Neural Comput & Applic
DOI 10.1007/s00521-016-2379-4

ORIGINAL ARTICLE

Ideology algorithm: a socio-inspired optimization methodology

Teo Ting Huan[1] · Anand J. Kulkarni[2,3] · Jeevan Kanesan[1] · Chuah Joon Huang[1] · Ajith Abraham[4]

Received: 7 January 2016 / Accepted: 21 May 2016

Abstract This paper introduces a new socio-inspired metaheuristic technique referred to as ideology algorithm (IA). It is inspired by the self-interested and competitive behaviour of political party individuals which makes them improve their ranking. IA demonstrated superior performance as compared to other well-known techniques in solving unconstrained test problems. Wilcoxon signed-rank test is applied to verify the performance of IA in solving optimization problems. The results are compared with seven well-known and some recently proposed optimization algorithms (PSO, CLPSO, CMAES, ABC, JDE, SADE and BSA). A total of 75 unconstrained benchmark problems are used to test the performance of IA up to 30 dimensions. The results from this study highlighted that the IA outperforms the other algorithms in terms of number function evaluations and computational time. The eminent observed features of the algorithm are also discussed.

Keywords Metaheuristic · Ideology algorithm · Socio-inspired optimization · Unconstrained test problems

✉ Anand J. Kulkarni
kulk0003@uwindsor.ca; anand.kulkarni@sitpune.edu.in

Teo Ting Huan
T900824@gmail.com

Jeevan Kanesan
jievan@um.edu.my

Chuah Joon Huang
jhchuah@um.edu.my

Ajith Abraham
ajith.abraham@ieee.org

1 Department of Electrical Engineering, Faculty of Engineering, University Malaya, Kuala Lumpur, Malaysia

2 Odette School of Business, University of Windsor, 401 Sunset Avenue, Windsor, ON N9B3P4, Canada

3 Department of Mechanical Engineering, Symbiosis Institute of Technology, Symbiosis International University, Pune, MH 412 115, India

4 Machine Intelligence Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, Auburn, WA 98071, USA

1 Introduction and motivation

Over the last two decades, metaheuristic optimization techniques have become increasingly popular and essential in applied mathematics [1–3]. Optimization algorithms are functioning as to find the best values for system variables under various conditions. Some well-known metaheuristics such as particle swarm optimization (PSO) [4], genetic algorithm (GA) [5], ant colony optimization (ACO) [6] are fairly well known, and they are applied in various fields. With regard to some drawbacks of classical optimization strategies as well as to achieve simplicity, flexibility and derivation-free mechanism, several metaheuristics have been designed [7–49].

Metaheuristics are inspired by simple concepts. They are usually related to physical phenomenon, animal's behaviour and evolutionary concepts. The simplicity allows researchers to simulate different natural concepts and propose new metaheuristics, their hybridization and improved versions. Second, the applicability of metaheuristics to a variety of problems without significant changes in the structure/framework of algorithms makes them flexible. In other words, little problem-specific information is required. Also, different techniques could be deployed to support the algorithm solving a variety of problem classes. Third, the majority of metaheuristics have

Published online: 08 June 2016

derivative-free mechanisms. Metaheuristics find the solutions stochastically in contrast with gradient-based optimization techniques. The process of optimization starts with random solution(s), and the calculation of derivative for deciding the search direction is not required. This makes metaheuristics appropriate to apply for real-world problems. Finally, the ability of metaheuristics to avoid local optima makes them reach quickly in the close neighbourhood of the region where the global optimum could be potentially located.

Generally, metaheuristics can be classified into three main classes: evolutionary, physics based and swarm based. Evolutionary algorithms (EA) are inspired by the concepts of evolution in nature. When the objective function for an optimization problem is nonlinear and nondifferentiable, EA techniques are typically used to find the global optimum [13–15]. EAs have been applied for various real-world engineering problems such as reverse engineer causal networks [16], commercial computer-automated exterior lighting design [17], nanoscale crossbar architectures [18], dynamic stochastic districting and routing problem [19], neural network classifier [20], assembly line configurations [21], configurations of mobile applications [22], electric power distribution networks [23], word sense disambiguation problem [24], surgery scheduling problems [25], image processing [26] and speech recognition [27].

The recent popular optimization techniques are from swarm intelligence (SI) domain. SI is characterized by its unique mechanism which mimics the behaviour of swarms of social insects, flocks of birds and schools of fish [15, 28]. The benefits of these approaches as compared with conventional techniques are the flexibility and robustness. These properties make SI a successful design paradigm for algorithms to deal with increasingly complex problems [29]. PSO simulates the social behaviour as a representation of the movement of organisms in the school of fish [30]. The comprehensive learning PSO (CLPSO) [32] and PSO2011 [33] are the recent versions of the standard PSO [4]. The ACO algorithm is proposed based on strategies of ants in accessing food sources [6]. In artificial bee colony (ABC) algorithm, the natural behaviour of honey bees in discovering food sources is imitated [15]. The cuckoo search (CS) algorithm is based on the behaviour of cuckoo species by laying their eggs in other nests of host birds [31]. A recently proposed algorithm, named covariance matrix adaptation evolution strategy (CMAES) [36], is based on basic genetic rules. The differential evolution (DE) algorithm [34, 35] is a population-based stochastic function minimizer. The adaptive differential evolution algorithm (JDE) [37], the parameter adaptive differential evolution algorithm (JADE) [13] and the self-adaptive differential evolution algorithm (SADE) [35] are recent versions of DE. Another recently proposed algorithm referred to as backtracking search algorithm (BSA) generates a trial individual using basic genetic operators (selection, mutation and crossover). A nonuniform crossover strategy which is more complex than the crossover strategies used in many genetic algorithms is used in the BSA [38].

The work presented in this paper is motivated from ideologies which exist in human society for ages. In the context of politics, there are numerous kinds of ideologies such as conservative, socialism, left-wing, right-wing, democratic, republic, and communism. There are several political parties exist in the world which follow these ideologies in different forms, for example Conservative Party and Labour Party (UK), Republican Party and Democratic Party (USA), Communist Party (China) and Bharatiya Janata Party (India).

This paper introduces a novel socio-inspired algorithm referred to as ideology algorithm (IA). The society individuals support or follow certain ideologies. These ideologies become the guide or way for the individuals to achieve their long-term goals. The IA is motivated from the competition within the members of a political party as well as competition amongst the leaders of different parties. Every local party follows certain ideology which motivates certain individuals stay associated with that party. Once associated with a party, every individual exhibits a self-interested behaviour and competes with its party members to improve and promote its rank. Every individual looks at its own local party leader as a benchmark and tries to reach as close as possible to it. Also, the individual watches other party leaders and compares itself with that leader. This may motivate it to choose different ideology associated with another party. Furthermore, every local party leader always desires to be a global leader. In other words, it competes with the other party leaders to be a global leader. Moreover, the local party leader desires to remain at least its own party leader. Thus, it also competes with the second best in the party which always desires to catch the local party leader position. In addition, the lowest rank individual following the party ideology desires to climb up in the party, however, for prolonged time if it understands that following its current party ideology is not improving its rank. Such deserted individual may change the ideology and resort to another party ideology. This competitive behaviour of individuals following certain ideology to improve and climb up in the party as well as compete with other party members is modelled. The mechanism enabled the IA to solve several numerical optimization problems with superior performance in terms of solution quality and computational cost as compared with other existing algorithms.

The remainder parts in this work are organized as follows: Sect. 2 describes the mechanism of IA. Section 3 provides the detailed results of the computational experiments conducted to validate the algorithm. In Sect. 4, conclusions and future directions are provided.

2 Ideology algorithm (IA)

In the context of the IA, every member or individual associated with a party is a possible solution. Its position in the party depends on the quality or fitness of its solution (objective function value). The individual with the best solution in a party is considered as local party leader, and the individual best amongst all the party leaders is considered as the global leader. The local party leader competes with every other party leader with a desire to be a global leader. It also competes with the second best individual in its own party as it is challenged by the second best in the party which desires to be the local party leader. The earlier makes the party leaders explore and locate promising search space. The later forces the party leader to look for a better solution in its own local neighbourhood as well as the second best in the party. This may increase its chances of remaining as the local party leader and improve. The individual in the party with worst solution checks the difference between its own solution and the penultimate worst in the same party. If the difference is greater than a pre-specified value, then such deserted individual understands that following the current party ideology is not worth for it. This makes him switchover to another party in a hope to be better off and climb up in that party. The framework makes the individual in every party to directly and indirectly compete with the same party individuals as well as other party individuals. This essentially makes every party to remain in competition and grow which motivates the individuals search for better solutions. The IA procedure is explained below in detail.

Consider a general unconstrained problem (in minimization sense) as follows:

$$\text{Minimize} \quad f(\boldsymbol{x}) = f(x_1, \ldots, x_i, \ldots, x_N)$$
$$\text{Subject to} \quad \Psi_i^l \le x_i \le \Psi_i^u, i = 1, \ldots, N \tag{1}$$

The IA procedure starts with equally dividing the entire population into P parties. In the beginning, the number of individuals M_p in every party $p(p = 1, \ldots, P)$ is equal, i.e. $M_1 =, \ldots, M_p, \ldots, = M_P$. Also the desertion parameter T associated with the worst individual, convergence parameter ε, maximum number of iterations $I_{\max}$ and reduction factor $R \in [0, 1]$ are chosen.

Step 1 *(Party Formation)*

Equally divide the sampling space associated with every variable $x_i, i = 1, \ldots, N$ into P party subspaces, i.e. $\forall x_i \boldsymbol{\Psi}_i^p = \left[\Psi_i^{l,p}, \Psi_i^{u,p}\right], p = 1, \ldots, P, i = 1, \ldots, N.$

Step 2 *(Evaluation)*

From within every party $p = 1, \ldots, P$ subspace $\boldsymbol{\Psi}_i^p = \left[\Psi_i^{l,p}, \Psi_i^{u,p}\right]$ associated with every variable $x_i, i = 1, \ldots, N$, M_p values are randomly sampled and associated objective functions are evaluated. Then, the individuals associated with every party $p, p = 1, \ldots, P$ could be represented as follows:

$$\forall p f\left(x_{1,m_p}, \ldots, x_{i,m_p}, \ldots, x_{N,m_p}\right), m_p = 1, \ldots, M_p$$
or
$$p = \begin{bmatrix} f\left(x_{1,1}, \ldots, x_{i,1}, \ldots, x_{N,1}\right) \\ \vdots \\ f\left(x_{1,m_p}, \ldots, x_{i,m_p}, \ldots, x_{N,m_p}\right) \\ \vdots \\ f\left(x_{1,M_p}, \ldots, x_{i,M_p}, \ldots, x_{N,M_p}\right) \end{bmatrix}, p = 1, \ldots, P \tag{2}$$

Step 3 *(Local Party Ranking)*

From within every party $p, p = 1, \ldots, P$ the evaluated individuals are ranked from the best individual referred to as local party leader $L_{p,b}$ to the local worst individual $L_{p,w}$ as shown in Fig. 1.

Step 4 *(Competition and Improvement for local party leader $L_{p,b}$)*

Every local party leader $L_{p,b}, (p = 1, \ldots, P)$ seeks to improve itself through introspection, local competition and global competition. The *introspection* refers to searching the close neighbourhood of its own current solution by modifying the current sampling space associated with its every variable $x_i^{L_{p,b}}, i = 1, \ldots, N$ as follows:

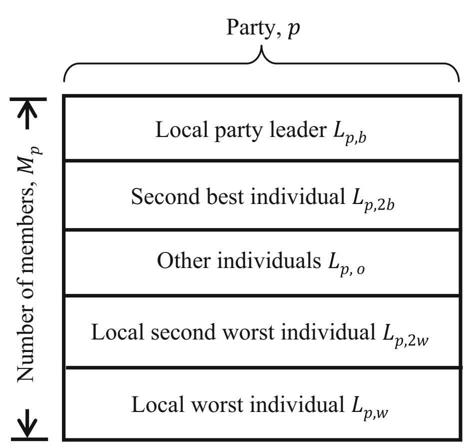

Fig. 1 The arrangement of individuals in a party

$$\Psi_{i,\text{insp}}^{L_{p,b}} \in \left[x_i^{L_{p,b}} - R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right), x_i^{L_{p,b}} + R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right)\right] \tag{3}$$

The *local competition* refers to competing with the second best $L_{p,2b}$ in its own party by searching in the close neighbourhood of its current solution. In other words, the current sampling space of every variable $i, i = 1, \ldots, N$ associated with every local party leader $L_{p,b}, (p = 1, \ldots, P)$ is updated to the close neighbourhood of the local best solution $L_{p,2b}$ as follows:

$$\Psi_{i,\text{lcmp}}^{L_{p,b}} \in \left[x_i^{L_{p,2b}} - R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right), x_i^{L_{p,2b}} + R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right)\right] \tag{4}$$

Local competition is necessary as the local party leader will always try to remain the best in its own party which may further lead the algorithm to efficiently search for better solution.

The *global competition* refers to searching in the close neighbourhood of the global leader. In other words, the current sampling space of every variable $i, i = 1, \ldots, N$ associated with every local party leader $L_{p,b}(p = 1, \ldots, P)$ is updated to the close neighbourhood of the global best solution $L_{p,\text{gb}}$ as follows:

$$\Psi_{i,\text{gcomp}}^{L_{p,b}} \in \left[x_i^{L_{p,\text{gb}}} - R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right), x_i^{L_{p,\text{gb}}} + R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right)\right] \tag{5}$$

where $L_{p,\text{gb}} = \min\left(L_{p,b}\right), p = 1, \ldots, P$ or $L_{p,\text{gb}} = \min\left(L_{1,b}, \ldots, L_{p,b}, \ldots, L_{P,b}\right)$.

Then, the local party leader $L_{p,b}$ samples variable values from within the updated sampling intervals $\Psi_{i,\text{insp}}^{L_{p,b}}$, $\Psi_{i,\text{lcmp}}^{L_{p,b}}$ and $\Psi_{i,\text{gcomp}}^{L_{p,b}}$ formed using introspection, local competition and global competition, respectively, and calculates corresponding objective functions. Then, one solution from within the three choices is selected based on the roulette wheel selection approach [42]. It is important to mention that the introspection will not make the leader ignore its recent local neighbourhood as its current solution could be far better than other individuals.

For each local worst individual $L_{p,w}(p = 1, 2, \ldots, P)$, the distance d between itself and the second worst individual $L_{p,2w}$ is evaluated as follows:

$$d = L_{p,w} - L_{p,2w} \tag{6}$$

If the difference is higher than a pre-specified value T, then the individual understands that it is deserted (worst off) and switches over to another randomly selected party $\tilde{p} \in [1, \ldots, P]$, i.e.

$$\begin{aligned} L_{\tilde{p},w+1}^{\tilde{p}} &= L_{p,w}^{p} \quad if d \geq T \\ L_{p,w}^{p} &= L_{p,o}^{p} \quad otherwise \end{aligned} \tag{7}$$

where $L_{p,o}(o = 1, 2, \ldots, O)$ is referred to as ordinary individual for party $(p = 1, 2, \ldots, P)$.

Step 5*(Updating party individuals)*

Every ordinary individual $L_{p,o}(o = 1, 2, \ldots, O)$ in every party $(p = 1, 2, \ldots, P)$ searches for a single solution in its own neighbourhood $\Psi_{i,o}^{L_{p,o}}$ as well as every local party leader's neighbourhood $\Psi_{i,\text{lopl}}^{L_{p,o}}$, $(p = 1, 2, \ldots, P)$. The best O solutions are chosen from within this pool, and the party individuals are updated. In other words, the current sampling space of every variable $(i, i = 1, \ldots, N)$ associated with every individual $L_{p,o}(o = 1, 2, \ldots, O), (p = 1, \ldots, P)$ other than the local party leader $L_{p,b}$ and deserted individual $L_{p,w}$ updates its sampling space in the close neighbourhood of itself [refer to Eqs. (8) and (9)], and every local party leader $L_{p,b}(p = 1, \ldots, P)$ is as follows:

$$\Psi_{i,o}^{L_{p,o}} \in \left[x_i^{L_{p,o}} - R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right), x_i^{L_{p,o}} + R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right)\right] \tag{8}$$

$$\Psi_{i,\text{lopl}}^{L_{p,o}} \in \left[x_i^{L_{p,b}} - R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right), x_i^{L_{p,b}} + R \times \left(\|\Psi_i^{u,p}\| - \left\|\Psi_i^{l,p}\right\|\right)\right] \tag{9}$$

In this way, each individual of every party is updated.

Step 6 *(Convergence)*

The parties are considered converged if any of the following conditions satisfied, else continue to Step 1:

(a) There is no significant improvement in the local party leader solutions for a significant number of iterations and/or
(b) The maximum number of iterations $I_{\max}$ is reached.

It is important to mention that the number of party members may change in every iteration as some of the individuals may leave a party and join any other in hope to improve. Figure 2 shows the general structure of IA, where Fig. 3 shows the flow chart of IA. The working mechanism of the IA is illustrated in Fig. 4, where the dots represent individuals or party members and the centralized dot represents leader in a party. Figure 5 illustrates the movement of individuals during a run solving multimodal Ackley function. It exhibits the ability of the algorithm to quickly jump out of the local minima and reach the global minimum solution.

1: Initialization
2: Party formation
3: Generation of party individuals
4: **repeat**
5: Evaluation
6: Local ranking
7: Competition and improvement
8: Updating party individuals
9: **until** convergence

Fig. 2 General structure of IA

3 Results and discussion

In this section, the tests and benchmark problems, statistical analysis, control parameters and convergence conditions used for the IA in the tests are presented. The performance of IA is investigated in detail. For experiments, each algorithm (PSO, CMAES, ABC, JDE, CLPSO, SADE, BSA and IA) is coded in MATLAB R2013a on Windows Platform with a T6400@4 GHz Intel Core 2 Duo processor with 4 GB RAM.

3.1 Benchmark problems

Two tests are conducted to examine the performance of IA and the comparison algorithms in solving the numerical optimization problems. Test 1 involved 50 widely used benchmark problems [15, 41]. Table 1 summarizes several features of the benchmark problems used in Test 1. Test 2 involved 25 benchmark problems used in CEC2005 [45]. Table 2 summarizes several features of the benchmark problems used in Test 2.

3.2 Control parameters

The values of the control parameters used in the experiments for IA are listed as shown in Table 3.

3.3 Stopping criterion

The predetermined stopping criterion is set to terminate the algorithms.

- Stop when the absolute value of the objective function evaluations is less than 10^{-16}.
- Stop when the maximum number of function evaluations reaches 200000.
- Stop when the maximum number of iterations ($I_{\max}$) is reached.

Parametric tests have been commonly used in the analysis of experiments. For example, a common way to test whether the difference between the results of two algorithms is nonrandom is to apply a paired t test, which checks whether the average difference in their performance over the problems is significantly different from zero. Nonparametric tests, besides their original definition for dealing with nominal or ordinal data, can be also applied to continuous data by conducting ranking-based transformations, adjusting the input data to the test requirements. They can perform two classes of analysis: pairwise comparisons and multiple comparisons. Pairwise statistical procedures perform individual comparisons between two algorithms, obtaining in each application a p value independent from another one [40]. Pairwise comparisons are the simplest kind of statistical tests that a researcher can apply within the framework of an experimental study. Such tests are directed to compare the performance of two algorithms when applied to a common set of problems. In multi-problem analysis, a value for each pair of algorithm is required (often an average value from several runs). In this section, we focus on the sign test, which is a quick and easy procedure that can provide a clearer view about the comparison. Then, the Wilcoxon signed-rank test is introduced as an example of a simple nonparametric test for pairwise statistical comparisons.

3.4 Statistical analysis

The Wilcoxon signed-rank test is a nonparametric procedure employed in hypothesis testing situations, involving a design with two samples. It is commonly used for answering the following question: Do two samples represent two different populations? This is analogous to the paired t test in nonparametric statistical procedures. Thus, it is a pairwise test that aims to detect significant differences between two sample means, i.e. the behaviour of two algorithms.

Table 4 shows the mean runtimes and simple statistical values for the results obtained in Test 1, whereas Table 6 lists the algorithms that obtained statistically better solutions compared with the other algorithms in Test 1, based on the Wilcoxon signed-rank test. Table 5 shows the mean runtimes and simple statistical values for the results obtained in Test 2, whereas Table 7 lists the algorithms that provided statistically better solutions compared with the other algorithms in Test 2, based on the Wilcoxon signed-rank test.

Table 8 presents the multi-problem-based pairwise statistical comparison results using the averages of the global minimum values obtained through 30 runs of IA and the comparison algorithms to solve the benchmark problems in

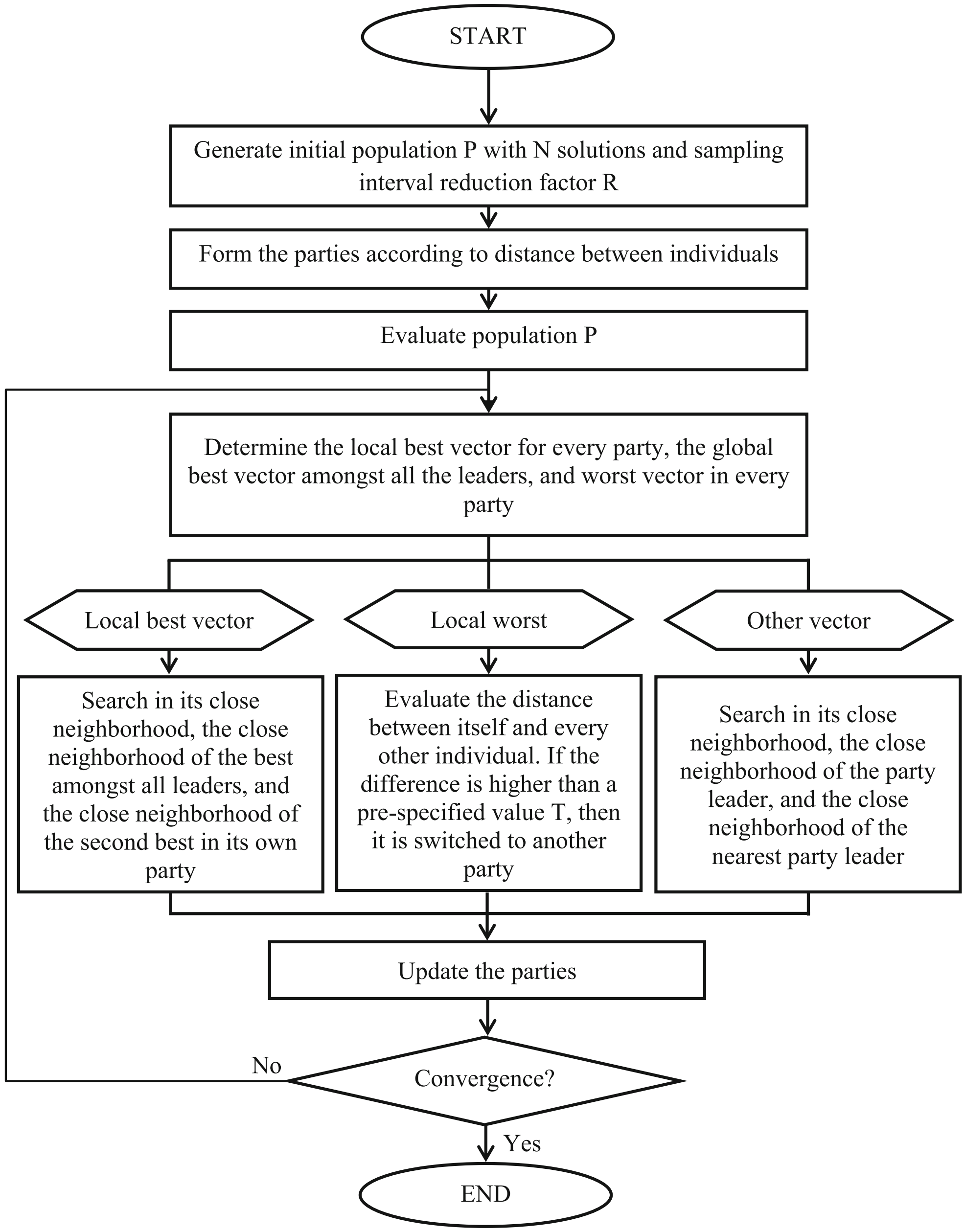

Fig. 3 Flow chart of ideology algorithm (IA)

Test 1 and Test 2. The results indicate that IA was statistically more successful than most of the comparison algorithms with a statistical significance value $\alpha = 0.05$.

In Tables 6 and 7, a '+' sign indicates cases in which the null hypothesis is rejected and IA displays a statistically superior performance in the problem-based statistical comparison tests at the 95 % significance level ($\alpha = 0.05$). The '$-$' sign indicates cases in which the null hypothesis was rejected and IA displayed an inferior performance; '=' indicates cases in which there was no statistical difference between the two algorithms' success in solving the problems. The last rows of Tables 6 and 7 depict the total counts in a format of '$+/=/-$' for the three statistical significance cases (marked with '+', '=' or '$-$') in the pairwise comparison.

When the $(+/=/-)$ values are examined, it can be said that IA is statistically more successful than most of the other algorithms in solving the problems in Tests 1 and 2. Although, the successes IA and BSA have had are statistically identical; IA has provided statistically better solutions than other algorithms.

For pairwise comparison of the problem-solving success of EAs, a problem-based or multi-problem-based statistical comparison method can be used [40]. A problem-based

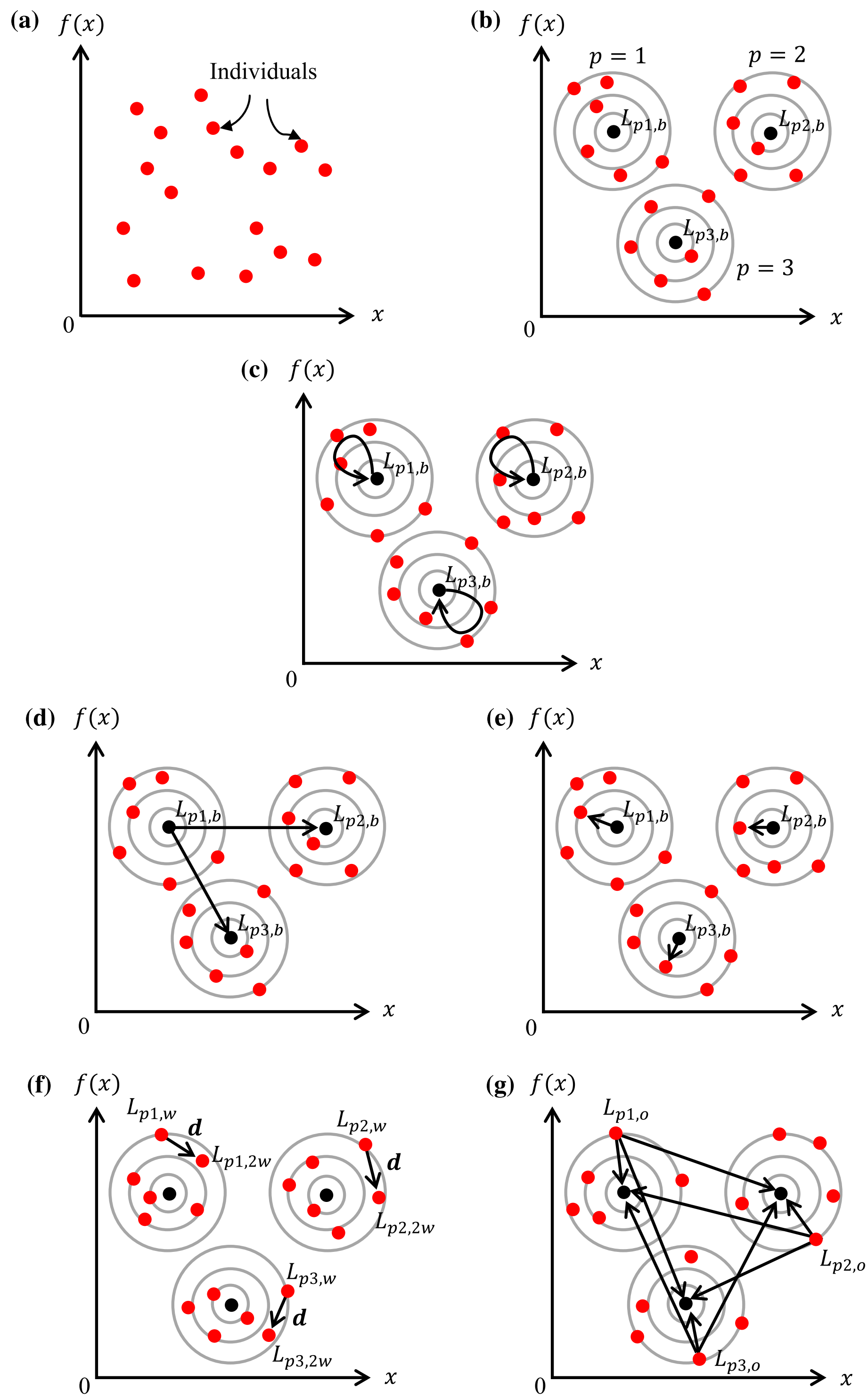

Fig. 4 **a** Initialization of the individuals. **b** Associate individuals with p parties ($p = 1, 2, \ldots, P$). **c** Introspection. **d** Global competition. **e** Local competition. **f** Search by local worst vector. **f** Search by ordinary individual $L_{p,o}$

comparison can use the global minimum values obtained for the problem as the result of several runs. Problem-based pairwise comparisons are widely used to determine which of two algorithms solves a specific numerical optimization problem with greater statistical success.

The global minimum values obtained are used in this paper as the result of 30 runs for its problem-based pairwise comparison of the algorithms. A multi-problem-based pairwise comparison can use the average of the global minimum values obtained as the result of several runs. Multi-problem-based pairwise comparisons determine which algorithm is statistically more successful in a test that includes several benchmark problems [40]. The average of global minimum values obtained is used in this

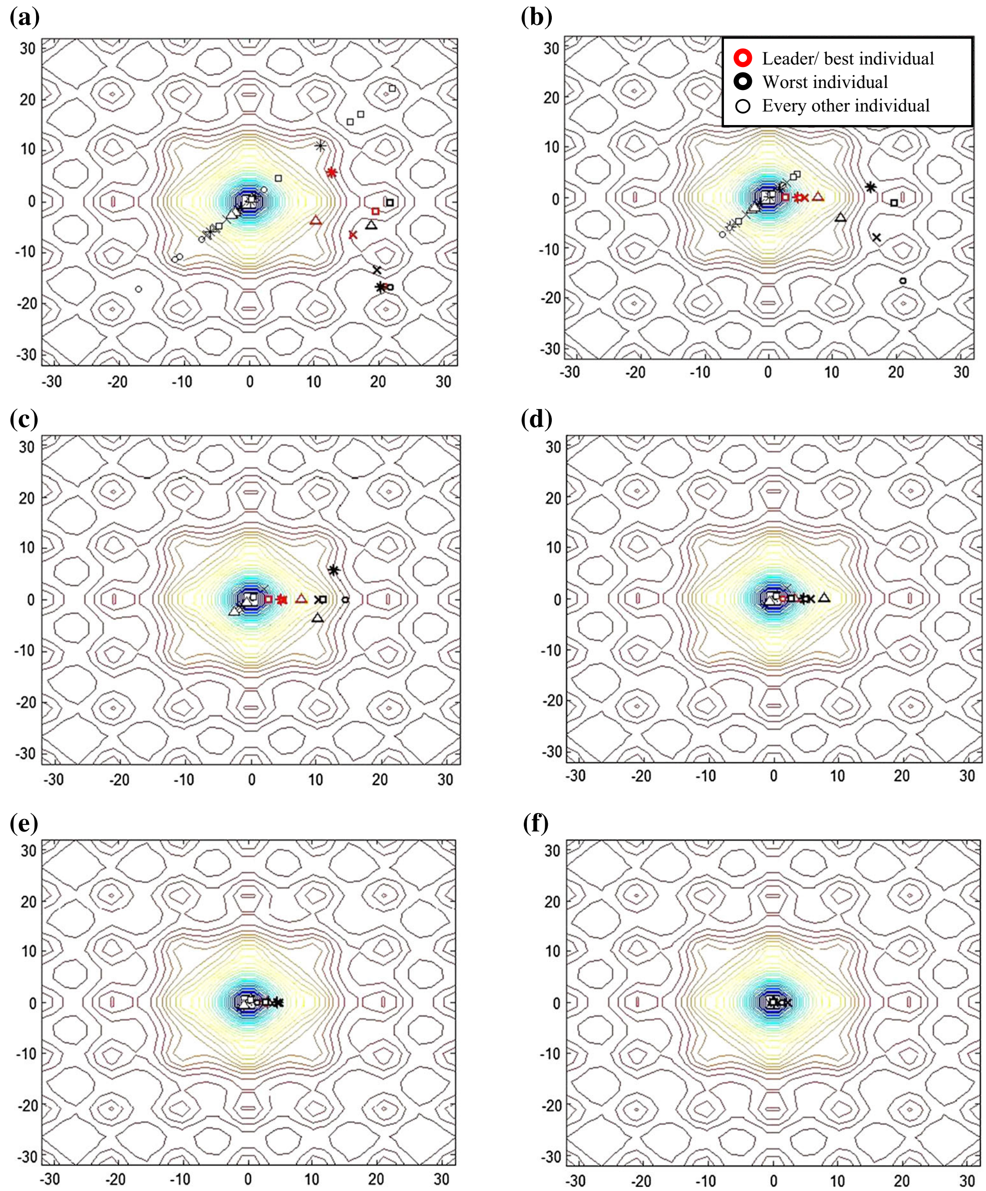

Fig. 5 Search pattern for benchmark function F5 (Ackley), different symbols denote different parties. **a** Initialization of population. **b** Iteration 1. **c** Iteration 3. **d** Iteration 5. **e** Iteration 10. **f** Convergence

paper as the result of 30 runs for its multi-problem-based comparison of the algorithms. The Wilcoxon signed-rank test was used for pairwise comparisons, with the statistical significance value $\alpha = 0.05$.

The null hypothesis H0 for this test is: 'There is no difference between the median of the solutions achieved by algorithm A and the median of the solutions obtained by algorithm B for the same benchmark problem'. In other words, we assume that median (A) = median (B).

To determine whether algorithm A reached a statistically better solution than algorithm B, or whether the alternative hypothesis was valid, the sizes of the ranks provided by the Wilcoxon signed-rank test (T+ and T− as defined in [40]) are examined thoroughly.

3.5 PSO versus IA

In PSO, the individual particles of a swarm symbolize potential solutions. They 'fly' through the search space of the problem, trying to seek an optimal solution. The current positions of the particles are broadcasted to other neighbouring particles. Previously identified 'good position' is then used as a starting point by the swarm for further search. On the other hand, the individual particles adjust their current positions and velocities. A distinct characteristic of PSO is its fast convergent behaviour and inherent adaptability, especially when compared to conventional EAs [47]. Theoretical analysis of PSO [4, 30] proves that particles in a swarm can switch between an exploratory

Table 1 The benchmark problems used in Test 1 (*Dim* dimension, *low and up* limitations of search space, *U* unimodal, *M* multimodal, *S* separable, *N* nonseparable)

Problem	Name	Type	Low	Up	Dimension
F1	Foxholes	MS	−65.536	65.536	2
F2	Goldstein-Price	MN	−2	2	2
F3	Penalized	MN	−50	50	30
F4	Penalized2	MN	−50	50	30
F5	Ackley	MN	−32	32	30
F6	Beale	UN	−4.5	4.5	5
F7	Bohachevsky1	MS	−100	100	2
F8	Bohachevsky2	MN	−100	100	2
F9	Bohachevsky3	MN	−100	100	2
F10	Booth	MS	−10	10	2
F11	Branin	MS	−5	10	2
F12	Colville	UN	−10	10	4
F13	Dixon-Price	UN	−10	10	30
F14	Easom	UN	−100	100	2
F15	Fletcher	MN	−3.1416	3.1416	2
F16	Fletcher	MN	−3.1416	3.1416	5
F17	Fletcher	MN	−3.1416	3.1416	10
F18	Griewank	MN	−600	600	30
F19	Hartman3	MN	0	1	3
F20	Hartman6	MN	0	1	6
F21	Kowalik	MN	−5	5	4
F22	Langermann2	MN	0	10	2
F23	Langermann5	MN	0	10	5
F24	Langermann10	MN	0	10	10
F25	Matyas	UN	−10	10	2
F26	Michalewics2	MS	0	3.1416	2
F27	Michalewics5	MS	0	3.1416	5
F28	Michalewics10	MS	0	3.1416	10
F29	Perm	MN	−4	4	4
F30	Powell	UN	−4	5	24
F31	Powersum	MN	0	4	4
F32	Quartic	US	−1.28	1.28	30
F33	Rastrigin	MS	−5.12	5.12	30
F34	Rosenbrock	UN	−30	30	30
F35	Schaffer	MN	−100	100	2
F36	Schwefel	MS	−500	500	30
F37	Schwefel_1_2	UN	−100	100	30
F38	Schwefel_2_22	UN	−10	10	30
F39	Shekel10	MN	0	10	4
F40	Shekel5	MN	0	10	4
F41	Shekel7	MN	0	10	4
F42	Shubert	MN	−10	10	2
F43	Six-hump camelback	MN	−5	5	2
F44	Sphere2	US	−100	100	30
F45	Step2	US	−100	100	30
F46	Stepint	US	−5.12	5.12	5
F47	Sumsquares	US	−10	10	30
F48	Trid6	UN	−36	36	6
F49	Trid10	UN	−100	100	10
F50	Zakharov	UN	−5	10	10

Table 2 The benchmark problems used in Test 2 (*Dim* dimension, *low and up* limitations of search space, *U* unimodal, *M* multimodal, *E* expanded, *H* hybrid)

Problem	Name	Type	Low	Up	Dimension
F51	Shifted sphere	U	−100	100	10
F52	Shifted Schwefel	U	−100	100	10
F53	Shifted rotated high conditioned elliptic function	U	−100	100	10
F54	Shifted Schwefels problem 1.2 with noise	U	−100	100	10
F55	Schwefels problem 2.6	U	−100	100	10
F56	Shifted Rosenbrock's	M	−100	100	10
F57	Shifted rotated Griewank's	M	0	600	10
F58	Shifted rotated Ackley's	M	−32	32	10
F59	Shifted Rastrigin's	M	−5	5	10
F60	Shifted rotated Rastrigin's	M	−5	5	10
F61	Shifted rotated Weierstrass	M	−0.5	0.5	10
F62	Schwefels problem 2.13	M	−100	100	10
F63	Expanded extended Griewank's + Rosenbrock's	E	−3	1	10
F64	Expanded rotated extended Scaffes	E	−100	100	10
F65	Hybrid composition function	HC	−5	5	10
F66	Rotated hybrid comp. Fn 1	HC	−5	5	10
F67	Rotated hybrid comp. Fn 1 with noise	HC	−5	5	10
F68	Rotated hybrid comp. Fn 2	HC	−5	5	10
F69	Rotated hybrid comp. Fn 2 with narrow global optimal	HC	−5	5	10
F70	Rotated hybrid comp. Fn 2 with the global optimum	HC	−5	5	10
F71	Rotated hybrid comp. Fn 3	HC	−5	5	10
F72	Rotated hybrid comp. Fn 3 with high condition number matrix	HC	−5	5	10
F73	Noncontinuous rotated hybrid comp. Fn 3	HC	−5	5	10
F74	Rotated hybrid comp. Fn 4	HC	−5	5	10
F75	Rotated hybrid comp. Fn 4	HC	−2	5	10

Table 3 The relevant control parameters used in the experiments for IA

Control parameter	Numerical values
Maximum number of iteration ($I_{\max}$)	30
Number of parties presented (p)	5
Initial population size (x)	150
Reduction factor (R)	0.000001

mode with large search step sizes, as well as an exploitative mode with smaller search step sizes.

Each particle in PSO is determined by its current position as shown in Eq. (10), as well as its current velocity as shown in Eq. (11) [48]. In each iteration, the particle's velocity is modified by its personal best position, which is the position giving the best fitness value. Also, they are determined by the global best position, which is the position of the best-fit particle from the swarm [47]. As a result, each particle searches around a region defined by its personal best position and global best position.

$$\vec{x}(t+1) = \vec{x}(t) + \vec{v}(t) + 1 \tag{10}$$

$$\vec{v}(t+1) = \omega\vec{v}(t) + \emptyset_1 \mathrm{rand}(0,1)(\vec{p}(t) - \vec{x}(t)) + \emptyset_2 \mathrm{rand}(0,1)(\vec{g}(t) - \vec{x}(t)) \tag{11}$$

The parameter ω is known as inertia weight, and it controls the magnitude of the old velocity, $\vec{v}(t)$ to calculate the new velocity, $\vec{v}(t+1)$. The parameters $\emptyset_1$ and $\emptyset_2$ determine the significance of $\vec{p}(t)$ and $\vec{g}(t)$, respectively. The procedure of PSO is as shown in Fig. 6.

The drawback of the basic PSO algorithm is that it easily suffers from the partial optimism, which might lead to reduced precision in speed and the regulation of direction. PSO is unable to solve the problems of scattering and optimization, as well as the problems of noncoordinate system, such as the solution to the energy field and the moving rules of the particles in the energy field [47–49]. In this paper, the proposed IA is being compared with PSO and its variants, CLPSO. The results proved that IA outperforms PSO and CLPSO in terms of runtime in most of the benchmark functions of Test 1 and Test 2. The statistical results of Test 1 as shown in Table 6 indicate that IA is equally good as compared with PSO and CLPSO. However, statistical results of Test 2 as shown in Table 7

Table 4 Statistical solutions obtained by PSO, CMAES, ABC, CLPSO, SADE, BSA and proposed IA in Test 1 (*mean* mean solution, *SD* standard deviation of mean solution, *best* best solution, *runtime* mean runtime in seconds)

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F1	Mean	1.3316029264876300	10.0748846367972000	0.9980038377944500	1.0641405484285200
	SD	0.9455237994690700	8.0277365400340800	0.0000000000000001	0.3622456829347420
	Best	0.9980038377944500	0.9980038377944500	0.9980038377944500	0.9980038377944500
	Runtime	72.527	44.788	64.976	51.101
F2	Mean	2.9999999999999200	21.8999999999995000	3.0000000465423000	2.9999999999999200
	SD	0.0000000000000013	32.6088098948516000	0.0000002350442161	0.0000000000000013
	Best	2.9999999999999200	2.9999999999999200	2.9999999999999200	2.9999999999999200
	Runtime	17.892	24.361	16.624	7.224
F3	Mean	0.1278728062391630	0.0241892995662904	0.0000000000000004	0.0034556340083499
	SD	0.2772792346028400	0.0802240262581864	0.0000000000000001	0.0189272869685522
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000003	0.0000000000000000
	Runtime	139.555	5.851	84.416	9.492
F4	Mean	0.0043949463343535	0.0003662455278628	0.0000000000000004	0.0007324910557256
	SD	0.0054747064090174	0.0020060093719584	0.0000000000000001	0.0027875840585535
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000003	0.0000000000000000
	Runtime	126.507	6.158	113.937	14.367
F5	Mean	1.5214322973725000	11.7040011684582000	0.0000000000000340	0.0811017056422860
	SD	0.6617570384662600	9.7201961540865200	0.0000000000000035	0.3176012689149320
	Best	0.0000000000000080	0.0000000000000080	0.0000000000000293	0.0000000000000044
	Runtime	63.039	3.144	23.293	11.016
F6	Mean	0.0000000041922968	0.2540232169641050	0.0000000000000028	0.0000000000000000
	SD	0.0000000139615552	0.3653844307786430	0.0000000000000030	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000005	0.0000000000000000
	Runtime	32.409	4.455	22.367	1.279
F7	Mean	0.0000000000000000	0.0622354533647150	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.1345061339146580	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	16.956	6.845	1.832	1.141
F8	Mean	0.0000000000000000	0.0072771062590204	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0398583525142753	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	17.039	2.174	1.804	1.139
F9	Mean	0.0000000000000000	0.0001048363065820	0.0000000000000006	0.0000000000000000
	SD	0.0000000000000000	0.0005742120996051	0.0000000000000003	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000001	0.0000000000000000
	Runtime	17.136	2.127	21.713	1.129
F10	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	17.072	1.375	22.395	1.099
F11	Mean	0.3978873577297380	0.6372170283279430	0.3978873577297380	0.3978873577297380
	SD	0.0000000000000000	0.7302632173480510	0.0000000000000000	0.0000000000000000
	Best	0.3978873577297380	0.3978873577297380	0.3978873577297380	0.3978873577297380
	Runtime	17.049	24.643	10.941	6.814
F12	Mean	0.0000000000000000	0.0000000000000000	0.0715675060725970	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0579425013417103	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0013425253994745	0.0000000000000000
	Runtime	44.065	1.548	21.487	1.251

Table 4 continued

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F13	Mean	0.6666666666666750	0.6666666666666670	0.0000000000000038	0.6666666666666670
	SD	0.0000000000000022	0.0000000000000000	0.0000000000000012	0.0000000000000002
	Best	0.6666666666666720	0.6666666666666670	0.0000000000000021	0.6666666666666670
	Runtime	167.094	3.719	37.604	18.689
F14	Mean	−1.0000000000000000	−0.1000000000000000	−1.0000000000000000	−1.0000000000000000
	SD	0.0000000000000000	0.3051285766293650	0.0000000000000000	0.0000000000000000
	Best	−1.0000000000000000	−1.0000000000000000	−1.0000000000000000	−1.0000000000000000
	Runtime	16.633	3.606	13.629	6.918
F15	Mean	0.0000000000000000	1028.3930784026900000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	1298.1521820113500000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	27.859	15.541	40.030	2.852
F16	Mean	48.7465164446927000	1680.3460230073400000	0.0218688498331872	0.9443728655432830
	SD	88.8658510972991000	2447.7484859066000000	0.0418409568792831	2.8815514827061600
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000016	0.0000000000000000
	Runtime	95.352	11.947	44.572	4.719
F17	Mean	918.9518492782850000	12340.2283326398000000	11.0681496253548000	713.7226974626920000
	SD	1652.4810858411400000	22367.1698875802000000	9.8810950146557100	1710.071307430120000
	Best	0.0000000000000000	0.0000000000000000	0.3274654777056860	0.0000000000000000
	Runtime	271.222	7.631	43.329	16.105
F18	Mean	0.0068943694819713	0.0011498935321349	0.0000000000000000	0.0048193578543185
	SD	0.0080565201649587	0.0036449413521107	0.0000000000000001	0.0133238235582874
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	73.895	2.647	19.073	6.914
F19	Mean	−3.8627821478207500	−3.7243887744664700	−3.8627821478207500	−3.8627821478207500
	SD	0.0000000000000027	0.5407823545193820	0.0000000000000024	0.0000000000000027
	Best	−3.8627821478207600	−3.8627821478207600	−3.8627821478207600	−3.8627821478207600
	Runtime	19.280	21.881	12.613	7.509
F20	Mean	−3.3180320675402500	−3.2942534432762600	−3.3219951715842400	−3.2982165473202600
	SD	0.0217068148263721	0.0511458075926848	0.0000000000000014	0.0483702518391572
	Best	−3.3219951715842400	−3.3219951715842400	−3.3219951715842400	−3.3219951715842400
	Runtime	26.209	7.333	13.562	8.008
F21	Mean	0.0003074859878056	0.0064830287538208	0.0004414866359626	0.0003685318137604
	SD	0.0000000000000000	0.0148565973286009	0.0000568392289725	0.0002323173367683
	Best	0.0003074859878056	0.0003074859878056	0.0003230956007045	0.0003074859878056
	Runtime	84.471	13.864	20.255	7.806
F22	Mean	−1.0809384421344400	−0.7323679641701760	−1.0809384421344400	−1.0764280762657400
	SD	0.0000000000000006	0.4136688304155380	0.0000000000000008	0.0247042912888477
	Best	−1.0809384421344400	−1.0809384421344400	−1.0809384421344400	−1.0809384421344400
	Runtime	27.372	32.311	27.546	19.673
F23	Mean	−1.3891992200744600	−0.5235864386288060	−1.4999990070800800	−1.3431399432579700
	SD	0.2257194403158630	0.2585330714077300	0.0000008440502079	0.2680292304904580
	Best	−1.4999992233524900	−0.7977041047646610	−1.4999992233524900	−1.4999992233524900
	Runtime	33.809	17.940	37.986	20.333
F24	Mean	−0.9166206788680230	−0.3105071678265780	−0.8406348096500680	−0.8827152798835760
	SD	0.3917752367440500	0.2080317241440800	0.2000966365984320	0.3882445165494030
	Best	−1.5000000000003800	−0.7976938356122860	−1.4999926800631400	−1.5000000000003800
	Runtime	110.798	8.835	38.470	21.599

Table 4 continued

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F25	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000004	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000003	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000001	0.0000000000000000
	Runtime	25.358	1.340	19.689	1.142
F26	Mean	−1.8210436836776800	−1.7829268228561700	−1.8210436836776800	−1.8210436836776800
	SD	0.0000000000000009	0.1450583631808370	0.0000000000000009	0.0000000000000009
	Best	−1.8210436836776800	−1.8210436836776800	−1.8210436836776800	−1.8210436836776800
	Runtime	19.154	26.249	17.228	9.663
F27	Mean	−4.6565646397053900	−4.1008953007033700	−4.6934684519571100	−4.6893456932617100
	SD	0.0557021530063238	0.4951250481844850	0.0000000000000009	0.0125797149251589
	Best	−4.6934684519571100	−4.6934684519571100	−4.6934684519571100	−4.6934684519571100
	Runtime	38.651	10.956	17.663	14.915
F28	Mean	−8.9717330307549300	−7.6193507368464700	−9.6601517156413500	−9.6397230986132500
	SD	0.4927013165009220	0.7904830398850970	0.0000000000000008	0.0393668145094111
	Best	−9.5777818097208200	−9.1383975057875100	−9.6601517156413500	−9.6601517156413500
	Runtime	144.093	6.959	27.051	20.803
F29	Mean	0.0119687224560441	0.0788734736114700	0.0838440014038032	0.0154105130055856
	SD	0.0385628598040034	0.1426911799629180	0.0778327303965192	0.0308963906374663
	Best	0.0000044608370213	0.0000000000000000	0.0129834451730589	0.0000000000000000
	Runtime	359.039	17.056	60.216	35.044
F30	Mean	0.0000130718912008	0.0000000000000000	0.0002604330013462	0.0000000000000001
	SD	0.0000014288348929	0.0000000000000000	0.0000394921919294	0.0000000000000002
	Best	0.0000095067504097	0.0000000000000000	0.0001682411286088	0.0000000000000000
	Runtime	567.704	14.535	215.722	194.117
F31	Mean	0.0001254882834238	0.0000000000000000	0.0077905311094958	0.0020185116261490
	SD	0.0001503556280087	0.0000000000000000	0.0062425841086448	0.0077448684015362
	Best	0.0000000156460198	0.0000000000000000	0.0003958766023752	0.0000000000000000
	Runtime	250.248	12.062	34.665	48.692
F32	Mean	0.0003548345513179	0.0701619169853449	0.0250163252527030	0.0013010316180679
	SD	0.0001410817500914	0.0288760292572957	0.0077209314806873	0.0009952078711752
	Best	0.0001014332605364	0.0299180701536354	0.0094647580732654	0.0001787238105452
	Runtime	290.669	2.154	34.982	82.124
F33	Mean	25.6367602258676000	95.9799861204982000	0.0000000000000000	1.1276202647057400
	SD	8.2943512684216700	56.6919245985100000	0.0000000000000000	1.0688393637536800
	Best	12.9344677422129000	29.8487565993415000	0.0000000000000000	0.0000000000000000
	Runtime	76.083	2.740	4.090	7.635
F34	Mean	2.6757043114269700	0.3986623855035210	0.2856833465904130	1.0630996944802500
	SD	12.3490058210004000	1.2164328621946200	0.6247370987465170	1.7930895051734300
	Best	0.0042535368984501	0.0000000000000000	0.0004266049929880	0.0000000000000000
	Runtime	559.966	9.462	35.865	23.278
F35	Mean	0.0000000000000000	0.4651202457398910	0.0000000000000000	0.0038863639514140
	SD	0.0000000000000000	0.0933685176073728	0.0000000000000000	0.0048411743884718
	Best	0.0000000000000000	0.0097159098775144	0.0000000000000000	0.0000000000000000
	Runtime	18.163	24.021	7.861	4.216
F36	Mean	−7684.6104757783800000	−6835.1836730901400000	−12569.4866181730000	−12304.9743375341000
	SD	745.3954005014180000	750.7338055436110000	0.0000000000022659	221.4322514436480000
	Best	−8912.8855854978200000	−8340.0386911070600000	−12569.4866181730000	−12569.4866181730000
	Runtime	307.427	3.174	19.225	10.315

Table 4 continued

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F37	Mean	0.0000000000000000	0.0000000000000000	14.5668734126948000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	8.7128443012950300	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	4.0427699323673400	0.0000000000000000
	Runtime	543.180	3.370	111.841	19.307
F38	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000005	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000001	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000003	0.0000000000000000
	Runtime	163.188	2.558	20.588	1.494
F39	Mean	−10.1061873621653000	−5.2607563471326400	−10.5364098166920000	−10.3130437162426000
	SD	1.6679113661236400	3.6145751818694000	0.0000000000000023	1.2234265179812200
	Best	−10.5364098166921000	−10.5364098166921000	−10.5364098166920000	−10.5364098166921000
	Runtime	31.018	11.024	16.015	8.345
F40	Mean	−9.5373938082045500	−5.7308569926624600	−10.1531996790582000	−9.5656135761215700
	SD	1.9062127067994200	3.5141202468383400	0.0000000000000055	1.8315977756329900
	Best	−10.1531996790582000	−10.1531996790582000	−10.1531996790582000	−10.1531996790582000
	Runtime	25.237	11.177	11.958	7.947
F41	Mean	−10.4029405668187000	−6.8674070870953700	−10.4029405668187000	−9.1615813354737300
	SD	0.0000000000000018	3.6437803702691000	0.0000000000000006	2.8277336448396200
	Best	−10.4029405668187000	−10.4029405668187000	−10.4029405668187000	−10.4029405668187000
	Runtime	21.237	11.482	14.911	8.547
F42	Mean	−186.7309073569880000	−81.5609772893002000	−186.730908831024000	−186.730908831024000
	SD	0.0000046401472660	66.4508342743478000	0.0000000000000236	0.0000000000000388
	Best	−186.7309088310240000	−186.7309088310240000	−186.730908831024000	−186.730908831024000
	Runtime	19.770	25.225	13.342	8.213
F43	Mean	−1.0316284534898800	−1.0044229658530100	−1.0316284534898800	−1.0316284534898800
	SD	0.0000000000000005	0.1490105926664260	0.0000000000000005	0.0000000000000005
	Best	−1.0316284534898800	−1.0316284534898800	−1.0316284534898800	−1.0316284534898800
	Runtime	16.754	24.798	11.309	7.147
F44	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000004	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000001	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000003	0.0000000000000000
	Runtime	159.904	2.321	21.924	1.424
F45	Mean	2.3000000000000000	0.0666666666666667	0.0000000000000000	0.9000000000000000
	SD	1.8597367258983700	0.2537081317024630	0.0000000000000000	3.0211895350832500
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	57.276	1.477	1.782	2.919
F46	Mean	0.1333333333333330	0.2666666666666670	0.0000000000000000	0.0000000000000000
	SD	0.3457459036417600	0.9444331755018490	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	20.381	2.442	1.700	1.074
F47	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000005	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000003	0.0000000000000000
	Runtime	564.178	2.565	24.172	1.870
F48	Mean	−50.0000000000002000	−50.0000000000002000	−49.9999999999997000	−50.0000000000002000
	SD	0.0000000000000361	0.0000000000000268	0.0000000000001408	0.0000000000000354
	Best	−50.0000000000002000	−50.0000000000002000	−50.0000000000001000	−50.0000000000002000
	Runtime	24.627	8.337	22.480	8.623

Table 4 continued

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F49	Mean	−210.0000000000010000	−210.0000000000030000	−209.999999999947000	−210.000000000003000
	SD	0.0000000000009434	0.0000000000003702	0.0000000000138503	0.0000000000008251
	Best	−210.0000000000030000	−210.0000000000030000	−209.999999999969000	−210.000000000004000
	Runtime	48.580	5.988	36.639	11.319
F50	Mean	0.0000000000000000	0.0000000000000000	0.0000000402380424	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000002203520334	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000210	0.0000000000000000
	Runtime	86.369	1.868	86.449	1.412

Problem	Statistics	CLPSO	SADE	BSA	IA
F1	Mean	1.8209961275956800	0.9980038377944500	0.9980038377944500	0.9980038690000000
	SD	1.6979175079427900	0.0000000000000000	0.0000000000000000	0.0000000000000035
	Best	0.9980038377944500	0.9980038377944500	0.9980038377944500	0.9980038685998520
	Runtime	61.650	66.633	38.125	43.535
F2	Mean	3.0000000000000700	2.9999999999999200	2.9999999999999200	3.0240147900000000
	SD	0.0000000000007941	0.0000000000000020	0.0000000000000011	0.0787814840000000
	Best	2.9999999999999200	2.9999999999999200	2.9999999999999200	3.0029461118668700
	Runtime	24.784	28.699	7.692	41.343
F3	Mean	0.0000000000000000	0.0034556340083499	0.0000000000000000	0.3536752140000000
	SD	0.0000000000000000	0.0189272869685522	0.0000000000000000	1.4205454130000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0014898619035614
	Runtime	38.484	15.992	18.922	34.494
F4	Mean	0.0000000000000000	0.0440448539086004	0.0000000000000000	0.0179485820000000
	SD	0.0000000000000000	0.2227372747439610	0.0000000000000000	0.0526650620000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000165491
	Runtime	48.667	33.019	24.309	322.808
F5	Mean	0.1863456353861950	0.7915368220335460	0.0000000000000105	0.0000000000000009
	SD	0.4389839299322230	0.7561593402959740	0.0000000000000034	0.0000000000000000
	Best	0.0000000000000080	0.0000000000000044	0.0000000000000080	0.0000000000000009
	Runtime	45.734	40.914	14.396	49.458
F6	Mean	0.0000444354499943	0.0000000000000000	0.0000000000000000	0.0082236060000000
	SD	0.0001015919507724	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0082236059357692
	Runtime	125.839	4.544	0.962	50.246
F7	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	2.926	4.409	0.825	38.506
F8	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	2.891	4.417	0.824	39.023
F9	Mean	0.0000193464326398	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000846531630676	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	33.307	4.303	0.829	40.896
F10	Mean	0.0006005122443674	0.0000000000000000	0.0000000000000000	0.8346587090000000
	SD	0.0029861918862801	0.0000000000000000	0.0000000000000000	0.0000000000000005
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.8346587086917530

Table 4 continued

Problem	Statistics	CLPSO	SADE	BSA	IA
	Runtime	28.508	4.371	0.790	39.978
F11	Mean	0.3978873577297390	0.3978873577297380	0.3978873577297380	0.4156431270000000
	SD	0.0000000000000049	0.0000000000000000	0.0000000000000000	0.0406451050000000
	Best	0.3978873577297380	0.3978873577297380	0.3978873577297380	0.4012748152492080
	Runtime	17.283	27.981	5.450	40.099
F12	Mean	0.1593872502094070	0.0000000000000000	0.0000000000000000	0.0014898620000000
	SD	0.6678482786713720	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000094069599934	0.0000000000000000	0.0000000000000000	0.0082029783984983
	Runtime	166.965	4.405	2.460	48.067
F13	Mean	0.0023282133668190	0.6666666666666670	0.6444444444444440	0.2528116640000000
	SD	0.0051792840882291	0.0000000000000000	0.1217161238900370	0.0000000006509080
	Best	0.0000120708732167	0.6666666666666670	0.0000000000000000	0.2528116633611470
	Runtime	216.261	47.833	21.192	67.463
F14	Mean	−1.0000000000000000	−1.0000000000000000	−1.0000000000000000	−0.9997989620000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000167151
	Best	−1.0000000000000000	−1.0000000000000000	−1.0000000000000000	−0.9997989624626810
	Runtime	16.910	28.739	5.451	39.685
F15	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	4.030	6.020	2.067	38.867
F16	Mean	81.7751618148164000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	379.9241117377270000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	162.941	5.763	7.781	48.262
F17	Mean	0.8530843976878610	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	2.9208253191698800	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0016957837829822	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	268.894	168.310	33.044	69.060
F18	Mean	0.0000000000000000	0.0226359326967139	0.0004930693556077	0.0000000000000000
	SD	0.0000000000000000	0.0283874287215679	0.0018764355751644	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	14.864	25.858	5.753	2.717
F19	Mean	−3.8627821478207500	−3.8627821478207500	−3.8627821478207500	−3.8596352620000000
	SD	0.0000000000000027	0.0000000000000027	0.0000000000000027	0.0033967610000000
	Best	−3.8627821478207600	−3.8627821478207600	−3.8627821478207600	−3.8613076574052300
	Runtime	17.504	24.804	6.009	46.167
F20	Mean	−3.3219951715842400	−3.3140689634962500	−3.3219951715842400	−2.5710247593206100
	SD	0.0000000000000013	0.0301641516823498	0.0000000000000013	0.0000000000000009
	Best	−3.3219951715842400	−3.3219951715842400	−3.3219951715842400	−2.5710247593206100
	Runtime	20.099	33.719	6.822	59.083
F21	Mean	0.0003100479704151	0.0003074859878056	0.0003074859878056	0.0016993410000000
	SD	0.0000059843325073	0.0000000000000000	0.0000000000000000	0.0000013058400000
	Best	0.0003074859941292	0.0003074859878056	0.0003074859878056	0.0016989914552560
	Runtime	156.095	45.443	11.722	48.920
F22	Mean	−1.0202940450426400	−1.0809384421344400	−1.0809384421344400	−1.4315374190000000
	SD	0.1190811583120530	0.0000000000000005	0.0000000000000005	0.0000000000000009
	Best	−1.0809384421344400	−1.0809384421344400	−1.0809384421344400	−1.4315374193830000

Table 4 continued

Problem	Statistics	CLPSO	SADE	BSA	IA
	Runtime	52.853	36.659	21.421	34.714
F23	Mean	−1.4765972735526500	−1.4999992233525000	−1.4821658762555300	−1.5000000000000000
	SD	0.1281777579497830	0.0000000000000009	0.0976772648082733	0.0000000000000000
	Best	−1.4999992233524900	−1.4999992233524900	−1.4999992233524900	−1.5000000000000000
	Runtime	42.488	36.037	18.930	41.848
F24	Mean	−0.9431432797743700	−1.2765515661973800	−1.3127183561646500	−1.5000000000000000
	SD	0.3184175870987750	0.3599594108130040	0.3158807699946290	0.0000000000000000
	Best	−1.5000000000003800	−1.5000000000003800	−1.5000000000003800	−1.5000000000000000
	Runtime	124.609	47.171	35.358	54.651
F25	Mean	0.0000041787372626	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000161643637543	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	31.632	4.090	0.813	35.662
F26	Mean	−1.8210436836776800	−1.8210436836776800	−1.8210436836776800	−1.8203821100000000
	SD	0.0000000000000009	0.0000000000000009	0.0000000000000009	0.0000000000000014
	Best	−1.8210436836776800	−1.8210436836776800	−1.8210436836776800	−1.8203821095139300
	Runtime	18.091	28.453	7.472	34.891
F27	Mean	−4.6920941990586400	−4.6884965299983800	−4.6934684519571100	−3.2820108350000000
	SD	0.0075270931220834	0.0272323381095561	0.0000000000000008	0.0000000000000023
	Best	−4.6934684519571100	−4.6934684519571100	−4.6934684519571100	−3.2820108345268900
	Runtime	25.843	38.446	11.971	45.085
F28	Mean	−9.6400278592589600	−9.6572038232921700	−9.6601517156413500	−6.2086254390000000
	SD	0.0437935551332868	0.0105890022905617	0.0000000000000007	0.0000000000000027
	Best	−9.6601517156413500	−9.6601517156413500	−9.6601517156413500	−6.2086254392105500
	Runtime	32.801	46.395	22.250	71.652
F29	Mean	0.0198686590210374	0.0140272066690658	0.0007283694780796	1.3116221610000000
	SD	0.0613698943155661	0.0328868042987376	0.0014793717464195	0.5590904820000000
	Best	0.0000175219764526	0.0000000000000000	0.0000000000000000	1.0960146962658900
	Runtime	316.817	92.412	191.881	34.697
F30	Mean	0.0458769685199585	0.0000002733806735	0.0000000028443186	0.0000000000000000
	SD	0.0620254411839524	0.0000001788830279	0.0000000033308990	0.0000000000000000
	Best	0.0005277712020642	0.0000000944121661	0.0000000004769768	0.0000000000000000
	Runtime	252.779	360.380	144.784	153.221
F31	Mean	0.0002674563703837	0.0000000000000000	0.0000000111676630	0.0071082040000000
	SD	0.0003044909265796	0.0000000000000000	0.0000000184322163	0.0000000000000000
	Best	0.0000023064754605	0.0000000000000000	0.0000000000000000	0.0071082039505830
	Runtime	227.817	220.886	149.882	43.098
F32	Mean	0.0019635752485802	0.0016730768406953	0.0019955316015528	0.0002254250000000
	SD	0.0043423828633839	0.0007330246909835	0.0009698942217908	0.0005270410000000
	Best	0.0004206447422138	0.0005630852254632	0.0006084880639553	0.0000023800831017
	Runtime	103.283	171.637	48.237	218.722
F33	Mean	0.6301407361590880	0.8622978494808570	0.0000000000000000	0.0000000000000000
	SD	0.8046401822326410	0.9323785263847000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	18.429	23.594	5.401	2.266
F34	Mean	5.7631786582751800	1.2137377447007000	0.3986623854300930	0.0000154715000000
	SD	13.9484817304201000	1.8518519388285700	1.2164328622195200	0.0000022373400000
	Best	0.0268003205820685	0.0001448955835246	0.0000000000000000	0.0000118803557196

Table 4 continued

Problem	Statistics	CLPSO	SADE	BSA	IA
	Runtime	187.894	268.449	34.681	7.250
F35	Mean	0.0019431819755029	0.0006477273251676	0.0000000000000000	0.0000000000000000
	SD	0.0039528023354469	0.0024650053428137	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	8.304	5.902	1.779	33.155
F36	Mean	−12210.8815698372000	−12549.746895737300000	−12569.486618173000000	−12569.3622100000000000
	SD	205.9313376284770000	44.8939348779747000	0.0000000000024122	0.0000000273871000
	Best	−12569.4866181730000	−12569.486618173000000	−12569.486618173000000	−12569.3622054081000000
	Runtime	31.499	34.383	11.069	2.306
F37	Mean	6.4655746330439100	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	8.2188901353055800	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.1816624029553790	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	179.083	109.551	57.294	100.947
F38	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	12.563	5.627	3.208	47.009
F39	Mean	−10.3130437162026000	−10.5364098166921000	−10.5364098166921000	−10.5063235800000000
	SD	1.2234265179736500	0.0000000000000016	0.0000000000000018	0.0000000025211900
	Best	−10.5364098166920000	−10.5364098166921000	−10.5364098166920000	−10.5063235792920000
	Runtime	37.275	28.031	7.045	55.666
F40	Mean	−10.1531996790582000	−9.9847854277673500	−10.1531996790582000	−10.1529842600000000
	SD	0.0000000000000076	0.9224428443735560	0.0000000000000072	0.0000000000542921
	Best	−10.1531996790582000	−10.1531996790582000	−10.1531996790582000	−10.1529842649756000
	Runtime	30.885	25.569	6.864	51.507
F41	Mean	−10.4029405668187000	−10.4029405668187000	−10.4029405668187000	−10.3988303400000000
	SD	0.0000000000000010	0.0000000000000018	0.0000000000000017	0.0000000001978980
	Best	−10.4029405668187000	−10.4029405668187000	−10.4029405668187000	−10.3988303385534000
	Runtime	31.207	27.064	8.208	53.190
F42	Mean	−186.730908831024000	−186.7309088310240000	−186.7309088310240000	−186.2926481000000000
	SD	0.0000000000000279	0.0000000000000377	0.0000000000000224	0.0000000000000578
	Best	−186.730908831024000	−186.7309088310240000	−186.7309088310240000	−186.2926480689880000
	Runtime	20.344	27.109	9.002	31.766
F43	Mean	−1.0316284534898800	−1.0316284534898800	−1.0316284534898800	−1.0304357800000000
	SD	0.0000000000000005	0.0000000000000005	0.0000000000000005	0.0014911900000000
	Best	−1.0316284534898800	−1.0316284534898800	−1.0316284534898800	−1.0314500753985900
	Runtime	18.564	27.650	5.691	39.897
F44	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	14.389	5.920	3.302	174.577
F45	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000538870000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000005399890
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000538860819891
	Runtime	3.042	4.307	0.883	2.215
F46	Mean	0.2000000000000000	0.0000000000000000	0.0000000000000000	−0.0153463301609662
	SD	0.4068381021724860	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	−0.0153463301609662

Table 4 continued

Problem	Statistics	CLPSO	SADE	BSA	IA
	Runtime	6.142	4.319	0.764	31.068
F47	Mean	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	15.948	6.383	4.309	31.296
F48	Mean	−49.4789234062579000	−50.0000000000002000	−50.0000000000002000	−44.7416748700000000
	SD	1.3150773145311700	0.0000000000000268	0.0000000000000361	0.0000000000000217
	Best	−49.9999994167392000	−50.0000000000002000	−50.0000000000002000	−44.7416748706606000
	Runtime	142.106	36.804	7.747	52.486
F49	Mean	−199.592588547503000	−210.0000000000030000	−210.0000000000030000	−150.5540859185450000
	SD	9.6415263953591700	0.0000000000004625	0.0000000000003950	0.0000000000000000
	Best	−209.985867409029000	−210.0000000000040000	−210.0000000000040000	−150.5540859185450000
	Runtime	187.787	54.421	11.158	70.887
F50	Mean	0.0000000001597805	0.0000000000000000	0.0000000000000000	0.0000000000000000
	SD	0.0000000006266641	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Runtime	157.838	4.930	5.702	33.573

prove that IA works better than PSO. The self-interested behaviour of every individual in IA enables them to communicate with each other in order to seek for better solutions. They respond adaptively to the shape of the fitness landscape. Thus, IA is able to achieve higher convergence rate in the iterative processes. It is because the efforts of improving the best solution depend on not only the current position of the particle itself but also the position of the global best individual $(L_{p,\mathrm{gb}})$, local best individual $(L_{p,b})$ and the local second best individual $(L_{p,2b})$. This can prevent the problem of falling into local optimum in high-dimensional space, which is the common problem faced by most of the EAs.

3.6 CMAES versus IA

The CMAES algorithm stands for covariance matrix adaptation evolution strategy. It is a mathematical-based algorithm that makes use of adaptive mutation parameters through computing a covariance matrix as shown in Fig. 7 [36]. One major drawback of CMAES is the cost in calculating the covariance matrix. The cost increases rapidly with increasing dimensions. Plus, sampling using a multivariate normal distribution and factorization of the covariance matrix also becomes increasingly expensive [48].

The IA is being compared with classical CMAES in this work. The relatively simpler structure of IA as compared with CMAES leads to the successful of IA in solving unconstrained benchmark problem in terms of runtime as shown in Tables 6 and 7. Overall the convergence speed of IA is higher than CMAES.

3.7 ABC versus IA

In ABC algorithm, the artificial bee colony is made up of employed bees, onlooker bees and scout bees. An onlooker bee waits on the dance area for making decision in choosing a food source. An employed bee goes to the previously visited food source to search for food. A scout bee carries out random search [41]. The working mechanism of ABC is described in Fig. 8.

An existing challenge to all stochastic optimization methods is the balance between exploration and exploitation. A poor optimization will meet the problems of premature convergence and get trapped from local minima. Meanwhile, excessively exploitative will cause the algorithm to converge very slowly. ABC is good at exploration but poor at exploitation; its convergence speed is also an issue in some cases [50]. The results of the proposed IA are being compared with ABC. Results from Table 6 denote that IA works equally well as ABC. However, Table 7 shows that IA has a superior performance as compared with ABC. This proves that IA works better in dealing with high-performance and more complicated benchmark functions in Test 2. Table 8 also proves that IA outperforms ABC as indicated in p values as well as T− and T+ values.

Table 5 Statistical solutions obtained by PSO, CMAES, ABC, CLPSO, SADE, BSA and proposed IA in Test 2 (*mean* mean solution, *SD* standard deviation of mean solution, *best* best solution, *runtime* mean runtime in seconds)

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F51	Mean	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	0.0000000000000000
	Best	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000
	Runtime	212.862	23.146	113.623	118.477
F52	Mean	−450.0000000000000000	−450.0000000000000000	−449.9999999999220000	−450.0000000000000000
	SD	0.0000000000000350	0.0000000000000000	0.0000000002052730	0.0000000000000615
	Best	−450.0000000000000000	−450.0000000000000000	−449.9999999999970000	−450.0000000000000000
	Runtime	230.003	23.385	648.784	139.144
F53	Mean	−44.5873911956554000	−450.0000000000000000	387131.24412139700000	−197.9999999999850000
	SD	458.5794120016290000	0.0000000000000000	166951.73365926400000	391.5169437474990000
	Best	−443.9511286079800000	−450.0000000000000000	165173.18530956000000	−449.9999999999990000
	Runtime	2658.937	35.464	240.094	1017.557
F54	Mean	−450.0000000000000000	77982.4567046980000000	140.4509447125110000	−414.0000000000000000
	SD	0.0000000000000460	131376.7365456010000000	217.2646715063190000	55.9309919639279000
	Best	−450.0000000000000000	−450.0000000000000000	−324.3395691109350000	−450.0000000000000000
	Runtime	247.256	32.726	209.188	143.767
F55	Mean	−310.0000000000000000	−310.0000000000000000	−291.5327549384120000	−271.0000000000000000
	SD	0.0000000000000000	0.0000000000000000	17.6942171217937000	60.5919079609218000
	Best	−310.0000000000000000	−310.0000000000000000	−307.7611364354020000	−310.0000000000000000
	Runtime	241.517	39.293	205.568	134.078
F56	Mean	393.4959999056240000	390.5315438816460000	391.2531452421960000	231.3986579112350000
	SD	16.0224965900462000	1.3783433976378300	3.7254660805238600	247.2968415284400000
	Best	390.0000000000150000	390.0000000000000000	390.0101471658490000	−140.0000000000000000
	Runtime	1178.079	27.894	159.762	153.715
F57	Mean	1091.0644335162500000	1087.2645466786700000	1087.0459486286000000	1141.0459486286000000
	SD	3.4976948942723200	0.5365230018001780	0.0000000000005585	83.8964879458918000
	Best	1087.0696772583000000	1087.0459486286000000	1087.0459486286000000	1087.0459486286000000
	Runtime	334.064	37.047	180.472	159.922
F58	Mean	−119.8190232990920000	−119.9261073509850000	−119.7446063439080000	−119.4450938018030000
	SD	0.0720107560874199	0.1554021446157740	0.0623866434489108	0.0927418223065644
	Best	−119.9302772694110000	−120.0000000000000000	−119.8779554779730000	−119.6575717927190000
	Runtime	602.507	49.209	265.319	160.806
F59	Mean	−324.6046006320200000	−306.5782069681560000	−330.0000000000000000	−329.8673387923880000
	SD	2.5082306041521000	21.9475396048756000	0.0000000000000000	0.3440030182812760
	Best	−329.0050409429070000	−327.0151228287200000	−330.0000000000000000	−330.0000000000000000
	Runtime	982.449	22.237	111.629	128.494
F60	Mean	−324.3311322538170000	−314.7871102989330000	−306.7949047862760000	−319.6763749798700000
	SD	3.0072222933667300	8.3115989308305500	5.1787864195870400	4.9173541245304800
	Best	−327.1650513120000000	−327.0151228287200000	−318.9403196374510000	−326.0201637716270000
	Runtime	1146.013	29.860	259.258	179.039
F61	Mean	92.5640111212146000	90.7642785704506000	94.8428485804138000	93.2972315784963000
	SD	1.5827416781636900	26.4613831425879000	0.6869412813090850	1.8766951726453600
	Best	90.1142082473923000	−45.0054133586912000	93.1500794016147000	91.0295373630387000
	Runtime	1310.457	44.217	308.501	282.150
F62	Mean	18611.314225480900000	−70.0486708747625000	−337.3273080760500000	400.3240208136310000
	SD	12508.786612631600000	637.4585182420270000	56.5730759032367000	688.3344299264300000
	Best	4568.3350537809200000	−460.0000000000000000	−449.1707421778360000	−434.8788220982740000
	Runtime	2381.974	34.857	232.916	202.941

Table 5 continued

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F63	Mean	−129.2373581503910000	−128.7850616923410000	−129.8343428775830000	−129.6294851450880000
	SD	0.5986210944493790	0.6157633658946230	0.0408016481905455	0.1054759371085400
	Best	−129.6861385930680000	−129.5105509483130000	−129.9098920058450000	−129.8125711770830000
	Runtime	2183.218	25.496	205.194	186.347
F64	Mean	−298.2835926212850000	−295.1290938304830000	−296.9323391084610000	−296.8839733969750000
	SD	0.5587676271753680	0.1634039984609270	0.2251930667702880	0.4330673614598290
	Best	−299.6022022972560000	−295.7382222729600000	−297.4659619544820000	−297.8411886637500000
	Runtime	2517.138	32.084	262.533	334.888
F65	Mean	417.4613663019860000	492.5045364088000000	120.0000000000000000	326.6601114362900000
	SD	153.9215808771580000	181.5709657779580000	0.0000000000000188	174.6877238188330000
	Best	120.0000000000000000	262.7619554120320000	120.0000000000000000	120.0000000000000000
	Runtime	3156.336	239.823	2285.787	1834.967
	NFE				
F66	Mean	221.4232628350220000	455.1151684594550000	258.8582688922670000	231.1806131539990000
	SD	12.2450207482898000	254.3583511786970000	11.8823213189685000	13.5473380962764000
	Best	181.5746616282570000	120.0000000000000000	235.6600739998890000	210.3582705649860000
	Runtime	4242.280	202.808	2237.308	1824.388
F67	Mean	217.3338617866620000	681.0349114021570000	265.0370119084380000	228.7309024901770000
	SD	20.6685850658838000	488.0618274343640000	12.4033917090208000	12.3682716268631000
	Best	120.0000000000000000	223.0782617790520000	241.9810089596350000	181.6799927773160000
	Runtime	8208.697	197.497	2159.392	5873.112
F68	Mean	668.9850326105730000	926.9488078829420000	513.8925774904480000	743.9859973770210000
	SD	275.8071370273340000	174.1027182659660000	31.0124861524005000	175.6497294240330000
	Best	310.0000000000000000	310.0000000000000000	444.4692044973030000	310.0000000000000000
	Runtime	3687.235	251.155	2445.259	1777.638
F69	Mean	708.2979222913040000	831.2324139697050000	500.5478931040730000	776.5150806087790000
	SD	256.2419561521300000	250.1848775931620000	31.2240894705539000	160.7307526692470000
	Best	310.0000000000000000	310.0000000000000000	407.3155842366960000	363.8314566805740000
	Runtime	5258.509	222.015	2341.791	1849.670
F70	Mean	711.2970397614200000	876.9306188768990000	483.2984167460740000	761.2954767038960000
	SD	258.9317052508320000	289.7296413284470000	99.3976740616107000	163.4084080635650000
	Best	310.0000000000000000	310.0000000000000000	155.5049931377980000	363.8314568648180000
	Runtime	4346.055	228.619	2250.917	1900.279
F71	Mean	1117.8857079625100000	1258.1065536572400000	659.5351969346130000	959.3735119754180000
	SD	311.0011859260640000	359.7382897536570000	98.5410511961986000	240.5568407069990000
	Best	560.0000000000000000	660.0000000000000000	560.0001912324020000	660.0000000000000000
	Runtime	3012.883	241.541	2728.060	1573.484
	NFE				
F72	Mean	1094.8305116977000000	−7.159E+49	915.4958100611630000	1133.7536009808600000
	SD	121.3539576317800000	4.387E+50	242.1993331983530000	42.1171260000361000
	Best	660.0000000000000000	−133.9585340104890000	660.0006867770510000	1088.9543269392600000
	Runtime	6363.267	290.334	2326.112	1730.723
F73	Mean	1304.3661550124000000	1159.9280867973000000	830.2290165794410000	1167.9040488743800000
	SD	262.1065863453340000	742.1215416320490000	60.2286903507069000	236.7325108248320000
	Best	919.4683107913200000	−460.7504508023100000	785.1725102979490000	785.1725102979490000
	Runtime	2165.640	238.261	2045.582	1580.067

Table 5 continued

Problem	Statistics	PSO2011	CMAES	ABC	JDE
F74	Mean	500.0000000000000000	653.3355378428050000	460.0000000000020000	510.0000000000000000
	SD	103.7237710925280000	302.5312999719650000	0.0000000000016493	113.7147065368360000
	Best	460.0000000000000000	460.0000000000000000	460.0000000000000000	460.0000000000000000
	Runtime	1811.980	165.962	1698.121	1366.710
F75	Mean	1107.9038127876700000	1401.6553278264300000	930.4565414149210000	1072.9924659809200000
	SD	127.9566489362040000	253.2428066220210000	87.9959072391079000	2.2606058314671500
	Best	1069.5511765775700000	1072.4973401423200000	862.4476004191700000	1068.5560012648600000
	Runtime	4060.091	214.580	2113.339	2951.018

Problem	Statistics	CLPSO	SADE	BSA	IA
F51	Mean	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000	−447.6018854297170000
	SD	0.0000000000000000	0.0000000000000000	0.0000000000000000	89.3142986500000000
	Best	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000
	Runtime	167.675	154.232	140.736	30.282
F52	Mean	−418.8551838547760000	−450.0000000000000000	−450.0000000000000000	−449.9967727000000000
	SD	51.0880511039985000	0.0000000000000000	0.0000000000000259	0.0176705780000000
	Best	−449.4789299923810000	−450.0000000000000000	−450.0000000000000000	−450.0000000000000000
	Runtime	1462.706	185.965	243.657	48.003
F53	Mean	62142.8213760465000000	245.0483283713550000	−449.9999567867430000	−449.7873452000000000
	SD	34796.1785167236000000	790.6056596723160000	0.0001175386756044	0.0000000000001734
	Best	17306.9066792474000000	−421.4054944641620000	−450.0000000000000000	−450.0000000000000000
	Runtime	1789.643	1808.954	1883.713	52.463
F54	Mean	−178.8320689185280000	−450.0000000000000000	−450.0000000000000000	−388.7807630000000000
	SD	394.8667499339530000	0.0000000000000000	0.0000000000000259	1.1928333530000000
	Best	−447.9901256558030000	−450.0000000000000000	−450.0000000000000000	−389.7573633109500000
	Runtime	1248.616	185.438	347.167	46.072
F55	Mean	333.4108259915760000	−309.9999999999960000	−309.9999999999980000	−310.8207993000000000
	SD	512.6920837704510000	0.0000000000133965	0.0000000000023443	0.0208030240000000
	Best	−309.9740055344430000	−310.0000000000000000	−310.0000000000000000	−310.8367924750510000
	Runtime	1481.686	210.684	386.633	44.84710031
F56	Mean	405.5233436479650000	390.2657719408230000	390.1328859704120000	390.8036739982730000
	SD	10.7480096852869000	1.0114275384776600	0.7278464357038200	0.0000000000000000
	Best	390.5776683413440000	390.0000000000000000	390.0000000000000000	390.8036739982730000
	Runtime	1441.859	1214.303	290.236	45.632
F57	Mean	1087.0459486286000000	1087.0459486286000000	1087.0459486286000000	1087.2265890000000000
	SD	0.0000000000004264	0.0000000000004814	0.0000000000004428	0.0019192200000000
	Best	1087.0459486286000000	1087.0459486286000000	1087.0459486286000000	1087.2262037455100000
	Runtime	267.342	259.760	332.132	52.621
F58	Mean	−119.9300269839980000	−119.7727713703720000	−119.8356122057440000	−119.6006412865410000
	SD	0.0417913553101429	0.1248514853682450	0.0704515460477787	0.0000000000000434
	Best	−119.9756745390830000	−119.9999999999980000	−119.9802847896350000	−119.6006412865410000
	Runtime	1586.286	648.489	717.375	52.56165118
F59	Mean	−329.4361898676470000	−329.9668346980970000	−330.0000000000000000	−327.1635938000000000
	SD	0.6229063711904190	0.1816538397880230	0.0000000000000000	0.0000000000001156
	Best	−330.0000000000000000	−330.0000000000000000	−330.0000000000000000	−327.163593801473
	Runtime	162.873	155.645	176.994	45.867
F60	Mean	−321.7278926895280000	−322.9689591871600000	−319.2544515903510000	−335.0171647000000000
	SD	1.8971778613701300	2.8254645254663600	3.3091959975390800	10.6369134000000000
	Best	−326.1788303102740000	−328.0100818858130000	−325.0252097523530000	−347.2509173436740000

Table 5 continued

Problem	Statistics	CLPSO	SADE	BSA	IA
	Runtime	1594.096	210.534	420.851	54.661
F61	Mean	94.6109567642977000	91.6859083842723000	92.3519494286347000	92.0170440500000000
	SD	0.6689129174038950	0.9033073777915270	1.0901581870340800	0.000000000000014453
	Best	92.9690673344598000	90.1363685040678000	90.2628852415150000	92.0170440535006000
	Runtime	1421.545	506.829	1771.860	60.350
F62	Mean	−447.8870804905020000	−394.5206365378250000	−437.1125728026770000	−410.1361631000000000
	SD	11.8934815947019000	128.6353424718180000	20.3541618366546000	34.8795385900000000
	Best	−459.6890294276810000	−460.0000000000000000	−459.1772521346520000	−421.5672584975600000
	Runtime	1636.440	1277.975	1466.985	48.480
F63	Mean	−129.8382867796110000	−129.7129164862680000	−129.8981409848090000	−122.2126680000000000
	SD	0.0372256921835666	0.0875456568200232	0.0682328484314248	0.0000000000000434
	Best	−129.9098505660780000	−129.8717592632560000	−129.9901230990300000	−122.2126679617240000
	Runtime	1526.365	660.986	1064.114	46.260
F64	Mean	−297.5119726691150000	−297.8403738182600000	−297.5359077431460000	−295.4721554000000000
	SD	0.3440115280624180	0.4536801689800720	0.4085859316264990	0.1118191570000000
	Best	−298.3030560759620000	−299.2417795907860000	−298.3869295150680000	−295.6307146941910000
	Runtime	1615.452	1289.814	1953.289	55.118
F65	Mean	131.3550392249760000	234.2689845349590000	120.0000000000000000	120.0000000000000000
	SD	26.1407360548431000	150.7595974059750000	0.0000000000000000	0.0000000000000000
	Best	120.0000000000000000	120.0000000000000000	120.0000000000000000	120.0000000000000000
	Runtime	3210.655	1932.016	2351.478	69.052
	NFE				
F66	Mean	231.5547154800990000	222.0256674919140000	234.4843380488580000	276.3946208000000000
	SD	11.5441451076421000	6.1841489800660300	8.9091119100451100	19.2196655800000000
	Best	214.7661703584830000	206.4520786020840000	219.6244910167680000	259.8700033222460000
	Runtime	8649.998	2970.950	8270.920	252.234
F67	Mean	240.3635189964930000	221.1801916743850000	228.3769828342800000	201.0516618000000000
	SD	14.8435137485293000	5.7037006844690500	8.7086794471239900	2.4309010810000000
	Best	221.3817133141830000	209.2509748304710000	204.6479138174220000	197.8966349103590000
	Runtime	4599.027	5938.879	8189.243	254.253
F68	Mean	892.4391527217660000	845.4504613493740000	587.5732354221340000	310.0161021000000000
	SD	79.1422224454971000	120.8505129523180000	250.0556329707140000	0.0370586450000000
	Best	738.3764781625320000	310.0000000000000000	310.0000000000000000	310.0014955442130000
	Runtime	8398.690	3073.274	4554.102	253.064
F69	Mean	863.8926908090610000	809.7183195902260000	587.6511686191670000	310.0029796000000000
	SD	96.5618989087194000	147.3158109824600000	236.1141037692630000	0.0082796490000000
	Best	493.0042540796450000	310.0000000000000000	310.0000000000000000	310.0000285440690000
	Runtime	9909.479	3213.601	4764.968	291.084
F70	Mean	844.6391674419360000	810.5227124472170000	612.0906184834040000	310.0041570000000000
	SD	113.6848457105400000	104.7139423525340000	249.5599278421970000	0.0128812140000000
	Best	489.0742585970560000	310.0000000000000000	310.0000000000000000	310.0002219576930000
	Runtime	9988.261	2818.575	4945.132	268.701
F71	Mean	911.4640642691360000	990.8546718748010000	836.1411004458200000	577.7786170000000000
	SD	238.3180009803040000	235.1014092849970000	128.9346234954740000	1.8288684190000000
	Best	560.0000121795840000	660.0000000000000000	560.0000000000000000	574.8590032551840000
	Runtime	10891.124	1769.459	2972.618	279.0646913
	NFE				

Table 5 continued

Problem	Statistics	CLPSO	SADE	BSA	IA
F72	Mean	1075.5292326436900000	1094.6823697304900000	984.5106541514410000	694.3706620000000000
	SD	166.9355145236330000	87.9884000140656000	199.1563947691970000	20.9754439100000000
	Best	660.0000000000020000	660.0000000000000000	660.0000000000000000	644.2542524502140000
	Runtime	9601.880	3854.148	10458.467	273.922
F73	Mean	1070.4327462836400000	1105.2511774948600000	976.2273885425320000	559.6581705000000000
	SD	203.0676662707430000	190.6172874229610000	160.1543461970300000	16.1193896300000000
	Best	785.1725102979480000	919.4683107913240000	785.1725102979480000	546.1130231359180000
	Runtime	7459.005	1901.540	4209.110	287.271
F74	Mean	493.3333333333340000	490.0000000000000000	460.0000000000000000	463.2262530000000000
	SD	137.2973951415090000	91.5385729888094000	0.0000000000000000	4.9321910760000000
	Best	460.0000000000000000	460.0000000000000000	460.0000000000000000	458.5444354721460000
	Runtime	3016.959	1410.399	1795.637	257.960
F75	Mean	1258.5157766524700000	1074.3695435628600000	1063.7363787709700000	471.2797518000000000
	SD	241.4024507676890000	2.8314182838917800	55.8479313799755000	2.2346287190000000
	Best	871.8607884176050000	1069.8723890709000000	856.8214538442850000	469.3372925643150000
	Runtime	5262.210	3410.902	4280.901	263.829

3.8 DE versus IA

DE is a population-based algorithm which uses the similar operators as GA: crossover, mutation and selection. The only difference is that GA relies on crossover where DE relies on mutation operation. DE algorithm uses mutation operation as a search mechanism and selection operation to direct the search in the search space as shown in Eqs. (12) and (13). By creating trial vectors using the components of existing individuals in the population, the crossover operator effectively sorts information about successful combinations, enabling better solution search space [15].

$$\text{Mutation}\quad \hat{x}_i = x_{r1} + F(x_{r3} - x_{r2}),\quad F = [0, 1] \tag{12}$$

$$x_{r1}, x_{r2}, x_{r3} | r_1 \neq r_2 \neq r_3 \neq i \tag{13}$$

$$\text{Crossover}\quad y_i^j = \begin{cases} \hat{x}_i^j, R_j \leq \text{CR} \\ x_i^j, R_j > \text{CR} \end{cases},\quad R_j = [0, 1] \tag{14}$$

During mutation, the parameter $\hat{x}_i$ is mutant solution vector, while F is scaling factor and i is an index of current solution. In the stage of crossover, CR is the crossover constant, while j represents the jth component of the corresponding array. In DE, a population of solution vectors is randomly created at the start. This population is successfully improved by applying mutation, crossover and selection operators as shown in Fig. 9. In DE algorithm, each new solution produced competes with a mutant vector and the better one wins the competition. In other words, the chance of succession is independent on their fitness values. Every new solution produced competes with its parent, and the better one wins the competition [15].

In this section, IA is being compared with the variants modified based on DE, which are JDE and SADE. The IA outperforms JDE and SADE in most of the benchmark functions of Test 1 and Test 2. The statistical results in Tables 6 and 7 indicate that IA performs better than JDE and SADE.

3.9 BSA versus IA

In BSA, three basic genetic operators—selection, mutation and crossover—are used to generate trial individuals. A random mutation strategy is performed such that only one direction individual is used for each target individual. BSA randomly chooses the direction individual from a randomly chosen individual from previous generation. BSA uses a nonuniform crossover strategy that is more complex as compared with other GAs [38]. The procedure of BSA is shown in Fig. 10.

BSA is divided into five processes: initialization, selection I, mutation, crossover and selection II. In the selection II stage, the T_is that have better fitness values than the corresponding P_is are used to update the P_is based on the concept of greedy selection. If the best individual of P (P_{best}) has better fitness value than the global minimum value, the global minimizer is updated to be P_{best}. Hence, the global minimum value is updated to be the fitness value of P_{best}.

$$\text{Mutation}\quad \text{Mutant} = P + F(oldP - P) \tag{15}$$

Crossover

$$\text{map}_{n,m} = 1, T_{n,m} := P_{n,m} \begin{cases} n \in \{1, 2, 3, \ldots, N\} \\ m \in \{1, 2, 3, \ldots, D\} \end{cases} \tag{16}$$

Table 6 Statistical results for each benchmark problem in Test 1 using two-sided Wilcoxon signed-rank test ($x = 0.05$)

Problem	PSO2011 versus IA				CMAES versus IA				ABC versus IA				JDE versus IA			
	p value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner
F1	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F2	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F3	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F4	4.3205E−08	0	465	+	6.7988E−08	0	465	+	3.3465E−07	0	465	+	4.3205E−08	0	465	+
F5	1.7289E−06	0	465	+	1.7289E−06	0	465	+	1.7289E−06	465	0	−	1.7289E−06	0	465	+
F6	1.7300E−06	465	0	−	1.7300E−06	0	465	+	1.7300E−06	465	0	−	1.7300E−06	465	0	−
F7	1.7344E−06	465	0	−	1.7344E−06	0	465	+	1.7344E−06	465	0	−	1.7344E−06	465	0	−
F8	1.7279E−06	0	465	+	1.7279E−06	0	465	+	1.7279E−06	0	465	+	1.7279E−06	0	465	+
F9	1.00E+00	0	0	=	1.7279E−06	0	465	+	4.3205E−08	0	465	+	1.00E+00	0	0	=
F10	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−
F11	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F12	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	0	465	+	4.3205E−08	465	0	−
F13	1.6594E−06	0	465	+	1.6594E−06	0	465	+	1.6594E−06	465	0	−	1.6594E−06	0	465	+
F14	1.5450E−06	465	0	−	1.6657E−06	465	0	−	1.5450E−06	465	0	−	1.5450E−06	465	0	−
F15	1.0135E−07	465	0	−	1.0135E−07	0	465	+	1.0135E−07	465	0	−	1.0135E−07	465	0	−
F16	6.8714E−07	0	465	+	6.8714E−07	0	465	+	6.8714E−07	0	465	+	6.8714E−07	0	465	+
F17	1.1048E−06	0	465	+	1.1048E−06	0	465	+	1.1048E−06	0	465	+	1.1048E−06	0	465	+
F18	1.0135E−07	465	0	−	1.0135E−07	465	0	−	1.0135E−07	465	0	−	1.0135E−07	465	0	−
F19	1.2033E−06	465	0	−	1.2033E−06	0	465	+	1.2033E−06	465	0	−	1.2033E−06	465	0	−
F20	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−
F21	1.7300E−06	0	465	+	1.7300E−06	0	465	+	1.7300E−06	0	465	+	1.7300E−06	0	465	+
F22	1.6647E−06	0	465	+	1.6647E−06	0	465	+	1.6647E−06	0	465	+	1.6647E−06	0	465	+
F23	1.7279E−06	0	465	+	1.7279E−06	0	465	+	1.7279E−06	0	465	+	1.7279E−06	0	465	+
F24	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−
F25	1.00E+00	0	0	=	1.00E+00	0	0	=	4.3205E−08	0	465	+	1.00E+00	0	0	=
F26	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−
F27	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−
F28	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−	4.3205E−08	465	0	−
F29	5.9869E−07	0	465	+	5.9869E−07	0	465	+	5.9869E−07	0	465	+	5.9869E−07	0	465	+
F30	1.6976E−06	465	0	−	1.6976E−06	465	0	−	1.6976E−06	465	0	−	1.6976E−06	465	0	−
F31	1.0789E−06	0	465	+	1.0789E−06	0	465	+	1.0789E−06	0	465	+	1.0789E−06	0	465	+
F32	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F33	4.3205E−08	0	465	+	4.3205E−08	0	465	+	1.9773E−07	465	0	−	4.3205E−08	0	465	+
F34	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F35	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+

Table 6 continued

Problem	PSO2011 versus IA				CMAES versus IA				ABC versus IA				JDE versus IA			
	p value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner
F36	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F37	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F38	3.3248E−07	465	0	–	3.3248E−07	465	0	–	4.3205E−08	0	465	+	3.3248E−07	465	0	–
F39	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F40	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F41	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F42	1.9618E−07	465	0	–	1.9618E−07	465	0	–	1.9618E−07	465	0	–	1.9618E−07	465	0	–
F43	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F44	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F45	4.3205E−08	0	465	+	4.3205E−08	0	465	+	1.00E+00	0	0	=	4.3205E−08	0	465	+
F46	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F47	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F48	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F49	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F50	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
+/=/−	24/2/24				29/1/20				24/1/25				24/2/24			

Problem	CLPSO versus IA				SADE versus IA				BSA versus IA			
	p value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner
F1	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F2	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F3	1.3422E−06	465	0	–	4.3205E−08	0	465	+	1.3422E−06	465	0	–
F4	3.3465E−07	465	0	–	4.3205E−08	0	465	+	3.3465E−07	465	0	–
F5	1.7289E−06	0	465	+	1.7289E−06	0	465	+	1.7289E−06	465	0	–
F6	1.7300E−06	465	0	–	1.7300E−06	465	0	–	1.7300E−06	465	0	–
F7	1.7344E−06	465	0	–	1.7344E−06	465	0	–	1.7344E−06	465	0	–
F8	1.7279E−06	0	465	+	1.7279E−06	0	465	+	1.7279E−06	0	465	+
F9	4.3205E−08	0	465	+	1.00E+00	0	0	=	1.00E+00	0	0	=
F10	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F11	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F12	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F13	0.0566	140	325	+	1.6594E−06	0	465	+	1.6594E−06	0	465	+
F14	1.5450E−06	465	0	–	1.5450E−06	465	0	–	1.5450E−06	465	0	–
F15	1.0135E−07	465	0	–	1.0135E−07	465	0	–	1.0135E−07	465	0	–
F16	6.8714E−07	0	465	+	6.8714E−07	0	465	+	6.8714E−07	0	465	+

Table 6 continued

Problem	CLPSO versus IA				SADE versus IA				BSA versus IA			
	p value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner
F17	1.1048E−06	0	465	+	1.1048E−06	465	0	–	1.1048E−06	465	0	–
F18	1.0135E−07	465	0	–	1.0135E−07	465	0	–	1.0135E−07	465	0	–
F19	1.2033E−06	465	0	–	1.2033E−06	465	0	–	1.2033E−06	465	0	–
F20	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F21	1.7300E−06	0	465	+	1.7300E−06	0	465	+	1.7300E−06	0	465	+
F22	1.6647E−06	0	465	+	1.6647E−06	0	465	+	1.6647E−06	0	465	+
F23	1.7279E−06	0	465	+	1.7279E−06	0	465	+	1.7279E−06	0	465	+
F24	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F25	4.3205E−08	0	465	+	1.00E+00	0	0	=	1.00E+00	0	0	=
F26	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F27	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F28	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F29	5.9869E−07	0	465	+	5.9869E−07	0	465	+	5.9869E−07	465	0	–
F30	1.6976E−06	465	0	–	1.6976E−06	465	0	–	1.6976E−06	465	0	–
F31	1.0789E−06	0	465	+	1.0789E−06	0	465	+	1.0789E−06	0	465	+
F32	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F33	4.3205E−08	0	465	+	4.3205E−08	0	465	+	1.9773E−07	465	0	–
F34	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F35	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F36	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F37	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F38	3.3248E−07	465	0	–	3.3248E−07	465	0	–	3.3248E−07	465	0	–
F39	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F40	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F41	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F42	1.9618E−07	465	0	–	1.9618E−07	465	0	–	1.9618E−07	465	0	–
F43	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F44	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F45	1.00E+00	0	0	=	1.00E+00	0	0	=	1.00E+00	0	0	=
F46	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F47	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F48	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F49	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F50	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
+/=/−	25/1/24				22/3/25				17/3/30			

Table 7 Statistical results for each benchmark problem in Test 2 using two-sided Wilcoxon signed-rank test ($x = 0.05$)

Problem	PSO versus IA				CMAES versus IA				ABC versus IA				JDE versus IA			
	p value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner
F51	6.9066E−07	465	0	–	6.9066E−07	465	0	–	6.9066E−07	465	0	–	6.9066E−07	465	0	–
F52	1.1826E−06	465	0	–	1.1826E−06	465	0	–	1.1826E−06	465	0	–	1.1826E−06	465	0	–
F53	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F54	1.7333E−06	465	0	–	1.7333E−06	0	465	+	1.7333E−06	0	465	+	1.7333E−06	465	0	–
F55	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F56	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	0	465	+	4.3205E−08	465	0	–
F57	6.7988E−08	0	465	+	6.7988E−08	0	465	+	6.7988E−08	465	0	–	6.7988E−08	0	465	+
F58	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	0	465	+
F59	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F60	3.9575E−05	36	429	+	1.1567E−06	0	465	+	1.1567E−06	0	465	+	1.1567E−06	0	465	+
F61	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	0	465	+	4.3205E−08	0	465	+
F62	1.4403E−07	0	465	+	1.4403E−07	0	465	+	2.9866E−07	6	459	+	1.4403E−07	0	465	+
F63	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F64	1.5117E−06	465	0	–	1.5117E−06	0	465	+	1.5117E−06	465	0	–	1.5117E−06	465	0	–
F65	4.3205E−08	0	465	+	4.3205E−08	0	465	+	1.00E+00	0	0	=	4.3205E−08	0	465	+
F66	7.8641E−07	465	0	–	7.8641E−07	0	465	+	7.8641E−07	465	0	–	7.8641E−07	465	0	–
F67	1.7333E−06	0	465	+	1.7333E−06	0	465	+	1.7333E−06	0	465	+	1.7333E−06	0	465	+
F68	6.9824E−07	0	465	+	6.9824E−07	0	465	+	6.9824E−07	0	465	+	6.9824E−07	0	465	+
F69	1.1881E−06	0	465	+	1.1881E−06	0	465	+	1.1881E−06	0	465	+	1.1881E−06	0	465	+
F70	1.2001E−06	0	465	+	1.2001E−06	0	465	+	1.2001E−06	0	465	+	1.2001E−06	0	465	+
F71	9.2745E−07	0	465	+	9.2745E−07	0	465	+	9.2745E−07	0	465	+	9.2745E−07	0	465	+
F72	1.9721E−07	0	465	+	4.3205E−08	465	0	–	1.9721E−07	0	465	+	1.9721E−07	0	465	+
F73	8.8940E−07	0	465	+	8.8940E−07	0	465	+	8.8940E−07	0	465	+	8.8940E−07	0	465	+
F74	1.7333E−06	0	465	+	1.7333E−06	0	465	+	0.0752	319	146	=	1.7333E−06	0	465	+
F75	1.7344E−06	0	465	+	1.7344E−06	0	465	+	1.7344E−06	0	465	+	1.7344E−06	0	465	+
+/=/−	18/0/7				17/0/8				15/2/8				17/0/8			

Problem	CLPSO versus IA				SADE versus IA				BSA versus IA			
	p value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner
F51	6.9066E−07	465	0	–	6.9066E−07	465	0	–	6.9066E−07	465	0	–
F52	1.1826E−06	0	465	+	1.1826E−06	465	0	–	1.1826E−06	465	0	–
F53	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	465	0	–
F54	1.7333E−06	0	465	+	1.7333E−06	465	0	–	1.7333E−06	465	0	–
F55	4.3205E−08	0	465	+	4.3205E−08	0	465	+	4.3205E−08	0	465	+

Table 7 continued

Problem	CLPSO versus IA				SADE versus IA				BSA versus IA			
	p value	T+	T−	Winner	*p* value	T+	T−	Winner	*p* value	T+	T−	Winner
F56	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F57	6.7988E−08	465	0	–	6.7988E−08	465	0	–	6.7988E−08	465	0	–
F58	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F59	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F60	1.1567E−06	0	465	+	1.1567E−06	0	465	+	1.1567E−06	0	465	+
F61	4.3205E−08	0	465	+	4.3205E−08	465	0	–	4.3205E−08	0	465	+
F62	1.4403E−07	465	0	–	9.9562E−04	87	378	+	1.4403E−07	465	0	–
F63	4.3205E−08	465	0	–	4.3205E−08	465	0	–	4.3205E−08	465	0	–
F64	1.5117E−06	465	0	–	1.5117E−06	465	0	–	1.5117E−06	465	0	–
F65	4.3205E−08	0	465	+	4.3205E−08	0	465	+	1.00E+00	0	0	=
F66	7.8641E−07	465	0	–	7.8641E−07	465	0	–	7.8641E−07	465	0	–
F67	1.7333E−06	0	465	+	1.7333E−06	0	465	+	1.7333E−06	0	465	+
F68	6.9824E−07	0	465	+	6.9824E−07	0	465	+	6.9824E−07	0	465	+
F69	1.1881E−06	0	465	+	1.1881E−06	0	465	+	1.1881E−06	0	465	+
F70	1.2001E−06	0	465	+	1.2001E−06	0	465	+	1.2001E−06	0	465	+
F71	9.2745E−07	0	465	+	9.2745E−07	0	465	+	9.2745E−07	0	465	+
F72	1.9721E−07	0	465	+	1.9721E−07	0	465	+	1.9721E−07	0	465	+
F73	8.8940E−07	0	465	+	8.8940E−07	0	465	+	8.8940E−07	0	465	+
F74	1.7333E−06	0	465	+	1.7333E−06	0	465	+	0.0752	319	146	=
F75	1.7344E−06	0	465	+	1.7344E−06	0	465	+	1.7344E−06	0	465	+
+/=/−	17/0/8				14/0/11				11/2/12			

1: Initialization
2: **repeat**
3: Calculate fitness values of particles
4: Modify the best particles in the swarm
5: Choose the best particle
6: Calculate the particles' velocity
7: Update the particles' position
8: **until** convergence

Fig. 6 General structure of PSO

1: Initialization
2: **repeat**
3: Sample and validate offspring's fitness value
4: Sort the offspring by fitness
5: Perform environmental selection
6: Update the evolution path for covariance matrix adaptation
7: Update the covariance matrix
8: Update the step size
9: Update the mean
10: **until** convergence

Fig. 7 General structure of CMAES

The proposed IA is compared with BSA. The results indicate that IA works equally well as BSA in Test 1 and Test 2 as shown in Tables 6 and 7. However, Table 8 shows that BSA is better than IA. The unique mutation and crossover strategies of BSA make it a powerful minimization technique. However, results of Test 1 and Test 2 denote that IA has higher convergence rate as compared with BSA because of the relatively simpler structure of IA.

4 Conclusions and future directions

In this work, a novel socio-inspired algorithm referred to as ideology algorithm (IA), which is mainly inspired from the human society individuals following certain ideology, is proposed. Several operators were proposed and mathematically modelled for equipping the IA with high exploration and exploitation. The performance of the proposed algorithm was benchmarked on 75 test functions in terms of exploration, exploitation, local optima avoidance, fitness improvement of the population and convergence rate. It can be concluded that the proposed algorithm benefits from high exploitation and convergence rate.

The IA is compared to seven well-known and recent algorithms: PSO, CMAES, ABC, JDE, CLPSO, SADE and BSA. Wilcoxon statistical tests were also conducted when comparing the algorithms. The results showed that the proposed algorithm outperforms other algorithms in the majority of test functions. The statistical tests proved that the results were statistically significant for the IA. Thus, it may be concluded from the results that the proposed IA is comparable with other algorithms. Also, it is able to be applied as alternative optimizer for different optimization problems.

It is concluded that the IA improves the overall fitness of random initial solutions on optimization problems from the overall individuals' fitness. IA effectively searches and converges towards promising search space. Thus, the proposed algorithm is able to discover different regions of an optimization problem. Other remarks based on the results of this study are as follows:

- Initial random walks of individuals around the parties emphasize exploration of the search space around the individuals.
- Effective in local optima avoidance since IA employs a population of search agents to approximate the global optimum.

Table 8 Multi-problem-based statistical pairwise comparison of PSO, CMAES, ABC, JDE, CLPSO, SADE, BSA and proposed IA

Other algorithm versus IA	*p* Value	T+	T−	Winner
PSO versus IA	0.0038	55	270	IA
CMAES versus IA	0.0087	65	260	IA
ABC versus IA	0.0207	69	231	IA
JDE versus IA	0.0058	60	265	IA
CLPSO versus IA	0.0025	50	275	IA
SADE versus IA	0.0264	80	245	IA
BSA versus IA	0.2904	113	187	BSA

Fig. 8 General structure of ABC

1: Initialization
2: **repeat**
3: Place the employed bees on their food sources
4: Place the onlooker bees on the food sources depending on their nectar amounts
5: Send the scouts to the search area for discovering new food sources
6: Memorize the best food source found so far
7: **until** convergence

1: Initialization
2: Evaluation
3: **repeat**
4: Mutation
5: Recombination
6: Evaluation
7: Selection
8: **until** convergence

Fig. 9 General structure of DE

1: Initialization
2: **repeat**
3: Selection I
4: Mutation
5: Crossover
6: Selection II
7: **until** convergence

Fig. 10 General structure of BSA

- Promising search spaces are ensured since individuals relocate to the position of the best individuals during optimization.
- The best individual from each iteration is saved and considered as the elite, so all individuals tend towards the best solution obtained so far as well.
- IA has very few parameters to adjust. Thus, it is a flexible algorithm for solving diverse class of problems.
- The unique mechanism of IA where the local party leader competes with every other party leader and the second best individual in its own party. This motivates the party leaders to explore a greater and promising search space. Also, they continuously look for a better solution in its own local neighbourhood.
- Every individual in every party to directly and indirectly compete with the same party individuals as well as other party individuals. This makes every party to remain in competition and grow which motivates the individuals search for better solutions.

Several research directions can be recommended for future studies with the proposed algorithm. A multi-objective version of the IA could be developed to solve a wider class of problems. The algorithm could be applied for solving real-world problems from healthcare as well as supply-chain disruption domain [51, 52]. In addition, structural analysis [53–55] could be one of the promising areas where IA could be applied. Furthermore, constraint handling techniques [56] need to be developed to make the algorithm more generic and powerful. This may help IA solve real-world problems which are inherently constrained.

Acknowledgments The authors would like to thank Frontier Science Research Cluster, University Malaya Research Fund: RG333-15AFR, for supporting this work. The authors would also like to thank anonymous reviewers for comments and suggestions that have resulted in a much improved manuscript.

References

1. De Carvalho MG, Laender AHF, Goncalves MA et al (2012) A genetic programming approach to record deduplication. IEEE Trans Knowl Data Eng 24:399–412
2. Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evolut Comput 13:398–417
3. Tasgetiren MF, Suganthan PN, Pan QK (2010) An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem. Appl Math Comput 215:3356–3368
4. Kennedy J, Eberhart R (1995) Particle swarm optimization in neural networks. Neural Networks, 1995. In: Proceedings IEEE international conference 4, 1942–1948
5. Bonabeau E, Dorigo M, Theraulaz G (1999) Swarm intelligence: from natural to artificial systems. Oxford University Press, Oxford
6. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. Comput Intell Mag IEEE 1(4):28–39
7. Rahimi S, Roodposhti MS, Abbaspour RA (2014) Using combined AHP—genetic algorithm in artificial groundwater recharge site selection of Gareh Bygone Plain, Iran. Environ Earth Sci 72(6):1979–1992
8. Jordehi AR (2015) Chaotic bat swarm optimisation (CBSO). Appl Soft Comput 26:523–530
9. Abbaspour RA, Samadzadegan F (2011) Time-dependent personal tour planning and scheduling in metropolises. Expert Syst Appl 38(10):12439–12452
10. Yildiz AR (2013) Hybrid Taguchi-differential evolution algorithm for optimization of multi-pass turning operations. Appl Soft Comput 13:1433–1439
11. Yildiz AR (2013) Comparison of evolutionary-based optimization algorithms for structural design optimization. Eng Appl Artif Intell 26(1):327–333
12. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
13. Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. IEEE Trans Evolut Comput 13:945–958
14. Neri F, Tirronen V (2010) Recent advances in differential evolution: a survey and experimental analysis. Artif Intell Rev 33(1):61–106
15. Karaboga D, Akay B (2009) A comparative study of artificial bee colony algorithm. Appl Math Comput 214(1):108–132
16. Chen Y, Mazlack LJ, Minai AA, Lu LJ (2015) Inferring causal networks using fuzzy cognitive maps and evolutionary algorithms with application to gene regulatory network reconstruction. Appl Soft Comput 37:667–679
17. Rocha H, Peretta IS, Lima GFM, Marques LG, Yamanaka K (2015) Exterior lighting computer-automated design based on multi-criteria parallel evolutionary algorithm: optimized designs

for illumination quality and energy efficiency. Expert Syst Appl 45:208–222
18. Zhong F, Yuan B, Li B (2015) A hybrid evolutionary algorithm for multi-objective variation tolerant logic mapping on nanoscale crossbar architectures. Appl Soft Comput 38:955–966
19. Lei H, Wang R, Laporte G (2015) Solving a multi-objective dynamic stochastic districting and routing problem with a co-evolutionary algorithm. Comput Oper Res 67:12–24
20. Guo W, Zhang Y, Chen M, Wang L, Wu Q (2015) Fuzzy performance evaluation of Evolutionary Algorithms based on extreme learning classifier. Neurocomputing 175:371–382
21. Chica M, Bautista J, Cordón O, Damas S (2015) A multiobjective model and evolutionary algorithms for robust time and space assembly line balancing under uncertain demand. Omega 58:55–68
22. Pascual GG, Lopez-Herrejon RE, Pinto M, Fuentes L, Egyed A (2015) Applying multiobjective evolutionary algorithms to dynamic software product lines for reconfiguring mobile applications. J Syst Softw 103:392–411
23. Stępień J, Filipiak S (2014) Application of the evolutionary algorithm with memory at the population level for restoration service of electric power distribution networks. Int J Electr Power Energy Syst 63:695–704
24. Menai MEB (2014) Word sense disambiguation using evolutionary algorithms—Application to Arabic language. Comput Hum Behav 41:92–103
25. Marques I, Captivo ME (2015) Bicriteria elective surgery scheduling using an evolutionary algorithm. Oper Res Health Care 7:14–26
26. Ono S, Maeda H, Sakimoto K, Nakayama S (2014) User-system cooperative evolutionary computation for both quantitative and qualitative objective optimization in image processing filter design. Appl Soft Comput 15:203–218
27. Ayllón D, Gil-Pita R, Utrilla-Manso M, Rosa-Zurera M (2014) An evolutionary algorithm to optimize the microphone array configuration for speech acquisition in vehicles. Eng Appl Artif Intell 34:37–44
28. Clerc M, Kennedy J (2002) The particle swarm—explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evolut Comput 6:58–73
29. Blum C, Merkle D (2008) Swarm intelligence: Introduction and applications., Natural Computing SeriesSpringer, Berlin
30. Marini F, Walczak B (2015) Particle swarm optimization (PSO). A tutorial. Chemometr Intell Lab Syst 149:153–165
31. Yang XS, Deb S (2009) Cuckoo search via levy flights. World Congress on Nature and Biologically Inspired Computing (NaBIC 2009), Coimbatore, India, 4:210–214
32. Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evolut Comput 10:281–295
33. Omran MGH, Clerc M (2011) http://www.particleswarm.info/. Accessed 15 Nov 2015
34. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11:341–359
35. Qin AK, Suganthan PN (2005) Self-adaptive differential evolution algorithm for numerical optimization. IEEE Trans Evolut Comput 1(3):1785–1791
36. Igel C, Hansen N, Roth S (2007) Covariance matrix adaptation for multi-objective optimization. Evol Comput 15(1):1–28
37. Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evolut Comput 10:646–657
38. Civicioglu P (2013) Backtracking search optimization algorithm for numerical optimization problems. Appl Math Comput 219:8121–8144
39. CoelloCoello CA (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41:113–127
40. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evolut Comput 1:3–18
41. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim 39:459–471
42. Kumar R, Jyotishree (2012) Blending roulette wheel selection and rank selection in genetic algorithms. Int J Mach Learn Comput 2(4):365–370
43. Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98
44. Moradi MH, Abedini M (2012) A combination of genetic algorithm and particle swarm optimization for optimal DG location and sizing in distribution systems. Electr Power Energy Syst 34:66–74
45. Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report 1–50
46. Heidari AA, Abbaspour RA, Jordehi AR (2015) An efficient chaotic water cycle algorithm for optimization tasks. Neural Comput Appl. doi:10.1007/s00521-015-2037-2
47. Li X, Yao X (2012) Cooperatively coevolving particle swarms for large scale optimization. IEEE Trans Evolut Comput 16(2):210–224
48. Selvi V, Umarani R (2010) Comparative analysis of ant colony and particle swarm optimization techniques. Int J Comput Appl 5(4):975–8887
49. Rini DP, Shamsuddin SM, Yuhaniz SS (2011) Particle swarm optimization: technique, system and challenges. Int J Comput Appl 14(1):975–8887
50. Murugan R, Mohan MR (2012) Modified artificial bee colony algorithm for solving economic dispatch problem. ARPN J Eng Appl Sci 7(10):1353–1366
51. Kulkarni AJ, Baki MF, Chaouch BA (2016) Application of the cohort-intelligence optimization method to three selected combinatorial optimization problems. Eur J Oper Res 250(2):427–447
52. Kulkarni AJ, Shabir H (2014) Solving 0-1 Knapsack Problem using Cohort Intelligence Algorithm. Int J Mach Learn Cybernet. doi:10.1007/s13042-014-0272-y
53. Hasançebi O, Kazemzadeh Azad S (2015) Adaptive dimensional search: a new metaheuristic algorithm for discrete truss sizing optimization. Comput Struct 154:1–16
54. Kazemzadeh Azad S, Hasançebi O (2014) An elitist self-adaptive step-size search for structural design optimization. Appl Soft Comput 19:226–235
55. Kazemzadeh Azad S, Hasançebi O, Kazemzadeh Azad S (2013) Upper bound strategy for metaheuristic based design optimization of steel frames. Adv Eng Softw 57:19–32
56. Deshpande AM, Phatnani GM, Kulkarni AJ (2013) Constraint handling in firefly algorithm. In: Proceedings of IEEE international conference on cybernetics. Lausanne, Switzerland, 13–15 June 2013, pp 186–190