2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Macroscopic Indeterminacy Swarm Optimization (MISO) algorithm for real-parameter search

Po-Chun CHANG
Advanced Analytics Institute
University of Technology Sydney
PoChun.CHANG@student.uts.edu.au

Xiangjian He
Computer Vision and Recognition Laboratory
University of Technology, Sydney (UTS)
Xiangjian.He@uts.edu.au

***Abstract*—Swarm Intelligence (SI) is a nature-inspired emergent artificial intelligence. They are often inspired by the phenomena in nature. Many proposed algorithms are focused on designing new update mechanisms with formulae and equations to emerge new solutions. Despite the techniques used in an algorithm being the key factor of the whole system, the evaluation of candidate solutions also plays an important role. In this paper, the proposed algorithm Macroscopic Indeterminacy Swarm Optimization (MISO) presents a new search scheme with indeterminate moment of evaluation. Here, we perform an experiment based on public benchmark functions. The results produced by MISO, Differential Evolution (DE) with various settings, Artificial Bee Colony (ABC), Simplified Swarm Optimization (SSO), and Particle Swarm Optimization (PSO) have been compared. The result shows MISO can achieve similar or even better performance than other algorithms.**

Keywords—global optimization; evolution strategies; swarm intelligence; evolutionary algorithm; artifical intelligence

I. INTRODUCTION

Swarm Intelligence (SI) has become a robust method for dealing with global optimization issues. Swarm behavior is a collective motion of a set of self-propelled particles [1]. It is often inspired from nature phenomena, such as collective behavior, emergent behavior, self-organization, and etc. Particles propose solutions to problem and refine them by interacting with neighbor particles. Many algorithms such as Differential Evolution (DE)) [2], Particle Swarm Optimization (PSO) [3], and Simplified Swarm Optimization (SSO) [4-7] have been successfully employed in some fields. Most of SI systems proceed in principle according to the scheme illustrated in Fig. 1.

While a SI system discovers satisfactory solutions after a number of iterations, the whole progression is similar to animals' food finding behavior in nature. Individual swarm particles move toward to global optima through random walk biased by the predefined evaluation criteria and update mechanism. In macro scales, the movement of particles is continuous in solution space over iteration. Similar to the moving object which can be observed by naked eyes is in continuous space-time, which can be explained by classical mechanics. On the other hand, an object can only exist in one place in any one instant of time. In micro scale, the movement of particles consists of series of infinitesimal interval states. That is, the infinite numbers of discontinuous points in space summates to a continuous point set. Before the evaluation process taken place, the exact positions of particles are indeterminate due to new particle positions often resampled with stochastic techniques.

Similar to the uncertainty principle in quantum mechanics, the precise positions of point set cannot be established without observation even if the predicted law of motion is known [8].

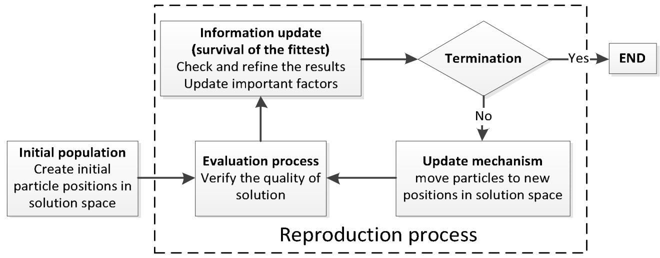


Fig. 1. The basic cycle of SI system

In our previous work, Simplified Swarm Optimization with Differential Evolution mutation strategy (SSODE) [9] introduced an update mechanism with multiple mutation formulae for synthesis new solution candidates. This paper extends previous research in SSODE and proposes a new algorithm: Macroscopic Indeterminacy Swarm Optimization (MISO). But before giving the explanation about MISO, the following paragraph summarizes the common iteratively refining principle used in many real world applications of meta-heuristic optimization.

By reviewing many algorithms (e.g. SSO[4-7], DE[2], and PSO[3]), every time the update mechanism is performed is immediately followed by evaluation process. In other words, the update mechanism is performed on a particle once per each instant of time (iteration). The whole optimization process is particles agents reach the region of local or global optima via stochastic but determinate trajectories (iteration-ordered sets of positions in solution space).

In this study, MISO presents a new concept Macroscopic Indeterminacy, which is inspired from the biased random walk in nature life. While observing certain animals such as birds, fishes, and insects, their swarming behaviors are coherent and synchronized with no leader in a group. Individual swarm members are constantly shifting their positions in response to the stimuli from 1) their colleagues or 2) the external environment (e.g. abstraction of food source). The whole optimization process is biased by the concentration gradient of food source. During the time interval that animals are randomly moving around in space, they frequently evaluate how close to the food source. Since the exact time step for applying evaluation process is uncertain, the time interval every time swarm members perform random walk is indeterminate.

This paper presents a new SI based algorithm called, Macroscopic Indeterminacy Swarm Optimization (MISO). The idea is, in every iteration, the update mechanism is executed multiple random times before evaluation process. The paper is organized as follows: Section II presents the literature review on the algorithms which bring new inspiration. Section III presents the proposed algorithm MISO. Section IV presents the experimental results while Section V concludes the paper.

II. REVIEWS FOR OPTIMIZATION ALGORITHMS

In many SI based algorithms, the initial population X is randomly generated and evaluated by objective function as well as fitness function. The system iteratively generates new solutions by reproduction process. The generated trial solution is accepted, if and only if it achieves the better evaluation result. TABLE I. shows the common steps for SI system [2, 10]:

TABLE I. SI SYSTEM COMMON STEPS

1. Initialize Population *X* with *NP* particles
2. Initial Evaluation: *eval(X)*
3. Update *the best solutions*
4. Repeat until *terminal criteria met*
   a. Reproduction process
      - Emerge trial population *U*
      - Evaluation: *eval(U)*
      - Re-constitute population *X*
   b. Update *the best solutions*
5. END

Differential Evolution (DE) uses the vector difference equation for generating new solutions, which is inspired by the mutation (TABLE II.) and crossover principles (1) in nature. There are many mutation schemes for emerge new solutions, TABLE II. lists five of them [11] [12]. The indices ${r_1}^i{}_{,G} \sim {r_5}^i{}_{,G}$ are mutually exclusive integers randomly chosen from $[1, NP]$, which are different form index i.

TABLE II. DE MUTATION SCHEMES

DE/Rand/1: $V_{i,G} = X_{r3^i,G} + F * \left(X_{r1^i,G} - X_{r2^i,G}\right)$
DE/Rand/2: $V_{i,G} = X_{r5^i,G} + F * \left(X_{r1^i,G} + X_{r2^i,G} - X_{r3^i,G} - X_{r4^i,G}\right)$
DE/best/1: $V_{i,G} = gb + F * \left(X_{r1^i,G} - X_{r2^i,G}\right)$
DE/best/2: $V_{i,G} = gb + F * \left(X_{r1^i,G} + X_{r2^i,G} - X_{r3^i,G} - X_{r4^i,G}\right)$
DE/rand-to-best/2: $V_{i,G} = X_{i,G} + F * \left(gb - X_{i,G} + X_{r1^i,G} - X_{r2^i,G}\right)$

Evaluation plays an important role in DE. The current set of solutions X is re-constituted as seen in equation (2). The fittest solutions discovered so far and will be used for the reproduction process in next generation. The better solutions will remain and update the knowledge that algorithm has learnt

Crossover:

$$U_{i,j,G} = \begin{cases} V_{i,j,G} & if\ \left(rand_{i,j}[0,1] \le CR\right) OR\ (j = j_{rand}) \\ X_{i,j,G} & otherwise \end{cases} \quad (1)$$

Re-constitute population *X*:

$$X_{i,G+1} = \begin{cases} U_i & if\ eval(U_i) > f\left(X_{i,G}\right) \\ X_{i,G} & otherwise \end{cases} \quad (2)$$

Simplified Swarm Optimization (SSO) [4-7] algorithm presents a special synthesis algorithm for recombination of vectors (TABLE III.).

TABLE III. SSO ALGORITHM

Control parameters: c_w, c_p, c_g, c_r

$$c_w + c_p + c_g + c_r = 1$$

$$C_w = c_w, C_p = C_w + c_p, C_g = C_p + c_g$$

$$U_{i,j,G} = \begin{cases} X_{i,j,G} & if\ \ 0 \le rand \le C_w \\ pbest_{i,j} & if\ \ C_w \le rand \le C_p \\ gbest_j & if\ \ C_p \le rand \le C_g \\ rand & if\ \ C_g \le rand \le 1 \end{cases}$$

SSODE [9] developed a new update mechanism from SSO algorithm structure with four mutation strategies which inspired from DE. SSODE pseudo code and mutation formulae are shown in TABLE IV. and TABLE V. .

TABLE IV. SSODE ALGORITHM

Control parameters: gm, cp, gr, cr

$gm + cp + gr + cr = 1$

$M_{gm} = gm,\ M_{cp} = M_{gm} + cp,\ M_{gr} = M_{cp} + gr$

$X^G = \{x_{i,j}^G \dots, x_{NP,D}^G\}$. for G^{th} generation

<u>LOOP Until termination criteria met</u>

$X^g = sort(X^G, 'desc')$

$$v_i^g = \begin{cases} F1\left(X^G, M_{gm}\right), & 1 \le i < round\left(NP * M_{gm}\right) \\ F2\left(X^G, M_{cp}\right), & round\left(NP * M_{gm}\right) \le i < round\left(NP * M_{cp}\right) \\ F3\left(X^G, M_{gr}\right), & round\ \left(NP * M_{cp}\right) \le i < round\left(NP * M_{gr}\right) \\ F4(X^G, rand), & round\left(NP * M_{gr}\right) \le i \le NP \end{cases}$$

$$x_{i,j}^{G+1} = \begin{cases} v_{i,j}^G, & \left(rand_j(0,1) \le CR\right) \\ x_{i,j}^G, & otherwise \end{cases}, \quad j = 1, \dots D$$

 # The goal is to minimize the evaluation result $eval(x)$.

$$pb_i = \begin{cases} x_i, & eval(x_i) < pb_i \\ pb_i, & else \end{cases}$$

$$gb = \begin{cases} pb_i, & eval(pb_i) < gb \\ gb, & else \end{cases}$$

END-LOOP

The best solutions discovered by individual particles are stored in particle best solutions $pb = \{pb_i\}, i = 1, \dots, NP$. The global best solution gb is the best solution in pb.

TABLE V. SSODE FOUR FORMULAE FOR MUTATION

Formula 1: $\boldsymbol{F1(x, F)}$ $v_i^G = x_i^G + F * (x_i^G - x_{r1}^G) + F * (x_{r2}^G - x_{r3}^G)$
Formula 2: $\boldsymbol{F2(x, F)}$ $v_i^G = x_i^G + F * (pb_{r1} - pb_{r2}) + F * (pb_{r3} - pb_{r4})$
Formula 3: $\boldsymbol{F3(x, F)}$ $v_i^g = gb + F * \left(x_{r1}^g - x_{r2}^g\right) + F * \left(x_{r3}^g - x_{r4}^g\right)$
Formula 4: $\boldsymbol{F4(x, F}$ $v_i^g = x_i^g + K * \left(x_{r1}^g - x_i^g\right) + F * \left(x_{r2}^g - x_{r3}^g\right)$

III. PROPOSED ALGORITHM MISO (MACROSCOPIC INDETERMINACY SWARM OPTIMIZATION)

MISO proposes that a particle's movement in a solution space has both continuous and discontinuous properties. Individual particles have their own best trajectories and continuously move from initial states to final positions. At the same time, every trajectory is formed by infinite points in time with discontinuous positions. There are operating rules with stochastic process to control particles' moving behavior in a solution space. Therefore, in every instant in time, the exact position is indeterminate without evaluation taken place.

Many algorithms (i.e. DE, PSO, ABC, etc.) evaluate a particle every time it shifts one position in space. Thus, a trial particle is generated with only one hop away from current one. Its positions are evaluated pre and post application, the indeterminacy of its movement is eliminated. Hypothetically, a function may be applied to particles to achieve better positions with continuous repetitions. The problem still remains as to the optimal point in time for the evaluation to take place. MISO has the advantage of resolving such problem. Algorithm efficiency and effectiveness are depended on the regulatory coordination between update mechanism and evaluation process. As a result, MISO proposes a new update mechanism (for synthesis new solution candidates) with uncertain time-interval for performing evaluation.

A. MISO algorithm

MISO is developed from common SI system framework. The new approach proposed by MISO is about the reproduction process (4.a step in TABLE I.). TABLE II. shows the pseudo-code for MISO.

1) Crucial vectors for particle interaction

The behavior of SI system is an emergent behavior from particles. Particles in computer are represented as vectors which store important knowledge. There are two crucial sets of vectors $\boldsymbol{iterBest}$ and $\boldsymbol{se}$ used in MISO algorithm.

$\boldsymbol{UpdateIterBest(gb)}$:

$$\begin{aligned}&if\ (\ eval(gb) > eval(iterBest_{iBIdx}) * 0.95)\\&\qquad iBIdx = iBIdx + 1;\\&\qquad iterBest_{iBIdx} = gb\\&endif\end{aligned} \tag{3}$$

$\boldsymbol{iterBest}$ is a set of vectors which only stores the significantly different $\boldsymbol{gb}$*s* (in ascending order) in all previous iterations. $\boldsymbol{iBIdx}$ is the last element as well as the latest gb. The significant difference is measured by the variance interval 0.05. gb is updated every time a better solution is discovered. Thus, $iterBest$ only records the gb which significantly better than the last element in $iterBest$ (3). This prevents $iterBest$ from being contaminated by the latest generations' results.

$\boldsymbol{UpdateSeeds(X, iterBest)}$:

$$\begin{aligned}&\boldsymbol{if(iBIdx < A_{size})}\\&\qquad se_{1\sim iBIdx} = iterBest_{iBIdx\sim 1}\\&\qquad se_{iBIdx+1\sim NP} = X_{iBIdx+1\sim NP}\\&\boldsymbol{else}\\&\qquad se_{1\sim A_{size}} = iterBest_{A_{size}\sim 1}\\&\qquad se_{A_{size}+1\sim NP} = X_{A_{size}+1\sim NP}\\&\boldsymbol{endif}\end{aligned} \tag{4}$$

After many iteration times, particles would be converged to local or global optima. The diversities of particles with their neighbors are decreased and less likely to produce significantly different particles. $\boldsymbol{se}$ is a set of vectors which stores the important knowledge learnt from previous generations. In order to maintain a certain level of diversity, a certain number of vectors are taken from $iterBest$ in descending order and are added into $\boldsymbol{se}$ (4).

2) PUO (Position Update Operator)

MISO presents a new update mechanism for effectively producing a new set of trial vectors $\boldsymbol{V}$. In SI system, the population of particles can be implicitly categorized into four classes [9]. Inspired by SSO algorithm which introduces three different thresholds for four different update strategies, MISO extends this concept into population level. Therefore, three thresholds separate the population into four groups. Unlike SSO, the four ratios na ,nb, nc, and nd control the number of particles in individual four classes: Class A, Class B, Class C, and Class D (TABLE VI.). Particles are randomly reordered before being regrouped into 4 groups.

TABLE VI. POSITION UPDATE OPERATOR $\boldsymbol{PUO(X, se)}$

Control parameters: na, nb, nc, nd
$na + nb + nc + nd = 1$ $M_{\text{na}} = \text{na}, M_{nb} = M_{\text{na}} + nb\ , M_{\text{nc}} = M_{\text{nb}} + nc$ $A_{end} = round(M_{\text{na}} * NP)$ $B_{end} = round(M_{nb} * NP)$ $C_{end} = round(M_{\text{nc}} * NP)$ $D_{end} = NP$
 # Class A $F = M_{na} + rand(0,1) * (nb)$ $for\ p = 1\sim A_{end}\ \{$ $\boldsymbol{V_p = X_{r1^p,G} + F * (X_{r2^p,G} - se_{r1^p,G})}$ $\}$ ***# Class B*** $F = M_{na} + rand(0,1) * (F + nc)$ $for\ p = (A_{end} + 1) \sim B_{end}\ \{$ $\boldsymbol{V_p = X_{r1^p,G} + F * (se_{p,G} - X_{r2^p,G})}$ $\}$ ***# Class C*** $F = M_{nb} + rand(0,1) * (F + nc)$ $for\ p = (B_{end} + 1) \sim C_{end}\ \{$ $V_p = X_{r1^p,G} + F * (se_{p,G} - X_{r2^p,G})$ $\}$ ***# Class D*** $F = M_{nc} + rand * (F + (1 - M_{nc}))$ $for\ p = (C_{end} + 1) \sim NP\ \{$ $\boldsymbol{V_p = gb + F * (se_{p,G} - X_{r1^p,G})}$ $\}$ *return V*
* $r_{1^p}$and $r_{2^p}$are non-repeated random sequence where $r_{1^p} \neq r_{2^p} \neq p$. * Differences between class B and class C are $\boldsymbol{F}$ and $\boldsymbol{p}$

The scaling rate $\boldsymbol{F}$ amplifies the differential variation between vectors which is dynamically generated according to different equations for different classes. The recombination mechanism shown in TABLE VII. is similar to crossover function in Evolutionary Algorithm (EA), which limits the migration only to randomly occur in some vector elements.

3) Uncertain time-interval for performing evaluation

Swarm optimization process can be illustrated as the movement of a set of particles from initial random positions toward to global optimal positions. Evaluation and information update is similar to human factor involved in verifying particles' positions. Each iteration can be considered as a time interval. Every time a set of particles changing positions can be considered as one instant state.

The uncertain time-interval means the repetition time of executing position update mechanism is indeterminate in every iteration.

TABLE VII. POSITION . MISO PSEUDO CODE (BASED ON THE FRAMEWORK IN TABLE 1)

1. Initialize Population *X* with *NP* particles
2. Initial Evaluation: *eval(X)*
3. Update $\boldsymbol{gb} = arg\,min_{i=1,\ldots,NP}\{eval(X_i)\}$
4. **Loop (until terminate criteria met)**

    $iterBest = \boldsymbol{UpdateIterBest}(gbest)$

    $se = \boldsymbol{UpdateSeeds}(X, iterBest)$

    $re = rand(1, D) \in \mathbb{N}_1$

    $for\ i = 1{\sim}re$ {

    V=$\boldsymbol{PUO}(\boldsymbol{X}, \boldsymbol{se})$

    $for\ j = 1{\sim}D$ {

    #Recombination

    $$U_{i,j} = \begin{cases} V_{i,j} & if\ (rand_{i,j} \le 0.5) \\ X_{i,j}^{G} & otherwise \end{cases}$$

    }

    }

    #Evaluation and update

    $$X_i^{G+1} = \begin{cases} U_i & if\ eval(U_i) < f(X_{i,G}) \\ X_{i,G} & otherwise \end{cases}, \quad i = 1, \ldots, NP$$

    $$gb = \begin{cases} U_i, & eval(U_i) < gb \\ gb, & else \end{cases}$$

5. **END**

In TABLE VII. , $\boldsymbol{re}$ is randomly generated in every iteration. It simulates the uncertain number of instants of time between current and previous evaluations performed on individual particles. $\boldsymbol{PUO}$ repeatedly updates the trial vectors $V = \{V_i\}, i = 1, \ldots, NP$ for re times. The recombination process is also performed $\boldsymbol{re}$ times for synthesizing updated solution candidates U. That is, every particle X_i performs re times hops to arrive position U_i in a solution space. The maximum number of interval states between two evaluations is equal to the dimension of search space D. It is because excessive updates for particles positions are not only time consuming but also renders the output results virtually random.

IV. EXPERIMENTS AND RESULTS

The purpose of this experiment is to evaluate the performance of proposed algorithm MISO. The default MISO algorithm with no control parameter setting is compared with DE [2, 14], ABC[15] , SSO [5-7] and PSO[16]. Here, DE/best/1 and DE/rand/1 mutation strategies are selected. Three recommended setting for DE parameters are chosen [2, 13]. The benchmark ABC source code is downloaded from ABC algorithm homepage [17]. SSO control parameter is set Cw ' ' '=[0.1, 0.3, 0.5,0.1], which is the same setting as Yeh et al. experiment [4]. The setting for PSO is $c_1 = c_2 = 2.0$ with decaying weight $w = 0.9{\sim}0.4$, which is able to perform global search at the beginning and local search at the end [18, 19]. The population size $NP = 10 \times D$ for all algorithms, except ABC $NP = 40$ [17]. Here, the experiment is performed in 30 dimensions $D = 30$. The parameter settings for all compared algorithms are summarized in TABLE VIII. .

TABLE VIII. PARAMETER SETTINGS FOR ALL COMPARED ALGORITHMS

			D	NP
MISO	-	-	30	300
DE/Best/1	DE_1_0503	F=0.5,CR=0.3	30	300
	DE_1_0901	F=0.9,CR=0.1	30	300
	DE_1_0909	F=0.9,CR=0.9	30	300
DE/Rand/1	DE_2_0503	F=0.5,CR=0.3	30	300
	DE_2_0901	F=0.9,CR=0.1	30	300
	DE_2_0909	F=0.9,CR=0.9	30	300
ABC	-	-	30	40
SSO(1,3,5,1)	cw=0.1, cp=0.3,cg=0.5,cr=0.1		30	300
PSO	c1=2, c2=2, w=0.9~04		30	300

The boundary handling method called Periodic mode [20] is applied to solve beyond boundary problem (5).

$$range = ub - lb$$
$$u_{i,j} = \begin{cases} ub - mod\left((lb - u_{i,j}), range\right) IF\ u_{i,j} < lb \\ lb + mod\left((u_{i,j} - ub), range\right) IF\ u_{i,j} > ub \\ u_{i,j} \quad IF\ u_{i,j} \in [lb, ub] \end{cases} \quad (5)$$
$$*\ mod = modulus$$

All algorithms are implemented in C language. The computer specification is: Intel(R) Core(TM) i5-2400 CPU @ 3.10GHz with 4GB memory. The operating system is 64-bit windows 7. The experiment is carried on 28 benchmark functions (see TABLE IX. & TABLE X.) from CEC 2013 Special Session on Real-Parameter Optimization [19].

TABLE IX. BENCHMARK FUNCTIONS SETTINGS

No.	Function name	fbias
Bm1	Sphere Function	-1400
Bm2	Rotated High Conditioned Elliptic Function	-1300
Bm3	Rotated Bent Cigar Function	-1200
Bm4	Rotated Discus Function	-1100
Bm5	Different Powers Function	-1000
Bm6	Rotated Rosenbrock's Function	-900
Bm7	Rotated Schaffers F7 Function	-800
Bm8	Roated Ackley's Function	-700
Bm9	Rotated Weierstrass Function	-600
Bm10	Rotated Griewank's Function	-500
Bm11	Rastrigin's Function	-400
Bm12	Rotated Rastrigin's Function	-300
Bm13	Non-Continuous Rotated Rastrigin's Function	-200
Bm14	Schwefel's Function	-100
Bm15	Rotated Schwefel's Function	100
Bm16	Rotated Katsuura Function	200
Bm17	Lunacek Bi_Rastrigin Function	300
Bm18	Rotated Lunacek Bi_Rastrigin Function	400
Bm19	Expanded Griewank's plus Rosenbrock's Function	500
Bm20	Expanded Scaffer's F6 Function	600

TABLE X. COMPOSITION BENCHMARK FUNCTIONS

No.	Function name	elements	fbias
Bm21	Composition Function 1 (n=5,Rotated)	Bm1,Bm3,Bm4,Bm5,Bm6	700
Bm22	Composition Function 2 (n=3,Unrotated)	Bm14	800
Bm23	Composition Function 3 (n=3,Rotated)	Bm15	900
Bm24	Composition Function 4 (n=3,Rotated)	Bm9,Bm12,Bm15	1000
Bm25	Composition Function 5 (n=3,Rotated)	Bm9,Bm12,Bm15	1100
Bm26	Composition Function 6 (n=5,Rotated)	Bm2,Bm9,Bm10,Bm12,Bm15	1200
Bm27	Composition Function 7 (n=5,Rotated)	Bm1,Bm9,Bm10,Bm12,Bm15	1300
Bm28	Composition Function 8 (n=5,Rotated)	Bm1,Bm7,Bm15,Bm19,Bm20	1400

Two measure criteria are examined: 1) performance within limited Function Evaluation times, 2) algorithm complexity and 3) overall run-time performance. The main consideration for optimization algorithm is the quality of discovered solutions. Assume the objective function $F(x)$, the optimum discovered by algorithm x^*, and the predefined global optimum o, and then the evaluation is based on the error value $err = eval(x) = |F(x^*) - F(o)|$.

A. Experiment based on the solution quality within given Function Evaluation times (FEs)

The maximum number of function evaluation $MaxFEs = 300{,}000$ is set for all algorithms. The statistical analysis is based on minimum, maximum and mean of error values, and the standard deviation of 30 applications of experiment. According to the No Free Lunch Theorem [15], there is no algorithm can outperform all others in all problems. Therefore, a generalized algorithm should be the one can achieve satisfactory results in many of the given problems. Overall 28 benchmark functions, MISO can achieve better performance and generalization ability than DE, ABC, SSO, and PSO algorithms in 14 benchmark functions (TABLE XI.).

TABLE XI. SUMMARY OF ALGORITHMS PERFORMANCES OVER 28 BENCHMARK FUNCTIONS

	MISO	DE(counts of all strategies)	ABC	SSO	PSO	Total
max	13	4	11	0	0	28
min	13	3	8	3	1	28
mean	14	3	10	1	0	28
std	8	12	8	0	0	28

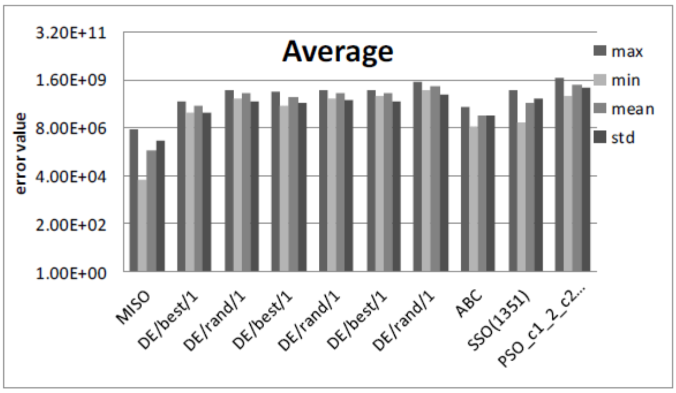


Fig. 2. The average of the evaluation results over all 28 functions by MISO, DE, ABC, SSO, & PSO.

Fig. 2 shows the average of the evaluation results over 28 benchmark functions. It shows the overall performance for MISO is significantly better than others. This means even though MISO cannot achieve better results than other algorithms in some benchmark problems, it can still discover acceptable solutions in many of them. TABLE XIII. lists some benchmark functions that MISO outperforms other algorithms

B. Experiment based on algorithm complexity

The Performance measurement based on the solution quality within $MaxFEs$ is not always a fair approach way for testing algorithm. One of the key issues is algorithmic complexity, which commonly refers to the amount of time for an algorithm to run. For solving the same problems within $MaxFEs$, some algorithms finish earlier than others. Based on the evaluation criteria for algorithm complexity in CEC 2013 Special Session [19], the results is shown in TABLE XII. . The experiment result shows MISO has the highest algorithm complexity, which means MISO takes longer time to run. It is reasonable because every iteration MISO executes the update mechanism multiple times before an evaluation takes place.

TABLE XII. SUMMARY OF ALGORITHM COMPLEXITY OVER 28 BENCHMARK FUNCTIONS

Time (sec)	Miso	F=0.5 CR=0.3		F=0.9 CR=0.1		F=0.9 CR=0.9		ABC	SSO	PSO
		DE/best/1	DE/rand/1	DE/best/1	DE/rand/1	DE/best/1	DE/rand/1			
mean	1.48E+01	3.46E+00	3.31E+00	3.22E+00	4.10E+00	3.79E+00	3.35E+00	3.35E+00	1.03E+01	8.51E+00
sum	4.13E+02	9.69E+01	9.27E+01	9.03E+01	1.15E+02	1.06E+02	9.37E+01	9.38E+01	2.87E+02	2.38E+02

C. Experiment based on the run-time performance

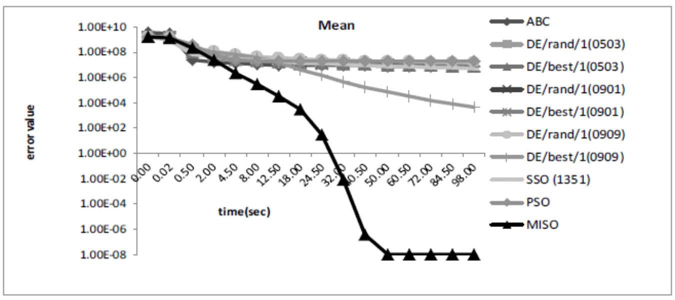
However, efficiency does not always mean to be effective. We perform additional experiments for verifying the run-time performance. The function error value is recorded after $T = \{t_i\}$, $t_i = 0.5 * (i)^2, i = 0 \sim 14$ in seconds. That is, $\{ 0, 0.5, 2, 4.5, 8, 12.5, 18, 24.5, 32, 40.5, 50, 60.5, 72, 84.5, 98 \}$ seconds. The statistical analysis is based on of error values of 30 applications of experiment. Here, we display the convergence graphs for some descriptive benchmark functions (Fig. 3~ Fig. 14). In the graphs, if error value ≤ 1.00E-08, than set it to 1.00E-08.

The graphs indicate that many algorithms like SSO, DE, ABC, and PSO are able to form fast convergence at the beginning. However, they may have premature optimization issue (see Fig. 3, 4, 5, 7, 6, &10) and suffer from local optima. Besides, as it can be seen from Fig. 8, 9, 11, 12, 13, & 14, after algorithms running a period of time, the performance of MISO is gradually improving. Finally, MISO achieves either fast convergence, or discovers the better optimal solutions than others.

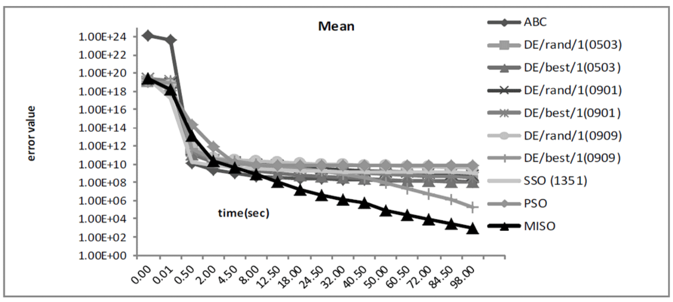
After all, according to these three experiments, we can conclude that an algorithm with high time complexity can be complemented by small number of function evaluation times. Despite of the fact that every iteration MISO takes longer time to execute, it can achieve satisfactory result within less number of function evaluations. Consuming lots of time in every iteration would pay off in the end. paper. Do not number text heads-the template will do that for you.

TABLE XIII. SUMMARY RESULTS OBTAINED BY MISO, DE, ABC, SSO, & PSO (MAXFES=300,000)

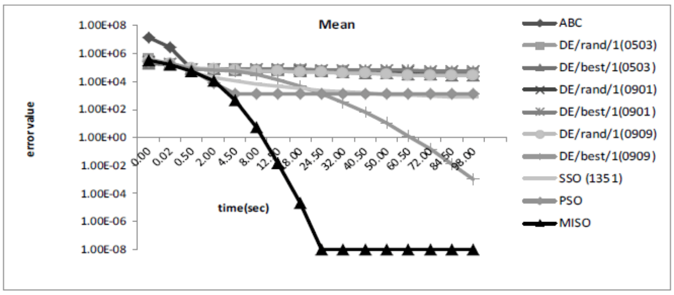
		MISO	F=0.5 CR=0.3 DE/best/1	F=0.5 CR=0.3 DE/rand/1	F=0.9 CR=0.1 DE/best/1	F=0.9 CR=0.1 DE/rand/1	F=0.9 CR=0.9 DE/best/1	F=0.9 CR=0.9 DE/rand/1	ABC	SSO (1351)	PSO
bm2	**max**	**3.15E+04**	**3.99E+07**	**4.38E+07**	**4.15E+07**	**4.02E+07**	**7.49E+07**	**8.50E+07**	**2.14E+07**	**5.88E+07**	**1.10E+08**
	mean	**1.82E+03**	2.54E+07	3.34E+07	3.01E+07	3.03E+07	5.49E+07	6.62E+07	1.50E+07	2.85E+07	5.02E+07
	min	**1.05E+01**	1.13E+07	2.01E+07	1.75E+07	1.95E+07	2.92E+07	4.49E+07	9.50E+06	5.42E+06	1.56E+07
	std	**4.51E+03**	6.24E+06	5.92E+06	6.47E+06	5.72E+06	1.12E+07	1.14E+07	3.14E+06	1.34E+07	2.14E+07
bm3	**max**	**1.92E+08**	4.22E+09	1.37E+10	1.20E+10	1.47E+10	1.45E+10	3.42E+10	2.18E+09	1.33E+10	5.49E+10
	mean	**1.84E+07**	2.42E+09	1.10E+10	6.81E+09	1.02E+10	1.08E+10	2.41E+10	8.42E+08	3.76E+09	2.51E+10
	min	**6.54E+05**	1.22E+09	5.58E+09	2.66E+09	5.41E+09	7.45E+09	1.35E+10	2.37E+08	4.04E+08	7.86E+09
	std	**3.23E+07**	6.78E+08	2.02E+09	2.11E+09	2.47E+09	2.01E+09	4.25E+09	5.22E+08	3.33E+09	1.08E+10
bm4	**max**	**2.95E-03**	8.84E+04	9.13E+04	1.01E+05	9.07E+04	7.42E+04	8.61E+04	8.92E+04	2.01E+04	4.03E+04
	mean	**6.69E-04**	7.17E+04	7.56E+04	7.97E+04	7.75E+04	5.86E+04	6.94E+04	7.12E+04	1.28E+04	1.28E+04
	min	**2.59E-05**	4.47E+04	5.90E+04	5.63E+04	4.60E+04	3.57E+04	4.88E+04	4.99E+04	8.05E+03	3.67E+03
	std	**7.66E-04**	9.06E+03	8.02E+03	1.03E+04	8.55E+03	8.78E+03	8.00E+03	8.22E+03	2.79E+03	7.50E+03
bm6	**max**	7.41E+01	8.15E+01	1.35E+02	5.83E+01	1.29E+02	9.93E+01	3.90E+02	1.88E+01	1.43E+02	1.46E+03
	mean	**1.15E+01**	3.00E+01	1.12E+02	3.80E+01	1.04E+02	7.45E+01	3.24E+02	1.51E+01	7.14E+01	6.13E+02
	min	**1.40E-03**	1.86E+01	7.30E+01	2.71E+01	7.81E+01	5.53E+01	2.27E+02	8.17E+00	1.58E+01	8.94E+01
	std	1.60E+01	1.25E+01	1.45E+01	7.18E+00	1.13E+01	9.73E+00	3.82E+01	**2.66E+00**	3.33E+01	2.42E+02
bm7	**max**	**3.98E+01**	1.12E+02	1.38E+02	1.39E+02	1.40E+02	1.24E+02	1.71E+02	1.47E+02	1.89E+02	3.16E+02
	mean	**1.40E+01**	9.07E+01	1.13E+02	1.11E+02	1.13E+02	1.08E+02	1.42E+02	1.19E+02	1.17E+02	1.40E+02
	min	**2.73E+00**	6.18E+01	8.53E+01	7.67E+01	7.44E+01	8.64E+01	1.15E+02	5.70E+01	4.81E+01	7.64E+01
	std	**8.62E+00**	1.06E+01	1.19E+01	1.26E+01	1.29E+01	1.03E+01	1.49E+01	1.79E+01	2.90E+01	4.55E+01
bm9	**max**	3.44E+01	3.21E+01	3.20E+01	3.20E+01	**3.18E+01**	3.49E+01	3.55E+01	3.31E+01	3.72E+01	4.11E+01
	mean	**2.79E+01**	2.97E+01	2.98E+01	2.93E+01	2.93E+01	3.30E+01	3.30E+01	2.97E+01	2.92E+01	3.43E+01
	min	**2.02E+01**	2.63E+01	2.72E+01	2.39E+01	2.39E+01	3.05E+01	2.97E+01	2.42E+01	2.25E+01	2.36E+01
	std	3.11E+00	1.35E+00	1.19E+00	1.72E+00	1.72E+00	**1.18E+00**	1.36E+00	1.83E+00	3.11E+00	4.23E+00
bm10	**max**	**1.27E-01**	5.33E+01	2.38E+02	9.75E+01	2.06E+02	2.83E+02	8.69E+02	8.09E-01	5.69E+01	1.37E+03
	mean	**1.54E-02**	3.03E+01	1.83E+02	7.18E+01	1.54E+02	2.03E+02	6.74E+02	3.86E-01	2.58E+01	7.49E+02
	min	**1.12E-04**	1.76E+01	1.20E+02	3.90E+01	9.82E+01	1.34E+02	4.91E+02	1.95E-01	7.73E+00	2.42E+02
	std	2.09E-02	7.27E+00	2.60E+01	1.35E+01	2.33E+01	3.00E+01	8.49E+01	1.22E-01	1.11E+01	2.43E+02
bm12	**max**	**1.72E+02**	1.84E+02	2.83E+02	2.49E+02	2.79E+02	2.50E+02	3.48E+02	3.20E+02	2.86E+02	3.57E+02
	mean	**1.19E+02**	1.53E+02	2.50E+02	2.10E+02	2.47E+02	2.27E+02	3.09E+02	2.63E+02	1.40E+02	2.35E+02
	min	6.22E+01	1.13E+02	2.08E+02	1.65E+02	2.01E+02	1.92E+02	2.65E+02	1.63E+02	**6.06E+01**	1.24E+02
	std	2.66E+01	1.79E+01	1.71E+01	2.03E+01	1.90E+01	**1.39E+01**	1.74E+01	3.43E+01	4.52E+01	5.44E+01
bm13	**max**	**1.93E+02**	2.22E+02	3.17E+02	2.87E+02	3.04E+02	2.69E+02	3.59E+02	3.73E+02	3.09E+02	4.20E+02
	mean	**1.62E+02**	1.85E+02	2.84E+02	2.41E+02	2.78E+02	2.36E+02	3.23E+02	3.12E+02	2.16E+02	3.02E+02
	min	**7.95E+01**	1.46E+02	2.45E+02	1.92E+02	2.32E+02	2.00E+02	2.78E+02	2.21E+02	1.10E+02	1.96E+02
	std	2.55E+01	1.63E+01	1.66E+01	1.83E+01	**1.50E+01**	1.55E+01	1.98E+01	3.18E+01	4.71E+01	4.91E+01
bm18	**max**	**2.27E+02**	2.58E+02	3.56E+02	3.13E+02	3.64E+02	2.93E+02	4.15E+02	3.89E+02	3.87E+02	3.49E+02
	mean	**2.06E+02**	2.33E+02	3.26E+02	2.84E+02	3.33E+02	2.70E+02	3.86E+02	3.40E+02	2.57E+02	2.20E+02
	min	1.71E+02	1.89E+02	2.85E+02	2.43E+02	2.90E+02	2.32E+02	3.25E+02	2.66E+02	1.58E+02	**1.41E+02**
	std	**1.07E+01**	1.53E+01	1.75E+01	1.69E+01	1.75E+01	1.28E+01	2.24E+01	2.82E+01	4.42E+01	5.01E+01
bm20	**max**	**1.25E+01**	1.48E+01	1.47E+01	1.50E+01	1.50E+01	1.43E+01	1.44E+01	1.50E+01	1.32E+01	1.48E+01
	mean	**1.19E+01**	1.41E+01	1.40E+01	1.43E+01	1.43E+01	1.39E+01	1.39E+01	1.45E+01	1.22E+01	1.35E+01
	min	**1.11E+01**	1.31E+01	1.29E+01	1.32E+01	1.29E+01	1.33E+01	1.27E+01	1.37E+01	1.11E+01	1.18E+01
	std	3.05E-01	3.64E-01	4.76E-01	3.65E-01	4.19E-01	**2.56E-01**	2.96E-01	2.80E-01	4.90E-01	8.74E-01
bm24	**max**	**2.47E+02**	2.87E+02	2.93E+02	2.94E+02	2.94E+02	2.97E+02	3.04E+02	2.97E+02	3.05E+02	3.33E+02
	mean	**2.26E+02**	2.77E+02	2.86E+02	2.83E+02	2.85E+02	2.87E+02	2.96E+02	2.87E+02	2.87E+02	3.04E+02
	min	**2.10E+02**	2.63E+02	2.72E+02	2.68E+02	2.62E+02	2.78E+02	2.83E+02	2.75E+02	2.59E+02	2.72E+02
	std	7.59E+00	5.60E+00	**4.55E+00**	5.75E+00	5.31E+00	4.69E+00	4.74E+00	4.96E+00	1.06E+01	1.40E+01
bm25	**max**	**2.81E+02**	3.07E+02	3.10E+02	3.11E+02	3.10E+02	3.17E+02	3.20E+02	3.15E+02	3.19E+02	3.63E+02
	mean	**2.34E+02**	2.98E+02	3.03E+02	3.02E+02	3.04E+02	3.10E+02	3.15E+02	3.07E+02	3.01E+02	3.45E+02
	min	**2.07E+02**	2.85E+02	2.88E+02	2.90E+02	2.94E+02	3.02E+02	3.09E+02	2.98E+02	2.77E+02	3.14E+02
	std	2.02E+01	4.46E+00	4.07E+00	4.25E+00	3.51E+00	3.55E+00	**2.81E+00**	4.30E+00	8.59E+00	1.05E+01
bm26	**max**	**2.00028E+02**	2.03E+02	2.04E+02	2.04E+02	2.04E+02	2.09E+02	2.10E+02	2.0144E+02	3.90E+02	2.04E+02
	mean	**2.00007E+02**	2.02E+02	2.03E+02	2.03E+02	2.03E+02	2.06E+02	2.06E+02	2.0084E+02	2.43E+02	2.01E+02
	min	**2.00001E+02**	2.01E+02	2.01E+02	2.01E+02	2.01E+02	2.03E+02	2.03E+02	2.0052E+02	2.01E+02	2.00E+02
	std	**6.75498E-03**	5.46E-01	6.15E-01	5.83E-01	6.69E-01	1.36E+00	1.50E+00	2.0228E-01	7.57E+01	9.95E-01



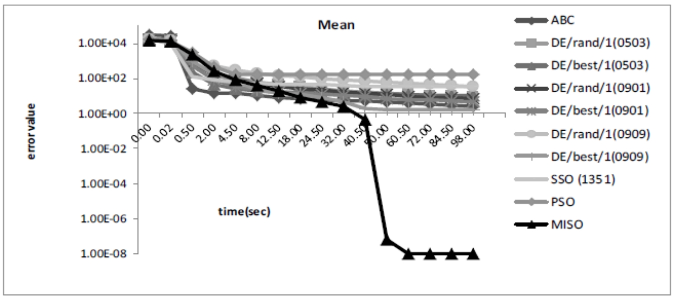
Bm2: Rotated High Conditioned Elliptic Function
Fig. 3. .



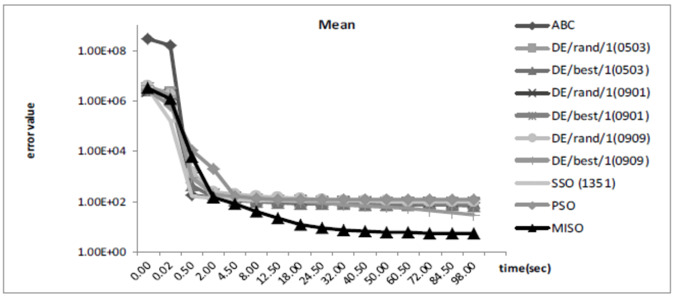
Bm3: Rotated Bent Cigar Function
Fig. 4. .



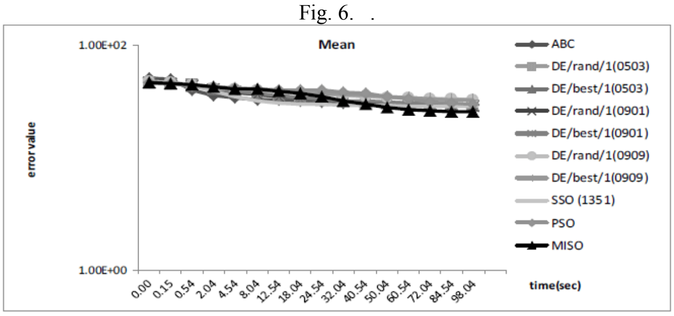
Bm4: Rotated Discus Function
Fig. 5. .



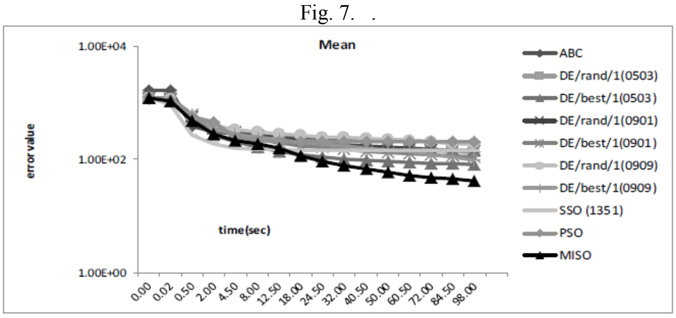
Bm6: Rotated Rosenbrock's Function
Fig. 6. .



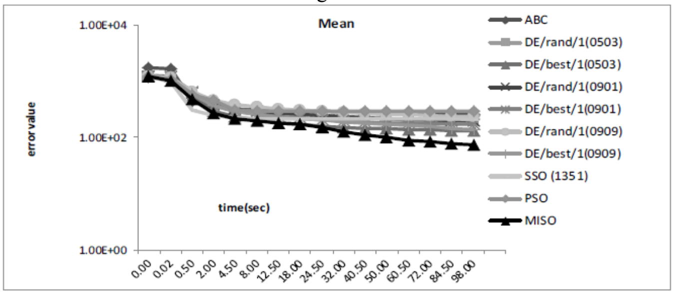
Bm7: Rotated Schaffers F7 Function
Fig. 7. .



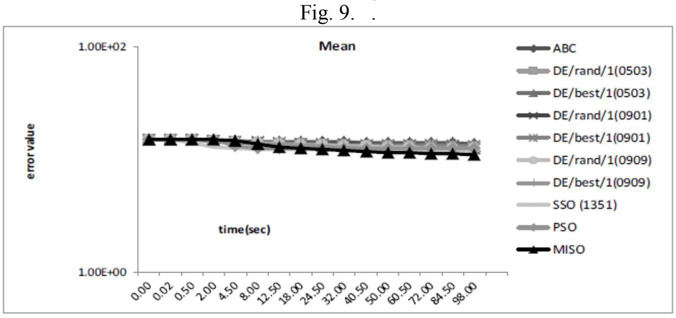
Bm9: Rotated Weierstrass Function
Fig. 8. .



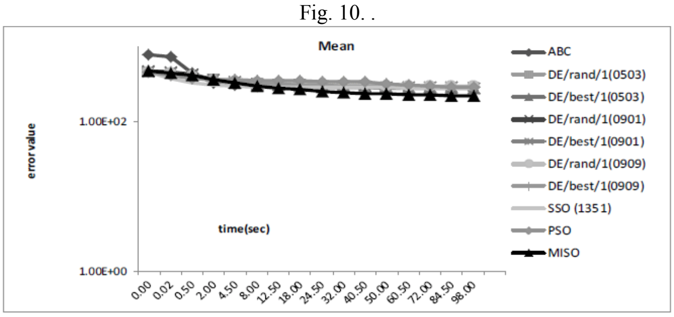
Bm12: Rotated Rastrigin's Function
Fig. 9. .



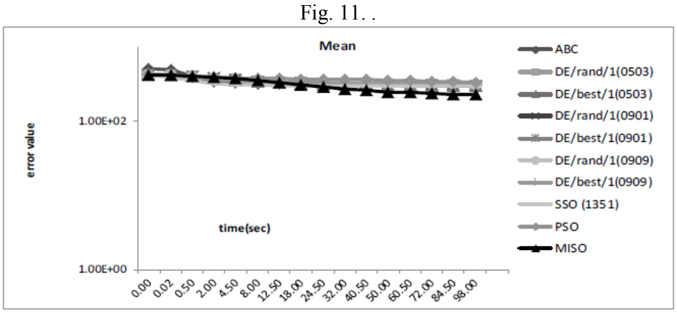
Bm13: Non-Continuous Rotated Rastrigin's Function
Fig. 10. .



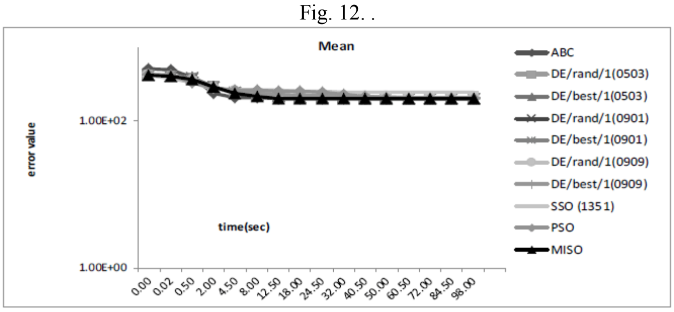
Bm20: Expanded Scaffer's F6 Function.
Fig. 11. .



Bm23: Composition Function 3
Fig. 12. .



Bm25: Composition Function 5.
Fig. 13. .



Bm26: Composition Function 6.
Fig. 14. .

V. Conclusion

The update mechanism, which is a strategy or set of steps and equations to produce new solutions, plays important role for discovering new solution candidates. Many representative algorithms such as SSO, DE, PSO, and ABC, have been proposed to solve global optimization problems. Their ideas were assured to solve the optimization problems based on their hypotheses. However, they do not guarantee success. Even more, they seem to be too ambitious, and try to synthesis better solution candidates in every iteration.

Macroscopic Indeterminacy Swarm Optimization (MISO) algorithm is proposed in this paper. In order to improve the performance of optimization algorithm, MISO considers well-designed update mechanism and suitable evaluation strategy need to be auxiliary to each other. Macroscopic Indeterminacy refers to biased random walk phenomena in nature. Lives find ways out to survive. Living organisms constantly change their positions in response to the stimuli from colleagues and their environment. Similarly, particle agents constantly change their positions in response to the stimuli from colleagues, and frequently response to conditions (solution quality) in solution space. That is, particles may update their positions multiple times before evaluation take place.

Based on benchmark functions provided by CEC conference [19], the experiments show MISO is superior SSO, DE, PSO, and ABC in many of them. In spite of the fact that algorithm complexity for MISO is significantly higher than others, it can achieve good results within small number of function evaluations. By given enough of processing time, MISO is able to come from behind and achieve efficient and effective results in many benchmark problems. The overall performance of MISO is significantly better than others

VI. Reference

[1] O. O'Loan, and M. Evans, "Alternating steady state in one-dimensional flocking," *Journal of Physics A: Mathematical and General,* vol. 32, no. 8, pp. L99, 1999.

[2] R. Storn, and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization,* vol. 11, no. 4, pp. 341-359, 1997.

[3] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm intelligence,* vol. 1, no. 1, pp. 33-57, 2007.

[4] W. C. Yeh, W. W. Chang, and Y. Y. Chung, "A new hybrid approach for mining breast cancer pattern using discrete particle swarm optimization and statistical method," *Expert Systems with Applications,* vol. 36, no. 4, pp. 8204-8211, 2009.

[5] W. C. Yeh, "A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems," *Expert Systems with Applications,* vol. 36, no. 5, pp. 9192-9200, 2009.

[6] Y. Wei-Chang, "Optimization of the Disassembly Sequencing Problem on the Basis of Self-Adaptive Simplified Swarm Optimization," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on,* vol. 42, no. 1, pp. 250-261, 2012.

[7] W.-C. Yeh, "New Parameter-Free Simplified Swarm Optimization for Artificial Neural Network Training and Its Application in the Prediction of Time Series," 2013.

[8] G. Shan, "From quantum motion to classical motion-seeking the lost reality," *Physics Essays,* vol. 14, no. 1, pp. 37-48, 2001.

[9] P.-C. Chang, and W.-C. Yeh, "Simplified swarm optimization with differential evolution mutation strategy for parameter search." p. 25.

[10] J. Kennedy, "Swarm Intelligence Handbook of Nature-Inspired and Innovative Computing," A. Zomaya, ed., pp. 187-219: Springer US, 2006.

[11] A. Qin, V. Huang, and P. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *Evolutionary Computation, IEEE Transactions on,* vol. 13, no. 2, pp. 398-417, 2009.

[12] R. Storn, "On the usage of differential evolution for function optimization." pp. 519-523.

[13] R. Mallipeddi, P. Suganthan, Q. Pan *et al.*, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Applied Soft Computing,* vol. 11, no. 2, pp. 1679-1696, 2011.

[14] K. V. Price, and R. M. Storn. "Differential Evolution Homepage," http://www1.icsi.berkeley.edu/~storn/code.html.

[15] D. Karaboga, and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *Journal of Global Optimization,* vol. 39, no. 3, pp. 459-471, 2007.

[16] J. Kennedy, and R. Eberhart, "Particle swarm optimization." pp. 1942-1948 vol.4.

[17] D. Karaboga, "Artificial bee colony (ABC) algorithm homepage," 2009.

[18] Y. Shi, and R. Eberhart, "Parameter selection in particle swarm optimization." pp. 591-600.

[19] Y. Shi, and R. C. Eberhart, "Empirical study of particle swarm optimization."

[20] W.-J. Zhang, X.-F. Xie, and D.-C. Bi, "Handling boundary constraints for numerical optimization by particle swarm flying in periodic search space." pp. 2307-2311.