous search for local refinements. The pseudo-code of MOA is outlined in Table 1, where

- *TL* is a temporary list used to record the newly generated global atoms and their fitness values;
- *TN* is a temporary node used to record a newly generated local atoms and its fitness value;
- *DS* is the used date structure;
- $DS(i, j)$ is the jth node from the top in the ith stack;
- $DS(i, j)$.A is an atom recorded in $DS(i, j)$ and $DS(i, j)$.Fv is its fitness value;

In step 1, new global search atoms explore the whole solution space according to:

$$atom_g = \{unifrnd(\underset{1}{\min}, \underset{1}{\max}), \ldots, unifrnd(\underset{N}{\min}, \underset{N}{\max})\} \qquad (1)$$

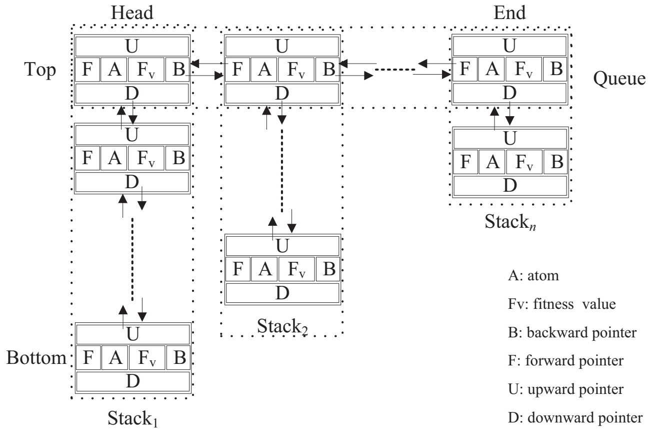


Fig. 1. Data structure of multivariant optimization algorithm.

where the function *unifrnd*($\underset{i}{\min}$, $\underset{i}{\max}$), ($i = 1, \ldots, N$) returns a random number uniformly distributed on the interval from $\underset{i}{\min}$ to $\underset{i}{\max}$. Then, their fitness values are evaluated by using the objective function (line 5). To carry out an efficient local exploitation, we update the queue in step 2 according to two rulers: one is that only the better atoms are recorded in the queue, the other is *DS*(1, $i-1$).Fv≤*DS*(1, i).Fv≤*DS*(1, $i+1$).Fv. In step 3, local atoms exploit each potential area centered at a global atom in the queue (line 10). A local search atom $atom_l$ in the neighborhood of the global atom $atom_g$ is generated according to:

$$atom_l = atom_g + r \times \{h_1, \ldots, h_N\} \tag{2}$$

where $h_i (i = 1, 2, \ldots, N)$ are random numbers uniformly distributed on [−1,1], $atom_g$ and r are the center and the radius of the neighborhood, respectively. After the evaluation (line 11), the new local atom is compared against the corresponding historical atom (line 12). A new atom will be remembered (lines 13–21), if its fitness value is smaller than that of the historical atom recorded in the bottom of its corresponding stack.

3. Convergence analysis

We analyze the global convergence of MOA based on a global optimization problem where **S** is a finite discrete solution space. Our objective is to locate a global optimal solution $\vec{x}^* \in \mathbf{S}$ which minimizes the objective function f on **S**. The search process of MOA can be described as a stochastic process $\{\vec{X}(t), t \geq 0\}$ on the finite solution space **S**. $\vec{X}(t) = (\vec{x}_{t,1}, \vec{x}_{t,2}, \ldots, \vec{x}_{t,n})(\vec{x}_{t,i} \in \mathbf{S}, n < \infty)$ represents the tth generation atoms in the structure whose state space is $\vec{E} = |\mathbf{S}|^n$ where n is the number of atoms. Assuming the global optima set is $\vec{E}^* = \{\vec{x}^* | \nexists \vec{x} \neq \vec{x}^*, f(\vec{x}) \leq f(\vec{x}^*)\}$

Theorem 1. *The search process* $\{\vec{X}(t), t \geq 0\}$ *of MOA is a finite homogeneous Markov chain.*

Proof. At the tth iteration, the structure has a state $\vec{X}(t)$ and atoms search the solution space to make a transition to the next state $\vec{X}(t+1)$ according to a probability $P(\vec{X}(t+1)|\vec{X}(t))$. The structure transfers to the next state $\vec{X}(t+1)$ with a probability which has nothing to do with its previous history but $\vec{X}(t)$. In other words, the stochastic process $\{\vec{X}(t), t \geq 0\}$ has the Markov property that $P(\vec{X}(t+1)|\vec{X}(t)) = P(\vec{X}(t+1)|\vec{X}(t), \vec{X}(t-1), \ldots, \vec{X}(0))$. Thus, $\{\vec{X}(t), t \geq 0\}$ is a Markov process.

Table 1
The pseudo-code of multivariant optimization algorithm.

```
1: Initialize randomly the global atoms, set iteration=1;
2: while the stopping criterion is not satisfied do
3:   for i=1:length of queue % step 1: global exploration
4:     TL(i).A←generate a global atom using Eq. (1);
5:     TL(i).Fv←evaluate TL(i).A according to the objective function;
6:   end for
7:   Update the queue; % step 2
8:   for i=1:length of queue
9:     for j=1:depth of the ith stack % step 3: local exploitation
10:        TN.A←generate a local atom using Eq.(2), in the area centers on DS(1,
   i).A;
11:        TN.Fv←evaluate TN.A according to the objective function;
12:        if DS(i,depth of the ith stack).Fv>TN.Fv
13:          for k=depth of the ith stack:2
14:            if DS(i, k).Fv>TN.Fv
15:              switch the places of the nodes DS(i, k) and DS(i, k − 1);
16:            else
17:              DS(i, k) ← TN %remember the historical information;
18:              break;
19:            end if
20:          end for
21:        end if
22:      end for
23:    end for
24:    set iteration=iteration+1;
25: end while
```

Because the solution space **S** is discrete and finite, the structure state space $|\mathbf{S}|^n$ is also discrete and finite. What is more, the transfer probability $P(\vec{X}(t+1)|\vec{X}(t))(t > 0)$ is independent of t:

$$P(\vec{X}(t+1) = \vec{X}_i | \vec{X}(t) = \vec{X}_j) = \zeta \quad (\zeta > 0) \tag{3}$$

where $\vec{X}_i$, $\vec{X}_j$ are two arbitrary states. Thus, the search process of MOA can be described sufficiently by a discrete finite homogeneous Markov chain. □

Theorem 2. *The time sequence of the number of optima in the structure* $\{F(\vec{X}(t)), t \geq 0\}$ *is a monotonically non-decreasing sequence as*

$$P(F(\vec{X}(t+1)) < m | F(\vec{X}(t) = m)) = 0 \quad (\forall t \geq 0, m > 0) \tag{4}$$

where $F(\vec{X}(t)) = |\vec{X}(t) \cap \vec{E}^*|$ *represents the numbers of global optima in the structure,*

Proof. In MOA, the worse atoms are abandoned but the better atoms are saved in the structure, so an atom in the structure will not be abandoned unless a new atom with better fitness is located. The best atom standing for the optimum is impossible to be abandoned so the number of optima in the structure will not decrease.□

Theorem 3. *The MOA method has a positive probability of locating more than one global optimum at any time during the search process*

$$P(F(\vec{X}(t+1)) > 0 | F(\vec{X}(t) = 0)) > 0 (\forall t \geq 0) \tag{5}$$

Proof. According to (3) the transition probability between two arbitrary states $\vec{X}_i$, $\vec{X}_j$ is positive, given $F(\vec{X}(t) = \vec{X}_i) = 0$, the probability $P(F(\vec{X}(t+1) = \vec{X}_j) > 0 | F(\vec{X}(t) = \vec{X}_i) = 0)$, $(\forall t \geq 0)$ is also positive. In MOA, a newly generated atom has the probability to be any solution because search atoms are generated in the whole solution space. So, for an atom the probability of being global optimum is larger than 0. Thus, the number of optima in the new generation atoms is probable to be larger than 0 given that the number of optima in the previous generation atoms is 0.□

Theorem 4. *MOA converges to more than one global optimal solution with probability 1:*

$$\lim_{t \to \infty} P(F(\vec{X}(t)) > 0) = 1 \tag{6}$$

Proof. Based on theorem 1, we define the probability of i global optima existing in the structure at the tth iteration as $P_i(t) = P(F(\vec{X}(t)) = i)$. From the Bayesian conditional probability

$$\begin{aligned} P_0(t+1) = {} & P(F(\vec{X}(t+1)) = 0 | F(\vec{X}(t)) = 0) \times P(F(\vec{X}(t)) = 0) \\ & + P(F(\vec{X}(t+1)) = 0 | F(\vec{X}(t)) \neq 0) \times P(F(\vec{X}(t)) \neq 0) \end{aligned}$$

and theorem 2 we have that $P(F(\vec{X}(t+1)) = 0 | F(\vec{X}(t)) \neq 0) = 0$ then

$$P_0(t+1) = P(F(\vec{X}(t+1)) = 0 | F(\vec{X}(t)) = 0) \times P_0(t)$$

From theorem 3 we know that $P(F(\vec{X}(t+1)) > 0 | F(\vec{X}(t)) = 0) > 0$, let $\xi = \min(P(F(\vec{X}(t+1)) > 0 | F(\vec{X}(t)) = 0))$, $t = 0, 1, \ldots$ then

$$P(F(\vec{X}(t+1)) > 0 | F(\vec{X}(t)) = 0) \geq \xi > 0 \tag{7}$$

From (7) we can get that

$$\begin{aligned} P(F(\vec{X}(t+1)) & = 0 | F(\vec{X}(t)) = 0) = 1 - P(F(\vec{X}(t+1)) \neq 0 | F(\vec{X}(t)) = 0) \\ & = 1 - P(F(\vec{X}(t+1)) > 0 | F(\vec{X}(t)) = 0) \leq 1 - \xi < 1 (t = 0, 1, \ldots) \end{aligned}$$

So

$$0 \leq P_0(t+1) \leq (1-\xi) P_0(t) \leq \cdots \leq (1-\xi)^{t+1} P_0(0)$$

Table 2
Dimensions, search ranges, and global optimum values of test functions.

f	Function name	Dimensions	Search ranges	Global optimum values
F1	Schwefel [27]	2	$[-500, 500]^2$	0
F2	Levy5	2	$[-10, 10]^2$	−176.1375
F3	Langermann [28]	2	$[0, 10]^2$	−5.1621
F4	Xin-She Yang [29]	2	$[-20, 20]^2$	−1
F5	Egg Holder [30]	2	$[-513, 513]^2$	−959.64
F6	Damavandi [24]	2	$[0, 14]^2$	0
F7	CF1 [25]	10	$[-5, 5]^{10}$	0
F8	CF2 [25]	10	$[-5, 5]^{10}$	0
F9	CF3 [25]	10	$[-5, 5]^{10}$	0
F10	CF4 [25]	10	$[-5, 5]^{10}$	0
F11	CF5 [25]	10	$[-5, 5]^{10}$	0
F12	CF6 [25]	10	$[-5, 5]^{10}$	0
F13	CEC05(F1+450)[26]	30	$[-100, 100]^{30}$	0
F14	CEC05(F2+450)[26]	30	$[-100, 100]^{30}$	0
F15	CEC05(F9+330)[26]	30	$[-5, 5]^{30}$	0
F16	CEC05(F10+330)[26]	30	$[-5, 5]^{30}$	0
F17	CEC05(F11−90)[26]	30	$[-0.5, 0.5]^{30}$	0
F18	CEC05(F15−120)[26]	30	$[-5, 5]^{30}$	0
F19	CEC05(F16−120)[26]	30	$[-5, 5]^{30}$	0
F20	CEC05(F17−120)[26]	30	$[-5, 5]^{30}$	0
F21	CEC05(F18−10)[26]	30	$[-5, 5]^{30}$	0
F22	CEC05(F19−10)[26]	30	$[-5, 5]^{30}$	0
F23	CEC05(F20−10)[26]	30	$[-5, 5]^{30}$	0

Because $\lim_{t\to\infty}(1-\xi)^{t+1} = 0$, and $0 \le P_0(0) \le 1$ so when $t \to \infty$

$$0 \le P_0(t+1) \le (1-\xi)^{t+1} P_0(0) = 0$$

Thus, $\lim_{t\to\infty} P_0(t+1) = 0$

From the above, we can get that

$$\begin{aligned}\lim_{t\to\infty} P(F(\vec{X}(t+1)) \ge 1) &= 1 - \lim_{t\to\infty} P(F(\vec{X}(t+1)) = 0) \\ &= 1 - \lim_{t\to\infty} P_0(t+1) = 1\end{aligned}$$

Thus MOA converges to more than one global optimal solution with probability 1.□

4. Numerical experiments

We compared MOA with several state-of-the-art algorithms to demonstrate the superior global convergence property of MOA in solving complex challenging problems.

4.1. Benchmark functions

We wish to study the convergence property of MOA when dealing with complex multimodal problems where the global optimum is hard to find. Firstly, six two-dimensional (2-D) problems and six ten-dimensional (10-D) novel composition problems from [25] are selected for comparison. These composition functions are randomly located, asymmetrical and multimodal problems. Although their dimension is low, they are good at trapping many heuristic optimization algorithms into local optima. To draw more robust conclusions, experiments are carried out based on five unimodal or simple multimodal 30-dimensional problems and six complex multimodal 30-dimensional problems, which were taken from the CEC'2005 special session on real-parameter optimization [26]. F13–F23 were added by some bias values so that their best values are all zero. An overview of these functions is presented in Table 2, while mathematical descriptions are presented in Appendix.

4.2. Parameter settings

Fourteen algorithms are chosen to compare with MOA, because they are widely known as classic "basic" algorithms or are designed for global optimization of complex problems or well-known improved algorithms. The GA, PSO-w, FA, CLPSO, DMS-PSO-HS and MOA algorithms and their parameter settings are listed as follows:

- GA with one elitist [31]: eighty percent of the new generation is produced by roulette selection and twenty percent is produced by crossover, uniformly distributed mutation is performed with probability 0.01;
- PSO with inertia weight (PSO-w) [32]: the inertia weight decreases from 1.4 to 0.5, acceleration constants $c_1 = c_2 = 1.49445$;
- FA [2]: The randomization parameter α is 0.2, the attractiveness β_0 is 1, the light absorption coefficient γ is 1;
- CLPSO [20]: the inertia weight w decreases from 0.9 to 0.4, acceleration constants $c = 1.49445$, refreshing gap $m = 7$;
- DMS-PSO-HS [10]: The best parameters suggested in [10] are as follows: $w = 0.729$, $c_1 = c_2 = 1.49445$, 10 sub-swarms, 5 particles in each sub-swarm, regrouping period $R = 10$, harmony memory considering rate $HMCR = 0.98$, the pitch adjusting rate $PAR(t)$ and the band $bw(t)$ width for generation t are calculated by:

$$PAR(t) = 0.01 + \frac{0.99 - 0.01}{\max_g} \times t$$

$$bw(t) = (0.05 \times (UB - LB))e^{\frac{0.0001}{(0.05 \times (UB - LB))} \times t}$$

where $\max_g$ is the maximum number of generations, UB and LB are the upper and lower bounds of the search range;
- MOA: The radius of the local neighborhood r is set at 0.05 times the absolute difference between the high value and the low value of the search ranges according to our experiences. An upper triangular structure, where the depth of the ith stack is $QL - i + 1$ for a queue with QL nodes, is used. Thus, the population contains QL global atoms and $0.5QL(QL+1)$ local atoms.

To compare fairly, the population size of MOA is set as close as that of the compared methods. When solving the two-dimensional

Table 3
Comparison of the average final fitness values on low-dimensional numerical optimization between the control algorithm MOA and the rest of algorithms selected in the experimental study. The signs in the parentheses are used in the computation of the multiple comparison sign test.

f	MOA	GA	PSO-w	FA	CLPSO	DMS-PSO-HS
F1	3E−05	161(+)	92.8(+)	3.35(+)	7.9(+)	**2E−05**(−)
F2	−**176**	−127(+)	−166(+)	−174(+)	−**176**(=)	−174(+)
F3	−**5.16**	−3.61(+)	−4.4(+)	−5.12(+)	−**5.16**(=)	−4.45(+)
F4	−**0.97**	3E−04(+)	−0.86(+)	−0.11(+)	−0.33(+)	1E−05(+)
F5	−**957**	−711(+)	−782(+)	−932(+)	−939(+)	−848(+)
F6	**1.63**	1.93(+)	1.93(+)	1.70(+)	2.00(+)	2.00(+)
F7	**7.91**	26.66(+)	93.33(+)	199.45(+)	31.81(+)	20(+)
F8	**58.1**	99.59(+)	122.02(+)	155.93(+)	84.14(+)	69.79(+)
F9	355.82	211.36(+)	266.11(+)	466.03(+)	201.05(+)	**126.64**(−)
F10	436	371.21(+)	392.62(+)	518.85(+)	366.93(+)	**259.55**(−)
F11	**15.55**	74.45(+)	99.25(+)	164.78(+)	44.49(+)	27.58(+)
F12	**480.06**	775.66(+)	802.11(+)	563.10(+)	532.52(+)	607.47(+)
No. of pluses		12	12	11	10	9
No. of minuses		0	0	0	0	3
r_j (significance level: 5%)		1	1	1	1	1

problems, the population size is set at 14 and the maximum fitness evaluations (FE_s) is set at 14000. When solving the ten-dimensional problems, the population size is set at 65, and the maximum of the number of fitness evaluations is set at 29,250 according to [20]. When solving the thirty-dimensional problems, the population size is set at 119 which is in close proximity to that in [5], and the maximum of the number of fitness evaluations is set at 150,000 according to [5]. All experiments were run 30 times per algorithm for per problem.

4.3. Convergence performance measures

We use the following two criteria measured over 30 runs to compare the convergence property of different algorithms.

1. The accuracy of convergence. The mean of the final fitness values in the 30 independent runs are calculated and recorded to analyze the convergence accuracy of the compared method. Note that a smaller average value suggests a higher convergence accuracy of the algorithm.

2. The probability of convergence. Convergence probability is the percentage of globally convergent runs where the fitness value of the best atom achieves a given fitness-threshold value (f_t).

The multiple comparison sign test has also been applied according to the description given in [33–35] to compare the performance of the control algorithm MOA with that of the other selected algorithms. A level of significance 5% is chosen. The +, − and = symbols represent win, lose and tied respectively.

The comparison operations in the management of structure table increase the computational complexity compared with PSO which are treated as the simplest optimization solver universally. However, a comparison operation needs less clock cycles than a multiply operation. Different with PSO, only comparison operations are used in MOA. Thus, it is unfair to compare the computational complexities of the "basic" MOA through the big O notation when the iteration number is small. Because, the computational complexity is calculated by counting the number of basic computer steps without considering the differences between different steps in the big O notation method [36]. What is more, most real life optimization problems require intensive and time consuming simulations for every function evaluation; the time spent by the optimization solver itself doing its calculations simply disappears in front of the real process simulation [37]. Hence the algorithm-related computation complexities are not compared.

4.4. Results and discussions on low-dimensional numerical optimization

The experimental results with respect to the convergence accuracy and the convergence probability are presented in Tables 3 and 4, respectively.

4.4.1. The accuracy of convergence on low-dimensional numerical optimization

As listed in Table 3, MOA can earn the smallest mean values compared with the other peer methods except on F1, F2, F9 and F10. DMS-PSO-HS achieve the best results on F1, F9 and F10. The reason lies in that the single group algorithms including GA, PSO-w, FA and CLPSO are easily be trapped into local optima of these

Table 4
Comparison of convergence probability on low-dimensional numerical optimization between the control algorithm MOA and the rest of algorithms selected in the experimental study. The signs in the parentheses are used in the computation of the multiple comparison sign test.

f	f_t	MOA	GA	PSO-w	FA	CLPSO	DMS-PSO-HS
F1	0.01	**1**	0.23(+)	0.33(+)	0(+)	0.93(+)	**1**(=)
F2	−175	**1**	0.5(+)	0.73(+)	0.27(+)	**1**(=)	0.93(+)
F3	−5	**1**	0.37(+)	0.67(+)	**1**(=)	**1**(=)	0.67(+)
F4	−0.5	**0.97**	0(+)	0.87(+)	0.13(+)	0.03(+)	0(+)
F5	−950	**0.87**	0.03(+)	0.2(+)	0.13(+)	0.73(+)	0.1(+)
F6	0.1	**0.13**	0.03(+)	0.03(+)	0(+)	0(+)	0(+)
F7	10	**0.90**	0.77(+)	0.47(+)	0(+)	0.10(+)	0.80(+)
F8	58	**0.50**	0.47(+)	0.27(+)	0(+)	0.10(+)	0.40(+)
F9	126	0	0.07(−)	0.03(−)	0(=)	0(=)	**0.47**(−)
F10	260	0	0(=)	0.17(−)	0(=)	0(=)	**0.47**(−)
F11	18	**0.90**	0.33(+)	0.57(+)	0(+)	0(+)	0.73(+)
F12	480	**0.30**	0(+)	0(+)	0(+)	0.03(=)	0(+)
No. of pluses			10	11	9	7	9
No. of minuses			1	0	0	0	2
r_j (significance level: 5%)			1	1	1	1	1

deceptive functions, it's most likely that a part of their final obtained optima are local optima.

As shown in the results of the Multiple Comparison Sign test in Table 3, the convergence accuracy of MOA is significantly better than that of GA, PSO-w, FA and CLPSO. MOA outperforms DMS-PSO-HS on most problems. This proves the proposed local exploitation in MOA, where the local group with the largest population is allotted to the most potential area, can help the algorithm exploite the local optimum areas at a high level.

4.4.2. *The probability of convergence on low-dimensional numerical optimization*

The convergence probabilities for all compared methods are presented in Table 4. It can be seen that the MOA method performs better on most of the test functions (higher convergence probability), which is consistent with the convergence analysis in Section 3. Especially on test functions F3, F4, F7, F8, F11 and F12, there is a sharp increase in the convergence probability compared with other methods.

As shown in the results of the multiple comparison sign test in Table 4, the convergence probability of MOA is significantly better than that of GA, PSO-w, FA and CLPSO. The better performance of MOA is due to the division of labor between global exploration and local exploitation. The global exploration provides MOA the capacity of escaping from local optima and locating new potential area where the global optimum may lies in. The local exploitation guarantees each local group to converge to the optimum in its area. To sum up, MOA has the ability of finding the global optimum with a higher probability compared with other methods on most functions.

To sum up, the global convergence properties of these peer methods vary on different problems. However, a prominent result is that MOA outperforms the other compared methods in the convergence accuracy and probability on most of the test functions which are characterized by deceptiveness. It is necessary to notice that the test functions in this experiment are all deceptive, which may contribute to emphasizing the global convergence ability of MOA.

4.5. *Results and discussions on 30-dimensional numerical optimization*

The experimental results on 30-dimensional numerical optimization with respect to the convergence accuracy are presented in Tables 5. The results of the SLPSO, APSO, AprMF, FPSO, G-CMA-ES and HRCGA were taken from [14], and the results of the JaDE, SaDE and ne-cDE algorithms were taken from [5].

4.5.1. *The accuracy of convergence on 30-dimensional numerical optimization*

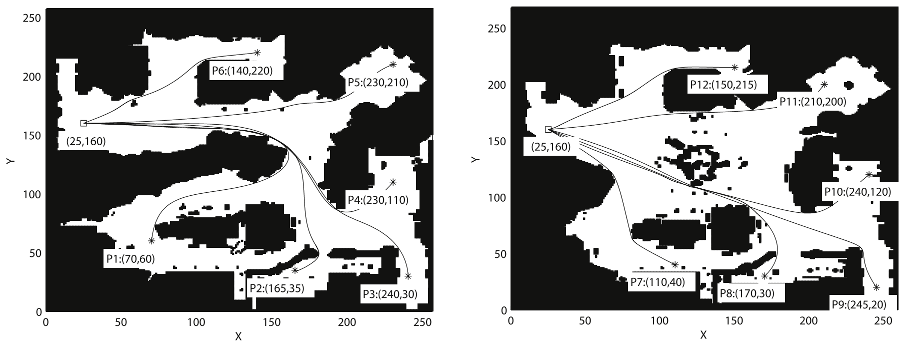
From the results on the unimodal or simple multimodal problems F13–F17 in Table 5, it can be seen that MOA is outperformed by the other compared algorithms on most of these problems. Although the dimensions of these simple problems are high, all these compared methods can easily locate the area of global optimum. It is normal that the convergence accuracies of the SLPSO, APSO, AprMF, JaDE and SaDE algorithms become higher while their searching scopes become smaller along with the iteration number increases. However, MOA can be competitive, at least for some of the considered problems. Clearly, MOA are not expected to outperform modern complex algorithms on simple problems because of the following two reasons. Firstly, MOA is supposed to be a light version, which does not employ any extra complex search component. As a simple framework, MOA uses a rather simple local search strategy, i.e., generating random local atoms in an area with fixed scope. Secondly, MOA is designed for complex challenging problems, for which most optimization technologies become invalid. In this sense, the results on functions F13–F17 should be read in the following way: notwithstanding its significant disadvantage, MOA displays a respectable performance when compared with complex, modern, and computer consuming algorithms. Future studies will consider the integration into the presented compact framework of more advanced local search techniques.

From the results on the complex challenging problems F18–F23 in Table 5, it can be seen that MOA outperforms the other compared algorithms on most of these problems. This fact can be, in our opinion, explained as a consequence of the MOA search logic. More specifically, the global exploration is executed in each loop, which lessens the probability of being trapped into the local optima, and the global atoms and the local atoms carry out the global exploration and the local exploitation respectively, which settles the contradiction between the global exploration and the local exploitation. These features make it a well-suited approach for solving complex challenging problems with multiple local traps.

As shown in the results of the multiple comparison sign test in Table 5, the global convergence accuracy of these peer algorithms vary on different problems with different complexities, and the convergence accuracy of MOA is almost the same as that of the compared algorithms on all of these simple and challenging problems. This result conforms with the "no free lunch theorem". However, a prominent result is that MOA outperforms the other compared

Table 5
Comparison of the average final fitness values on 30-dimensional numerical optimization between the control algorithm MOA and the rest of algorithms selected in the experimental study. The signs in the parentheses are used in the computation of the multiple comparison sign test.

f	MOA	SLPSO	APSO	AprMF	JaDE	SaDE	ne-cDE
F13	0.71	1.00e−13(−)	7.20e−14(−)	0.76(+)	1.54e−13(−)	2.313e−34(−)	1.82e−15(−)
F14	87.35	1.40e−06(−)	0.0182(−)	4.21e+04(+)	1.50e+02(+)	1.527e−01(−)	1.183e+04(+)
F15	6.72	1.17e−13(−)	5.74(−)	3.92(−)	2.03e+01(+)	2.18e+01(+)	1.203e+02(+)
F16	1.21e+02	1.15e+02(−)	1.33e+02(+)	2.31e+02(+)	2.65e+01(−)	3.25e+01(−)	2.20e+02(+)
F17	4.05e+01	3.29e+01(−)	3.04e+01(−)	3.09e+01(−)	2.78e+01(−)	1.66e+01(−)	3.80e+03(+)
f	MOA	SLPSO	APSO	AprMF	FPSO	G-CMA-ES	HRCGA
F18	301	305(+)	363(+)	583(+)	459(+)	200(−)	320(+)
F19	400	420(+)	426(+)	404(+)	549(+)	400(=)	400(=)
F20	512	914(+)	656(+)	393(−)	988(+)	1.5e+03(+)	465(−)
F21	751	1.56e+03(+)	1.71e+03(+)	1.26e+03(+)	837(+)	1.94e+03(+)	400(−)
F22	196	457(+)	312(+)	1(−)	644(+)	659(+)	231(+)
F23	278	300(+)	300(+)	1(−)	620(+)	400(+)	572(+)
No. of minuses		5	4	6	–	–	–
No. of pluses		6	7	5	–	–	–
r_j (significance level: 5%)		1	1	1	–	–	–



(a) Map-den520d

(b) Map-den400d

Fig. 2. Testing environments.

methods in the convergence accuracy on most of these complex challenging functions.

5. The shortest path planning experiments

The MOA, GA, PSO-w, FA, CLPSO and DMS-PSO-HS algorithms are used to solve the shortest path planning problems, to assess the convergence property of MOA in application oriented problems.

5.1. Path planner based on an optimization algorithm

To transfer a path planning task into an optimization problem, the Bezier curve is used to describe a path [38]. The optimization algorithm is employed to locate the position of the optimal control points of the Bezier curve.

A path $P = \{(P_x^c(t), P_y^c(t))|t = 0.001, \ldots, 1.000\}$ reconstructed by a d-dimensional solution $atom = \{P_x^1, P_y^1, \ldots, P_x^{d/2}, P_y^{d/2}\}$ is described by:

$$\begin{cases} P_x^c(t) = P_x^s B_{0,N}(t) + \sum_{i=o}^{d/2} P_x^i B_{i,N}(t) + P_x^e B_{N,N}(t) \\ P_y^c(t) = P_y^s B_{0,N}(t) + \sum_{i=o}^{d/2} P_y^i B_{i,N}(t) + P_y^e B_{N,N}(t) \end{cases}$$

where P_x^s and P_y^s are the x-coordinate and y-coordinate of the origin positions respectively, (P_x^i, P_y^i) represents the coordinate of the ith control point of a Bezier curve, the ending position is determined by P_x^e and P_y^e, N equals $d/2 + 1$, $B_{i,N}(t)$ is a Bernstein polynomial represented by:

$$B_{i,N}(t) = C_N^i t^i (1-t)^{N-i}$$

The fitness value of an atom is defined as follows:

$$f(atom) = f_{len}(P) + \lambda \times N_{im}$$
$$f_{len}(P) = \sum_{k=2}^{1000} \sqrt{\left(P_x^c\left(\frac{k}{1000}\right) - P_x^c\left(\frac{k-1}{1000}\right)\right)^2 + \left(P_y^c\left(\frac{k}{1000}\right) - P_y^c\left(\frac{k-1}{1000}\right)\right)^2} \tag{8}$$

where $f_{len}(P)$ is the length of P. 1000 points on P are used to describe the path. If all points are in negotiable areas, the path could be defined as feasible. Otherwise, penalty will be imposed. The constant value λ is used to balance the proportion of path length and penalty and is set to be 500 in our study. N_{im} is the number of points in impassable areas or out of the map.

5.2. Experimental setup

The den520d map and the den400d map [39] illustrated in Fig. 2 are used as the testing environments. The □ and * marks respectively stand for the origin positions and the destinations, and the Bezier curves stand for paths planned by using MOA. The coordinates of the origin and destination in each path planning task are listed in Table 6.

The parameter settings of the GA, PSO-w, FA, CLPSO, and DMS-PSO-HS algorithms are the same as those introduced in Section 4.2. The population size of each algorithm is set at 65, and all these algorithms terminate when 3250 fitness evaluations are exhausted, because experimental results show that more searching iterations make no contribution to the solution quality of any algorithm.

5.3. Results and discussions on path planning experiments

A run where a feasible path is obtained is regarded as a feasible run. The average path length over feasible runs is used as the index of convergence accuracy. The probability of convergence is indexed by the rate of the feasible runs.The average final path length over feasible runs is calculated by $A_{pl} = \frac{1}{N}\sum_{n=1}^{N} f(\boldsymbol{P}_n^*)$ where $\boldsymbol{P}_n^*$ is the shortest path obtained at the nth feasible run, N is the number of feasible runs. The A_{pl} values are calculated only in feasible runs, so that the very large path length values in infeasible runs have no influence on the A_{pl} value. A small A_{pl} value suggests a good performance on optimality. The Multiple Comparison Sign test [34] is used as a methodology for comparing the perform of the adopted algorithms. The feasible rate is the quotient of the number of feasible runs and the number of total runs. A large feasible rate suggests a good performance on convergence probability.

5.3.1. The accuracy of convergence on path planning problems

The results on average final path length are illustrated in Table 7.The results of the Multiple Comparison Sign test show that MOA significantly outperforms GA, FA and CLPSO on convergence accuracy, but it is hard to say which one is the best one among MOA, PSO-w and DMS-PSO-HS. It's worth mentioning that the statistics results are over feasible runs only, so it is not surprising that PSO-w and DMS-PSO-HS obtained low A_{pl} values, the reasons are as follows: firstly, once a particle in PSO-w moves into the area where the attraction of the global optimum can reach, the swarm will converge to the global optimum as close as possible, because it's search scope shrinking along with the iteration number increases. Secondly, the auxiliary local exploitation technology in DMS-PSO-HS contributes to increase the quality of a intermediate solution. Consistent with the conclusion in the 30-D unimodal or simple

Table 6
Path planning tasks.

No.	Map	Origin	Destination	*No*.	Map	Origin	Destination
P1	den520d	(25,160)	(70,60)	P7	den400d	(25,160)	(110,40)
P2	den520d	(25,160)	(165,35)	P8	den400d	(25,160)	(170,30)
P3	den520d	(25,160)	(240,30)	P9	den400d	(25,160)	(245,20)
P4	den520d	(25,160)	(230,110)	P10	den400d	(25,160)	(240,120)
P5	den520d	(25,160)	(230,210)	P11	den400d	(25,160)	(210,200)
P6	den520d	(25,160)	(140,220)	P12	den400d	(25,160)	(150,215)

Table 7
Comparison of the average final path length over feasible runs between the control algorithm MOA and the rest of algorithms selected in the experimental study. The signs in the parentheses are used in the computation of the multiple comparison sign test.

No.	MOA	GA	PSO-w	FA	CLPSO	DMS-PSO-HS
P1	288.1	316.4(+)	**276.1**(−)	337.2(+)	315.5(+)	285.2(−)
P2	264.6	274.5(+)	**256.6**(−)	286.8(+)	282.6(+)	260.8(−)
P3	293.2	316.1(+)	**281.0**(−)	334.7(+)	324.0(+)	284.8(−)
P4	260.0	274.4(+)	**250.1**(−)	291.9(+)	286.4(+)	258.6(−)
P5	**218.8**	227.5(+)	220.2(+)	250.8(+)	238.0(+)	218.8(=)
P6	141.1	156.5(+)	**133.8**(−)	195.8(+)	190.0(+)	136.5(−)
P7	**158.5**	180.3(+)	166.9(+)	199.5(+)	−(+)	−(+)
P8	**233.3**	276.9(+)	233.6(+)	277.5(+)	−(+)	−(+)
P9	282.4	284.2(+)	277.6(−)	557.2(+)	280.0(−)	**272.3**(−)
P10	**258.8**	419.3(+)	400.8(+)	298.1(+)	−(+)	−(+)
P11	**197.4**	206.5(+)	202.0(+)	223.8(+)	−(+)	−(+)
P12	159.6	161.7(+)	142.7(−)	209.1(+)	−(+)	**141.4**(−)
Number of pluses		12	7	12	11	4
Number of minuses		0	5	0	1	7
r_j(significance level: 5%)		1	1	1	1	1

Table 8
Comparison of feasible rate between the control algorithm MOA and the rest of algorithms selected in the experimental study. The signs in the parentheses are used in the computation of the multiple comparison sign test.

No.	MOA	GA	PSO-w	FA	CLPSO	DMS-PSO-HS
P1	56.7%	36.7%(+)	20.0%(+)	26.7%(+)	36.7%(+)	93.3%(−)
P2	73.3%	26.7%(+)	6.7%(+)	33.3%(+)	40.0%(+)	60.0%(+)
P3	46.7%	3.3%(+)	6.7%(+)	20.0%(+)	3.3%(+)	13.3%(+)
P4	86.7%	33.3%(+)	23.3%(+)	63.3%(+)	43.3%(+)	76.7%(+)
P5	100.0%	90.0%(+)	96.7%(+)	96.7%(+)	100.0%(=)	100.0%(=)
P6	96.7%	53.3%(+)	36.7%(+)	96.7%(=)	53.3%(+)	80.0%(+)
P7	100.0%	100.0%(=)	80.0%(+)	100.0%(=)	0.0%(+)	0.0%(+)
P8	100.0%	70.0%(+)	83.3%(+)	100.0%(=)	0.0%(+)	0.0%(+)
P9	90.0%	13.3%(+)	13.3%(+)	36.7%(+)	90.0%(=)	53.3%(+)
P10	96.7%	53.3%(+)	63.3%(+)	100.0%(−)	0.0%(+)	0.0%(+)
P11	100.0%	100.0%(=)	100.0%(=)	100.0%(=)	0.0%(+)	0.0%(+)
P12	86.7%	60.0%(+)	20.0%(+)	100.0%(−)	0.0%(+)	76.7%(+)
Number of pluses		10	11	6	10	10
Number of minuses		0	0	2	0	1
r_j(significance level: 5%)		1	1	1	1	1

multimodal numerical experiments, the simple MOA can be competitive, at least for some of the considered problems.

5.3.2. *The probability of convergence on path planning problems*

The results on feasible rate are illustrated in Table 8. The results of the multiple comparison sign test show that MOA significantly outperforms the other compared algorithms except for FA on feasible rate. Consistent with the conclusion in complex challenging numerical experiments, MOA has the ability of finding the global optimum with a higher probability compared with the other compared methods for path planning problems. The better performance of MOA on feasible rate is due to the division of labor between global exploration and local exploitation, and the alternant global and local search strategy.

6. Conclusion

We present a detailed description and convergence proofs of a new "basic" heuristic method in this paper. The proposed method is characterized by two features: global atoms explore the whole solution space in each loop; multivariant search groups carry out the global exploration and the local exploitation respectively. These features enable MOA to discourage the premature convergence and to settle the contradiction between the global exploration and the local exploitation. The global convergence property of this characteristic method is proved through the convergence analysis and the experiments. The results against some peer optimization algorithms demonstrate the superior performance of MOA in the global convergence accuracy and probability for complex challenging problems. In summary, MOA is a global convergence method for global optimization problems.

As a "basic" framework, MOA is light and compact. The parameters in MOA are suggested to be set as follows: *TL* equals 1–2 times the dimension of problems, *r* equals 0.05–0.1 times the absolute difference between the high value and the low value of the search ranges. MOA is characterized by that the global and local atoms conduct the alternate global exploration and local exploitation, respectively. As a result, MOA is suitable for solving complex challenging problems, but it doesn't do well in solving simple problems such as the unimodal problems or the simple multimodal problems with unimodal envelop. To improve the convergence accuracy of MOA, future studies will consider the integration into the presented "basic" framework of more advanced adaptive local search techniques or using other explicitly, implicitly or automatically local search intensity.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61261007 and 11303094) and the Key Program of Yunnan Natural Science Foundation the Grant (2013FA008). We would like to thank professor Suganthan and Yang for sharing the codes of CLPSO, HS-DMS-PSO and FA on http://www.ntu.edu.sg/home/epnsugan and http://www.mathworks.com/matlabcentral/fileexchange/29693-firefly-algorithm. We also want to express our appreciation to professor García and his coauthors for sharing the Java implementation of statistical inference in computational intelligence and data mining on http://sci2s.ugr.es/sicidm/.

Appendix A. Benchmark test functions

F1: Schwefel [27]

$$f_1(x,y) = 418.9828872 \times 2 - x\sin(\sqrt{|x|}) - y\sin(\sqrt{|y|})$$

F2: Levy5

$$f_2(x,y) = \sum_{i=1}^{5} i\cos((i-1)x+i) \times \sum_{i=1}^{5} i\cos((i+1)y+i) + (x+1.42513)^2 + (y+0.80032)^2$$

F3: Langermann [28]

$$f_6(x,y) = -\sum_{i=1}^{5} \frac{c_i\cos(\pi((x-a_i)^2+(y-b_i)^2))}{e^{\frac{(x-a_i)^2+(y-b_i)^2}{\pi}}}$$

$a = [3, 5, 2, 1, 7]$, $b = [5, 2, 1, 4, 9]$, $c = [1, 2, 5, 2, 3]$

F4: Xin-She Yang [29]

$$f_3(\vec{x}) = e^{-\sum_{i=1}^{D}(x_i/\beta)^{2m}} - 2e^{-\sum_{i=1}^{D} x_i^2} \prod_{i=1}^{D} \cos^2(x_i)$$

$\beta = 15, m = 3, D = 2$

F5: Egg Holder [30]

$$f_4(x,y) = -x\sin(\sqrt{|x-y-47|}) - (y+47) \times \sin(\sqrt{|0.5x+y+42|})$$

F6: Damavandi [24]

$$f_5(x,y) = \left(1 - |\frac{\sin(\pi(x-2))\sin(\pi(y-2))}{\pi^2(x-2)(y-2)}|^5\right)(2+(x-7)^2+2(y-7)^2)$$

Functions 7–12 are composition problems selected from [25] where the details of how to construct them are presented. The codes of these functions with the orthogonal matrix **M**, the shifted global optimum and controlling parameters are available at http://www.ntu.edu.sg/home/EPNSugan/.

F7: Composition function 1 (CF1)
$f_1, \ldots, f_{10}$: Sphere function

$$[\delta_1, \ldots, \delta_{10}] = [1, \ldots, 1]$$

$$[\lambda_1, \ldots, \lambda_{10}] = [0.05, \ldots, 0.05]$$

F8: Composition function 2 (CF2)
$f_1, \ldots, f_{10}$: Griewank's function

$$[\delta_1, \ldots, \delta_{10}] = [1, \ldots, 1]$$

$$[\lambda_1, \ldots, \lambda_{10}] = [0.05, \ldots, 0.05]$$

F8: Composition function 3 (CF3)
$f_1, \ldots, f_{10}$: Griewank's function

$$[\delta_1, \ldots, \delta_{10}] = [1, \ldots, 1]$$

$$[\lambda_1, \ldots, \lambda_{10}] = [1, \ldots, 1]$$

F10: Composition function 4 (CF4)
f_{1-2}: Acklley's function
f_{3-4}: Rasrigin's function
f_{5-6}: Weierstrass function
f_{7-8}: Griewank's
f_{9-10}: Sphere function

$$[\delta_1, \ldots, \delta_{10}] = [1, \ldots, 1]$$

$$[\lambda_1, \ldots, \lambda_{10}] = [5/32, 5/32, 1, 1, 10, 10, 0.05, 0.05, 0.05, 0.05]$$

F11: Composition function 5 (CF5)
f_{1-2}: Rasrigin's function
f_{3-4}: Weierstrass function
f_{5-6}: Griewank's
f_{7-8}: Acklley's function
f_{9-10}: Sphere function

$$[\delta_1, \ldots, \delta_{10}] = [1, \ldots, 1]$$

$$[\lambda_1, \ldots, \lambda_{10}] = [0.2, 0.2, 10, 10, 0.05, 0.05, 5/32, 5/32, 0.05, 0.05]$$

F12: Composition function 6 (CF6)
f_{1-2}: Rasrigin's function
f_{3-4}: Weierstrass function
f_{5-6}: Griewank's
f_{7-8}: Acklley's function
f_{9-10}: Sphere function

$$[\delta_1, \ldots, \delta_{10}] = [1, \ldots, 1]$$

$$[\lambda_1, \ldots, \lambda_{10}] = [0.02, 0.04, 3, 4, 0.025, 0.03, 3.5/32, 0.125, 0.045, 0.05]$$

Functions 13–23 are composition problems, which are fully detailed in [26]. We will not describe them again.

References

[1] M.J. Hirsch, C.N. Meneses, P.M. Pardalos, M.G.C. Resende, Global optimization by continuous grasp, Optim. Lett. 1 (2) (2007) 201–212, http://dx.doi.org/10.1007/s11590-006-0021-6.

[2] X.S. Yang, Firefly algorithm, stochastic test functions and design optimisation, Int. J. Bio-Inspir. Comput. 2 (2) (2010) 78–84.

[3] E. Garcia-Gonzalo, J. Fernandez-Martinez, A brief historical review of particle swarm optimization (PSO), J. Bioinform. Intell. Control 1 (1) (2012) 3–16.

[4] S. Sivanandam, S. Deepa, Introduction to Genetic Algorithms, Springer, Berlin, 2007.

[5] E. Mininno, F. Neri, F. Cupertino, et al., Compact differential evolution, IEEE Trans. Evol. Comput. 15 (1) (2011) 32–54.
[6] P.J. Angeline, Evolutionary optimization versus particle swarm optimization: philosophy and performance differences, in: Evolutionary Programming VII, 1998, Springer, Berlin, 1998, pp. 601–610.
[7] Y. Shi, R.C. Eberhart, Fuzzy adaptive particle swarm optimization, in: Proceedings of the 2001 Congress on Evolutionary Computation, IEEE, Washington, 2001, pp. 101–106.
[8] A.K. Gupta, K.G. Smith, C.E. Shalley, The interplay between exploration and exploitation, Acad. Manag. J. 49 (4) (2006) 693–706.
[9] A. Ahrari, A requirement for the mutation operator in continuous optimization, Optim. Lett. 7 (8) (2013) 1681–1690.
[10] S.Z. Zhao, P.N. Suganthan, Q.K. Pan, et al., Dynamic multi-swarm particle swarm optimizer with harmony search, Expert Syst. Appl. 38 (4) (2011) 3735–3742.
[11] Q.H. Nguyen, Y.S. Ong, M.H. Lim, A probabilistic memetic framework, IEEE Trans. Evol. Comput. 13 (3) (2009) 604–623.
[12] C. Garcia-Martinez, M. Lozano, F. Herrera, D. Molina, A.M. Sanchez, Global and local real-coded genetic algorithms based on parent-centric crossover operators, Eur. J. Oper. Res. 185 (2008) 1088–1113.
[13] Z. Zhan, J. Zhang, Y. Li, H.S. Chung, Adaptive particle swarm optimization, IEEE Trans. Syst. Man Cybern. B: Cybern. 39 (2009) 1362–1381.
[14] C. Li, S. Yang, T.T. Nguyen, A self-learning particle swarm optimizer for global optimization problems, IEEE Trans. Cybern. 42 (3) (2012) 627–646.
[15] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Trans. Evol. Comput. 8 (2004) 204–210.
[16] M.A.M. de Oca, T. Stutzle, M. Birattari, M. Dorigo, Frankenstein's PSO: a composite particle swarm optimization algorithm, IEEE Trans. Evol. Comput. 13 (5) (2009) 1120–1132.
[17] A. Auger, N. Hansen, A restart CMA evolution strategy with increasing population size, in: 2005 Congr. Evol. Comput., IEEE, Edinburgh, Scotland, 2005, pp. 1769–1776.
[18] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[19] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[20] J.J. Liang, A.K. Qin, P.N. Suganthan, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (3) (2006) 281–295.
[21] X.S. Yang, Nature-Inspired Metaheuristic Algorithms, Luniver Press, Frome, 2008.
[22] I. Fister, I. JrFister, X.S. Yang, A comprehensive review of firefly algorithms, Swarm Evol. Comput. 13 (2013) 34–46.
[23] V.L. Huang, P.N. Suganthan, J.J. Liang, Comprehensive learning particle swarm optimizer for solving multiobjective optimization problems, Int. J. Intell. Syst. 21 (2) (2006) 209–226.
[24] N. Damavandi, S. Safavi-Naeini, A hybrid evolutionary programming method for circuit optimization, IEEE Trans. Circuits Syst. I: Regul. Pap. 52 (5) (2005) 902–910.
[25] J.J. Liang, P.N. Suganthan, K. Deb, Novel composition test functions for numerical global optimization, in: IEEE Swarm Intelligence Symposium, IEEE, Washington, 2005, pp. 68–75.
[26] P.N. Suganthan, N. Hansen, J.J. Liang, Y.-P.C.K. Deb, A. Auger, S. Tiwari, Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, in: Nanyang Technological University, Singapore, Tech. Rep., 2005.
[27] H.P. Schwefel, Numerical Optimization for Computer Models, John Wiley Sons, 1981.
[28] H. Bersini, M. Dorigo, S. Langerman, Results of the first international contest on evolutionary optimization, in: IEEE International Conf. on Evolutionary Computation, Nagoya, Japan, 1996, pp. 611–615.
[29] M. Jamil, X.S. Yang, A literature survey of benchmark functions for global optimisation problems, Int. J. Math. Model. Numer. Optim. 4 (2) (2013) 150–194.
[30] E.P. Adorio, U.P. Diliman, MVF-Multivariate Test Functions Library in C for Unconstrained Global Optimization, 2005.
[31] A.B. Djurišić, Elite genetic algorithms with adaptive mutations for solving continuous optimization problems-application to modeling of the optical constants of solids, Optics Commun. 151 (1) (1998) 147–159.
[32] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proc. IEEE Congr. Evol. Comput., IEEE, Washington, 1998, pp. 69–73.
[33] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of non-parametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.
[34] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Inf. Sci. 180 (10) (2010) 2044–2064.
[35] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (1) (2006) 1–30.
[36] S. Dasgupta, C.H. Papadimitriou, U.V. Vazirani, Algorithms (2006) 13–15.
[37] A. Gavana, Global Optimization Benchmarks and AMPGO, Available from: http://infinity77.net/global_optimization/index.html.
[38] J.J. Liang, H. Song, B.Y. Qu, Z.F. Liu, Comparison of three different curves used in path planning problems based on particle swarm optimizer, Math. Prob. Eng. 2014 (2014) (2014) 1–15, http://dx.doi.org/10.1155/2014/623156.
[39] N.R. Sturtevant, Benchmarks for grid-based pathfinding, IEEE Trans. Comput. Intell. AI Games 4 (2012) (2012) 144–148.