Scientific Research and Essays Vol. 7(41), pp. 3566 -3583, 27 October, 2012
Available online at http://www.academicjournals.org/SRE
DOI: 10.5897/SRE12.514
ISSN 1992-2248 ©2012 Academic Journals

Full Length Research Paper

A novel meta-heuristic algorithm for numerical function optimization: Blind, naked mole-rats (BNMR) algorithm

Mohammad Taherdangkoo[1]* Mohammad Hossein Shirzadi[2] and Mohammad Hadi Bagheri[3]

[1]Department of Communications and Electronics, Faculty of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran.
[2]Department of Industrial Engineering, University of Tehran, Tehran, Iran.
[3]Center for Evidence-Based Imaging, Department of Radiology, Brigham and Women's Hospital, Harvard Medical School, Brookline, MA, USA.

Accepted 24 September, 2012

Optimization algorithms inspired by the world of nature have turned into powerful tools for solving the complicated problems. However, they have still some drawbacks need the investigation of new and better optimization algorithms. In this paper, we propose a new meta-heuristic algorithm called blind naked mole-rats (BNMR) algorithm. This algorithm has been developed based on the social behavior of the blind naked mole-rats colony in searching the food and protecting the colony against invasions. By introducing this algorithm, we have tried to overcome many disadvantages of the usual optimization algorithms including getting trapped in local minimums or having low rate of convergence. Using several benchmark functions, we demonstrate the superior performance of the proposed algorithm in comparison with some other well-known optimization algorithms.

Key words: Meta-heuristic algorithm, optimization algorithm, BNMR algorithm.

INTRODUCTION

The meta-heuristic algorithms have been mostly derived from the behavior of biological systems (for example, genetic algorithm (GA), particle swarm optimization (PSO), stem cells algorithm (SCA)) or physical systems (for example, simulated annealing (SA)). Perhaps the main reason for selecting and developing these algorithms is the simplicity in formulating and understanding their development.

Among the well-known optimization algorithms, we can mention the genetic algorithm (GA). This algorithm is a technique of optimization firstly proposed by Holland (Holland, 1975). This algorithm is based on the idea of evolution in the nature and it looks into the problem on a fully random basis.

This method is based on some biological techniques like genetic and mutation; the search is carried out in order to find better responses in each generation compared to the previous one. Among the features of the genetic algorithms are its ability to run in parallel way and its capability for searching very large and complicated spaces (Goldberg, 1989; Chang et al., 2010).

Another well-known optimization algorithm is particle swarm optimization (PSO) (Clerc and Kennedy, 2002), which was developed for the optimization problems, and its features have so far been proved to be useful for both continuous and discrete functions. In this algorithm each particle is considered as a member of the society using the experiences of its previous particle as well as that of others in order to reach the ultimate goal. This algorithm is able to find the global optimum of the function in question in consequent iterations (Chakraborty et al., 2010).

Simulated annealing (SA) algorithm was presented basically intending to simulate between minimizing the target function of a problem and cooling until the time it

*Corresponding author. E-mail: mtaherdangkoo@yahoo.com or taherdangkoo@shirazu.ac.ir. Tel: +98-9128243251.

gets to a state of basic energy (Kirkpatrick et al., 1983). In this algorithm, the first response is an important parameter and plays an important role and changes according to the type of problem.

Stem cells algorithm (SCA) has been designed based on the biological behavior of stem cells in order to restore or evolve the damaged organ in the matured human body (Taherdangkoo et al., 2011; Taherdangkoo et al., 2012b). This algorithm uses the features of stem cells like self-renewal and the ability to evolve in a complete organ.

The artificial bee colony (ABC) algorithm has been designed based on the social behavior of honey bees in their colonies looking for food sources. This algorithm has many advantages and some disadvantages caused that some constraints and parameters have been defined in order to resolve the disadvantages of this method and to increase the convergence rate in different problems, yielding a better performance (Karaboga and Basturk, 2007; Akay and Karaboga, 2010).

Among key components of swarm intelligence we can mention self-organization and division of labor. Group cooperation is the key to reach the optimal and ideal response within shortest period of time.

In this paper, we have used BNMR algorithm for numerical functions optimization, and we use some Benchmark functions with high dimensions in order to prove better performance of this algorithm compared with other well-known optimization algorithms. The comparison part indicates better convergence rate and more precision in achieving the optimum response compared with other optimization algorithms.

BLIND, NAKED MOLE-RATS COLONY IN REAL WORLD

The blind naked mole-rats live in underground tunnels in some African countries and in green areas with almost fixed temperature and humidity. The temperature in the tunnels where the blind naked mole-rats live must always be between 30 to 35°C. Each tunnel has nearly a length of three kilometers (Brett, 1991; Bennett and Jarvis, 1988). These animals are the only small mammals that live in community and in large colonies, just like honey bees or ant colonies. Each colony consists of three groups of moles as follows:

1. A queen accompanied by one to three males that carry out the reproduction process.
2. Employed moles that search for food sources and build suitable nests for the queen and the offspring's.
3. Soldier moles that are responsible for cleaning the tunnels and protect the colony against the invasions.

In a real colony, the number of soldier moles is much higher than the employed moles. The tunnels where the blind naked mole-rats live are built with a unique regularity. These tunnels have chambers built for special purposes: kitchen room is used for collecting the food, toilet room is used for placing the wastes, and when they are filled up, another chamber is dug for this purpose. There are also some labyrinthine rooms. When invaders enter the tunnel, the soldiers quickly block their break into the tunnel with their wastes. Where their speed for this process is low, one of the soldiers sacrifices his life in order to help other soldiers find adequate time to stop the invaders breakthrough.

BLIND, NAKED MOLE-RATS (BNMR) ALGORITHM

The BNMR algorithm is an optimization algorithm, designed based on the social behaviour of blind naked mole-rats in a large colony. The research process starts from the centre of the colony, where the queen and offspring live. Note that for simplification purposes, we have placed the employed moles and soldier moles in one single group, which hereby is called the employed moles.

In the beginning with the production of the initial population of the blind naked mole-rats colony starts working in the whole problem space on a completely random way. Note that the number of the population is two times of the number of food sources and each of the food sources represents a response for problem space. Let us define some parameters as follows:

$$Members\ of\ BNMR = [M_1, M_2, ..., M_N] \quad (1)$$

Where N is the number of members related to the number of problem's unknown parameters. Initial production of food sources within the parameter's borders is defined as follows:

$$x_i = x_i^{Min} + \beta(x_i^{Max} - x_i^{Min})$$
$$i = 1, ..., S \quad (2)$$

Where x_i represents the i^{th} food source and β is a random variable in the interval of [0, 1] and S represents the number of food sources.

Hereby, the food sources (responses) in the search process are considered as targets to be found by employed moles (for example, finding the location of food sources and their neighbours, determining the volume of enrichment, discharging food sources and storing them in the kitchen room).

The random movement of employed moles starts from the centre of the colony towards food sources and their neighbours. Moreover, the employed moles carry out the process of digging labyrinthine tunnels to find food sources.

The conditions of food sources in terms of temperature and humidity must always be suitable and almost stable.

These conditions are considered as an attenuation coefficient in the form of a random variable in the interval of [0, 1] in our algorithm in order to determine the movement of the employed moles from the target food sources towards their neighbors. In addition, we should consider the underground temperature in our algorithm defined as follows:

$$H(x) = \rho(x)C(x)\frac{\Delta T(x,t)}{\Delta t}$$
$$(\rho C) = f_s(\rho C)_s + f_a(\rho C)_a + f_w(\rho C)_w$$
$$f_s + f_a + f_w = 1 \quad (3)$$

Where H represents the soil temperature changing with the depth x as its variable. $\rho(x)$ and $C(x)$ are the thermal properties of the soil, i.e. the density and specific heat capacity respectively. ρ and C are variable related to x which means that during the movement, they vary with area's changes, however, due to the simplification of the algorithm, we considered them as constant. Empirically, the multiplication of ρ and C should be a value in the range of $[2\ \ 4]$. $\frac{\Delta T(x,t)}{\Delta t}$ shows the rate of the soil temperature changing with the time and it will be updated at each iteration in the proposed algorithm. The symbol f represents the volumetric contribution (in percent) of each element in the compound, while, the subscripts s, a, w indicate the components of the soil, for example sand, air and water, respectively. The sum of these volumetric percents is equal to unity in order to have a weighted averaging sum to obtain ρC. In moving towards searching the neighbors of food sources, the attenuation coefficient A must be updated in each iteration. When the temperature arrives at zero (for example, in real conditions where temperature is inclined towards 30°C), the search of neighborhoods of food sources is carried out with lower intensity and while the temperature is close to average volume in the interval of [0, 1], the search of the neighborhoods of food sources is carried out with higher intensity. We consider this fact by the following equation:

$$A_i^t = \mathrm{A}_i^{t-1}[1 - \exp(\frac{-\alpha t}{T})] \quad (4)$$

Where T is derived from equation (3) and α is a random variable in the interval of [0, 1], but for the sake of simplification in algorithm's implementation and calculation, we consider $\alpha = 0.95$ as a fixed parameter, and t is the iteration step.

Next, for each food source, two employed moles are sent, so the number of food sources is the half of the number of employed moles in the colony. After finding foods, their original quality and the path to reach them is saved in the memory of employed moles. In returning to the colony, the employed moles that carry information regarding the food sources share the information with others and with the queen. The obtained food sources are categorized with the probability P by the queen within a table sorted from the richest to lowest values, based on both the original quality of those food sources and the shorter paths to reach the food sources from the centre of the colony. The probability P is computed as follows:

$$P_i = \frac{Fitness_i = FS_i \times R_i}{\sum_{j=1}^{N} Fitness_j} \quad (5)$$

Where $Fitness_i$ is evaluated by its employed moles, FS_i is related to the richest food source, R_i is related to the route for reaching food source and N is the number of food sources which is the half of the number of employed moles.

The first food source with highest probability is select by the queen. Then, the two employed moles that carry the information of this food source are selected to do the recruitment process, for moving toward the food source. After reaching this food source, one of the two employed moles, which is the head of the group, randomly chooses some of the soldiers and leads the process of collecting the food. The other employed mole becomes the head of another group and by choosing some other soldiers starts searching the neighborhood area of the food source. All accompanying moles will carry information regarding the existence of food resources in the area adjacent to the main food source. The process for collecting the food from the food resources and concurrently searching their neighbors will reduce significantly the time to get to an optimum response in the implementation of the proposed algorithm and consequently increase the convergence rate. This process is repeated for all food resources along with their neighborhoods until no food resource is available.

Should be note, the search for neighbors of each food source is done in different directions starting from the food source. The directions are in 45 degrees from each other. So, there are 8 directions for searching the neighbors of a food source.

The next part of implementing the BNMR algorithm includes the consideration of defending the colony and preventing the invaders' break-in into the tunnels. In implementing our proposed algorithm, those points with low cost function are determined in each iteration and considered as invaders and they are eliminated from the implementation process (that is, they are not considered as members of the whole population involved in the process). The number of eliminated points in each iteration is augmented related with respect to that in the previous iteration by a factor, which is fixed by the

designer. It is computed by the following equation:

$$B_i^t = \zeta \times B_i^{t-1} \quad (6)$$

Where $\zeta \geq 1$ is a factor, which is set by the designer and B_i^t is the number of eliminated points for i^{th} food source in iteration *t*. Note that we replace the eliminated points in each iteration by new ones randomly selected from the space of selection.

Using this idea, we apply some kinds of mutation process in the proposed algorithm, so that the new members having new information will prevent the algorithm being trapped in local minimums.

The pseudo code of the BNMR algorithm as follows:

```
Begin
    Objective Function, f(x), X = [x_1, x_2, ..., x_N]^T
    Initialize population, X_i, { i = 1, ..., N} and removal rate, B_i, { i = 1, ..., N}.
    Sort the initial population based on their objective function value.
    Define α, β, ζ, close neighbor and far neighbor from candidate food source.
    while ( t < Maximum number of iteration )
    Generate new solution by equation (2).
        if ( rand < A_i )
        Select a solution among the best solution by equation (5).
        Generate a local solution (neighbors around candidate food source) from previous step.
        end if
        Generate a new solution by search randomly
        if ( rand < B_i and f(x_i^t) < f(x_Optimum^t))
        Accept and save the new solutions in the memory of mole−rat
        end if
    search for find the new solutions in current iteration and compare with the solutions in previ
    end while
    end (obtain the optimum response of the BNMR algorithm )
```

EXPERIMENTAL RESULTS

Test Benchmark Function

In order to assess the performance of our proposed algorithm that of other introduced optimization algorithms, we have compared the efficiency and the accuracy of BNMR algorithm using the 20 Benchmark functions (Suganthan, 2005) and the accuracy of all the mentioned algorithms using eight Benchmark functions (Suganthan, 2005). For all Benchmark functions, 50 independent runs were applied with different random seeds for generating the random variables, each contains 5000 iterations, and the population size was set to 100 for all optimization algorithms mentioned before except the BNMR algorithm. Note that when the space size goes up (for example more that 50 (Dimension>50)) the performance of optimization algorithms drops dramatically because the algorithm will face with several optimal answers. In this case of distribution, the selected answer in that considered space also adds complexity to the subject randomly. Table 1 shows the list of Benchmark functions.

Settings for algorithms

The number of Maximum generation and population size are common control parameters of the algorithms. In the experiments, maximum number of generation for the dimension of 10, 20, 30 and 40 are considered. Table 2 shows other control parameters of the algorithms and the schemes, with the value of these control parameters applied for GA, PSO, SA, SCA, ABC and BNMR algorithm.

Results comparison

We considered the performance of the BNMR algorithm and comparison between all mentioned optimization algorithms using 24 benchmark functions (Suganthan et al., 2005).

These 24 functions consist of two categories. The first one is Unimodal functions: { F_1 (Shifted Sphere Function), F_2 (Shifted Schwefel's problem 1.2), F_3 (Shifted Rotated High Conditioned Elliptic Function), F_4 (Shifted

Table 1. The list of Benchmark functions.

Function	Global Min	Search range	Initial range	Formulae
F_{21}	0	[-5, 5]	[-5, 0]	Suganthan et al. (2005)
F_{22}	0	[-5, 5]	[-5, 0]	Suganthan et al. (2005)
F_{23}	0	[-5, 5]	[-5, 0]	Suganthan et al. (2005)
F_{24}	0	[-5, 5]	[-5, 0]	Suganthan et al. (2005)
Ackley	0	[-32.7, 32.7]	[-32.7, 16]	$20+e-20e^{\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)}-e^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)}$
Rosenbrock	0	[-2.04, 2.04]	[-2.04, 0]	$\sum_{i=1}^{D-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$
Schwefel	0	[-500, 500]	[-500, 200]	$418.9829xD-\sum_{i=1}^{D}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$
Weierstrass	0	[-0.5, 0.5]	[-0.5, 0.2]	$\sum_{i=1}^{D}\left(\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k\left(x_i+0.5\right)\right)\right]\right)$ $-D\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k 0.5\right)\right]$, $a=0.5, b=3, k_{\max}=20$

Table 2. Values of parameters of each of six algorithms.

Algorithm	Parameter	Value
GA [2]	Population	100
	Crossover	0.95
	Mutation rate	0.001
	Number of iterations	5000
PSO [5]	Number of Swarm	100
	$\varphi_1=\varphi_2$	2
	$w_{\min}$	0.7
	$w_{\max}$	0.9
	Number of iterations	5000
SA [6]	Probability threshold	0.98
	Initial temperature	5
	Temperature multiplier	0.98
	Final temperature	0.01
	Number of iterations detect steady stat	500
	Number of iterations	5000
SCA [7]	Population	100
	$\zeta_{\min}$	0.01 or 0.98
	$\zeta_{\max}$	0.98
	Number of Iterations	5000

Table 2. Contd.

ABC [9]	Population	100
	Number of sites selected for neighborhood search	10
	Number of bees recruited for best sites	5
	Number of iterations	5000
BNMR	Population	100
	α	0.95
	ζ	0.95
	Number of iterations	5000

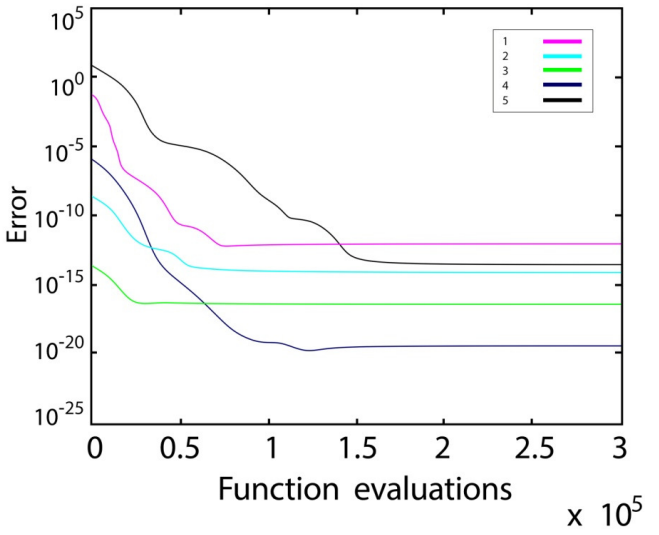

Figure 1. The results of applying the BNMR on F_1-F_5 (Convergence of Functions F_1-F_5).

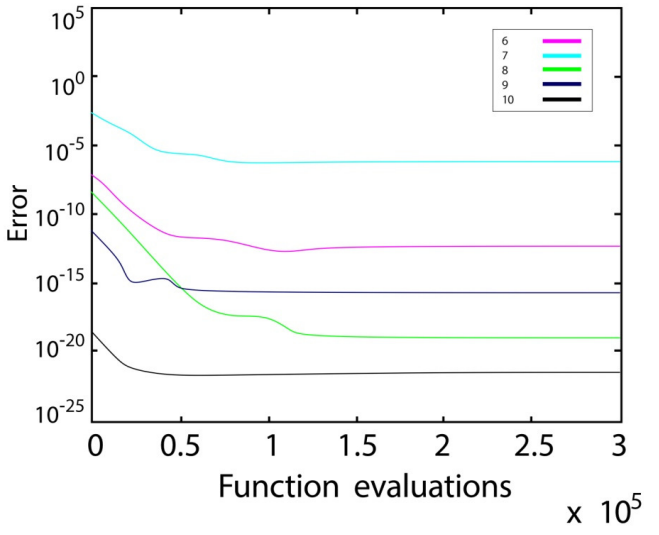

Figure 2. The results of applying the BNMR on F_6-F_{10} (Convergence of Functions F_6-F_{10})

Functions: F_6 (Shifted Rosenbrock's Function), F_7 (Shifted Rotated Griewank's Function without Bounds), F_8 (Shifted Rotated Ackley's Function with Global Optimum on Bounds), F_9 (Shifted Rastrigin's Function), F_{10} (Shifted Rotated Rastrigin's Function), F_{11} (Shifted Rotated Weierstrass Function), F_{12} (Shifted Schwefel's Problem 2.13)}, {Expanded Functions: F_{13} (Expanded Extended Griewank's plus Rosenbrock's Function), F_{14} (Shifted Rotated Expanded Scaffer's)}, {Hybrid Composition Functions: F_{15} (Shifted Rosenbrock's Function), F_{16} (Shifted Rotated Griewank's Function without Bounds), F_{17} (Shifted Rotated Ackley's Function with Global Optimum on Bounds), F_{18} (Shifted Rastrigin's Function), F_{19} (Shifted Rotated Rastrigin's Function), F_{20} (Shifted Rotated Weierstrass Function), F_{21} (Shifted Schwefel's Problem 2.13, F_{22} (Shifted Schwefel's Problem 2.13, F_{23} (Shifted Schwefel's Problem 2.13, F_{24} (Shifted Schwefel's Problem 2.13, and the properties of these 24 functions are available in (Suganthan et al., 2005).

Each algorithm has been executed 50 independent times and every time with different random seeds for generating the random variables for each Benchmark function. Figures 1, 2, 3 and 4 show the convergence of the BNMR algorithm on the Benchmark functions F_1 to F_{20} by the number of fitness evaluations, and Figures 5, 6, 7 and 8 show the convergence of all the mentioned algorithms on the Benchmark functions F_{21}, F_{22}, F_{23}, F_{24} by the number of fitness evaluations, and Figures 9, 10, 11 and 12 show the performance of all the mentioned algorithms on Ackley, Rosenbrock, Schwefel and Weierstrass functions by the number of Iteration.

We want to show the comparison of the convergence of all the mentioned optimization algorithms by the number of fitness evaluations and the number of iterations. As can be seen, the BNMR algorithm has better convergence than the other optimization algorithms. In the other words, this proves that BNMR algorithm has the

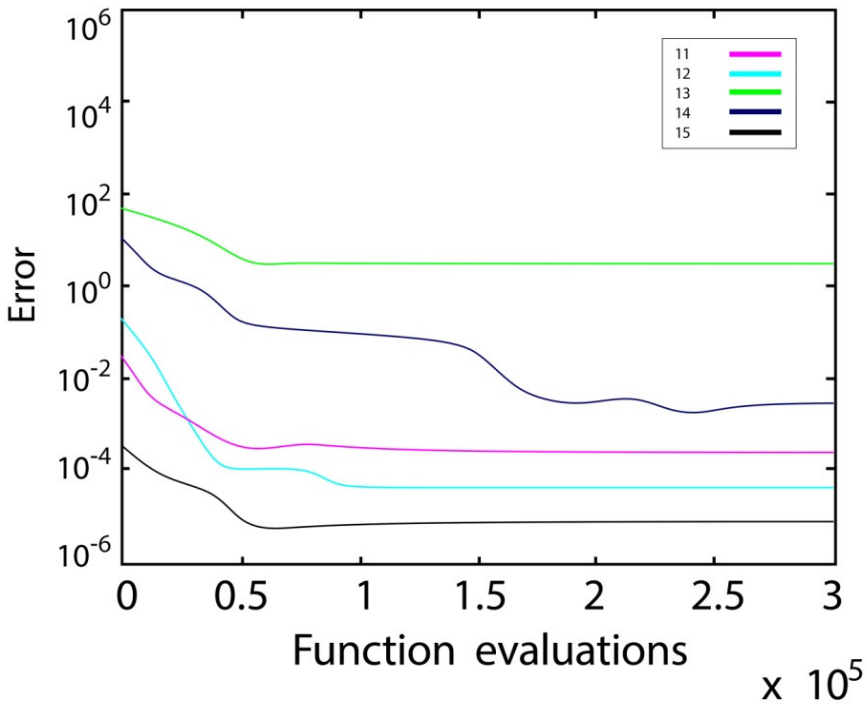

Figure 3. The results of applying the BNMR on $F_{11}-F_{15}$ (Convergence of Functions $F_{11}-F_{15}$).

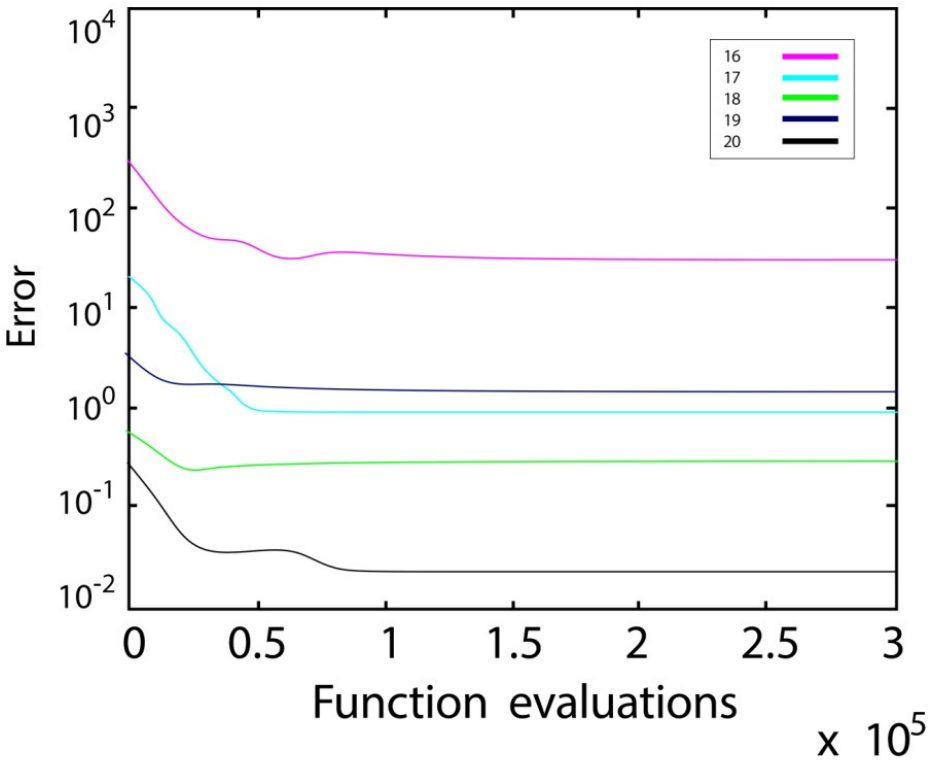

Figure 4. The results of applying the BNMR on $F_{16}-F_{20}$ (Convergence of Functions $F_{16}-F_{20}$).

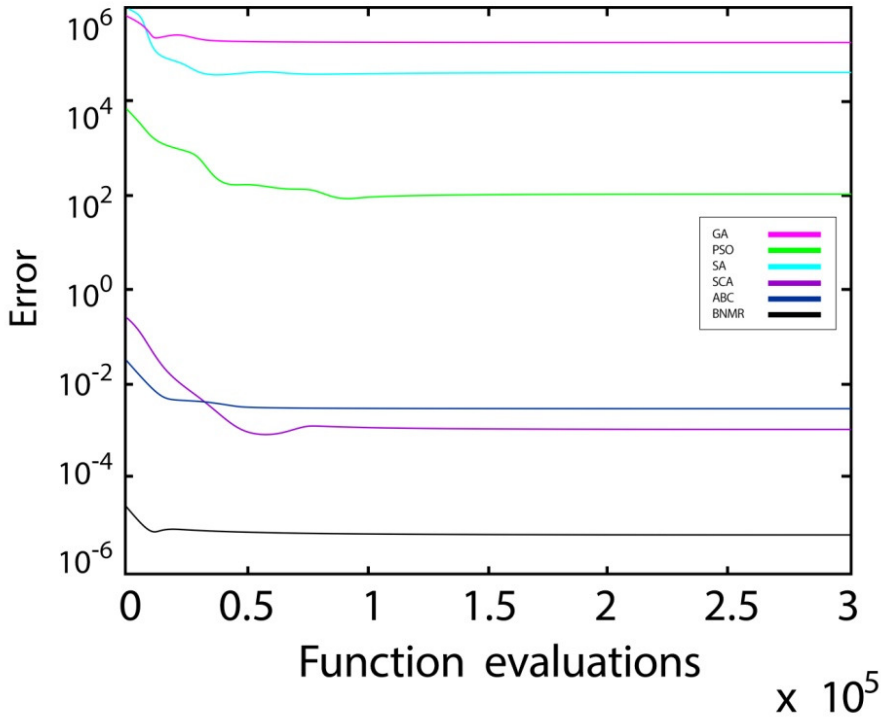

Figure 5. The results of applying the different algorithms on F_{21} (Convergence of Function F_{21} for all algorithms).

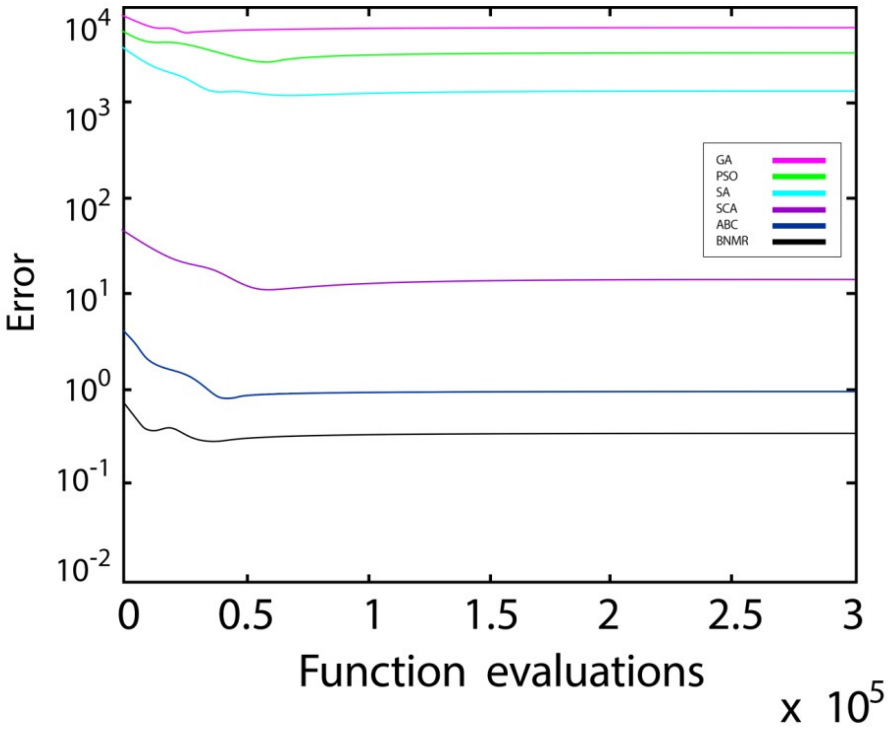

Figure 6. The results of applying the different algorithms on F_{22} (Convergence of Function F_{22} for all algorithms).

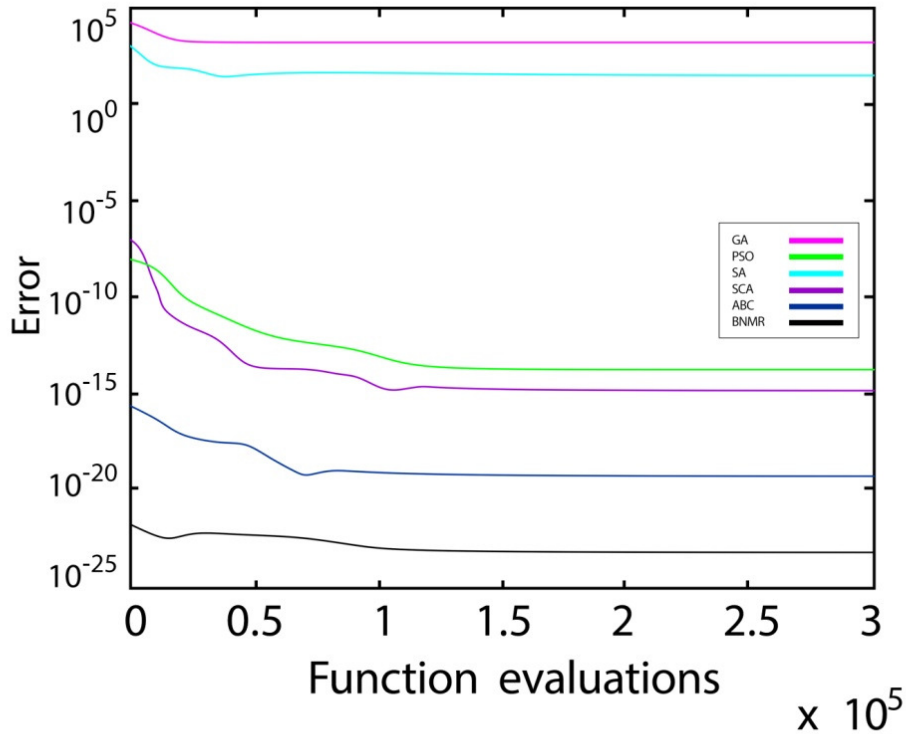

Figure 7. The results of applying the different algorithms on F_{23} (Convergence of Function F_{23} for all algorithms).

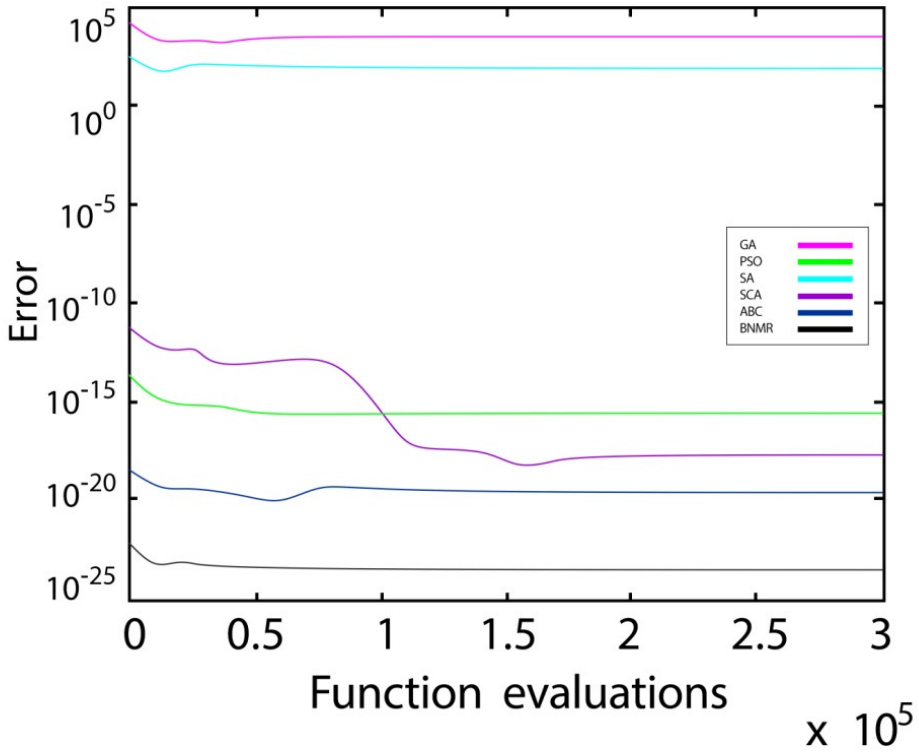

Figure 8. The results of applying the different algorithms on F_{24} (Convergence of Function F_{24} for all algorithms).

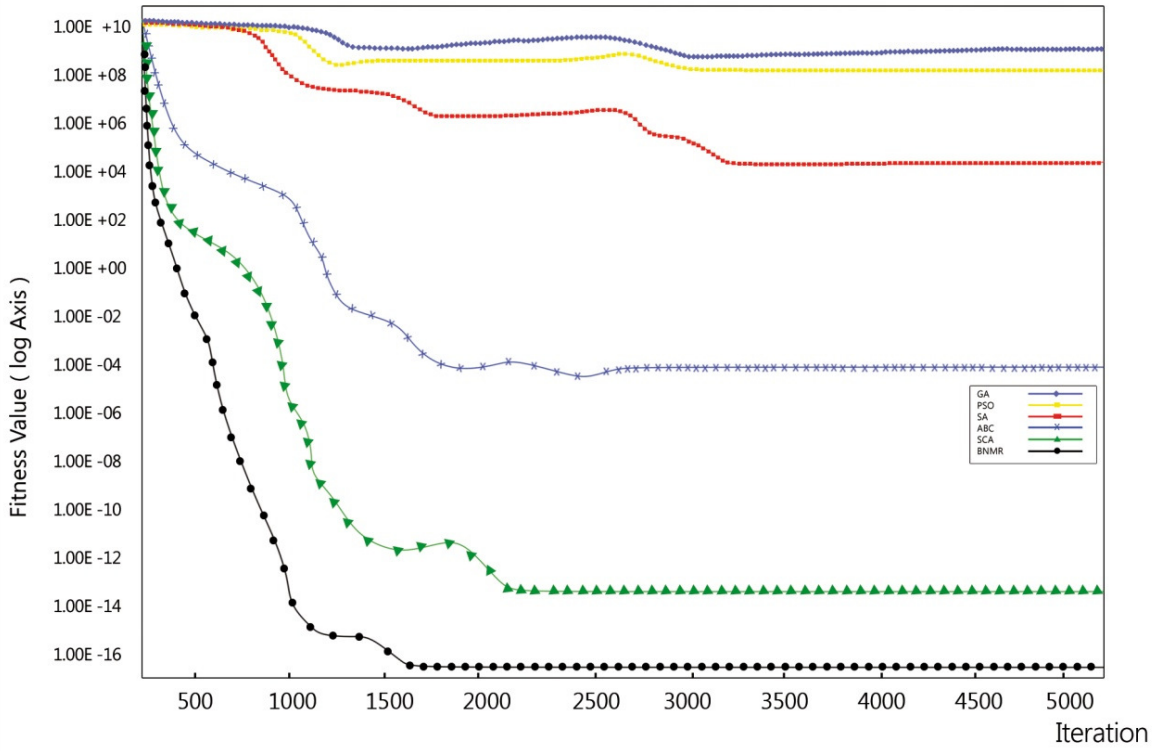

Figure 9. The results of applying different algorithms on Ackley function.

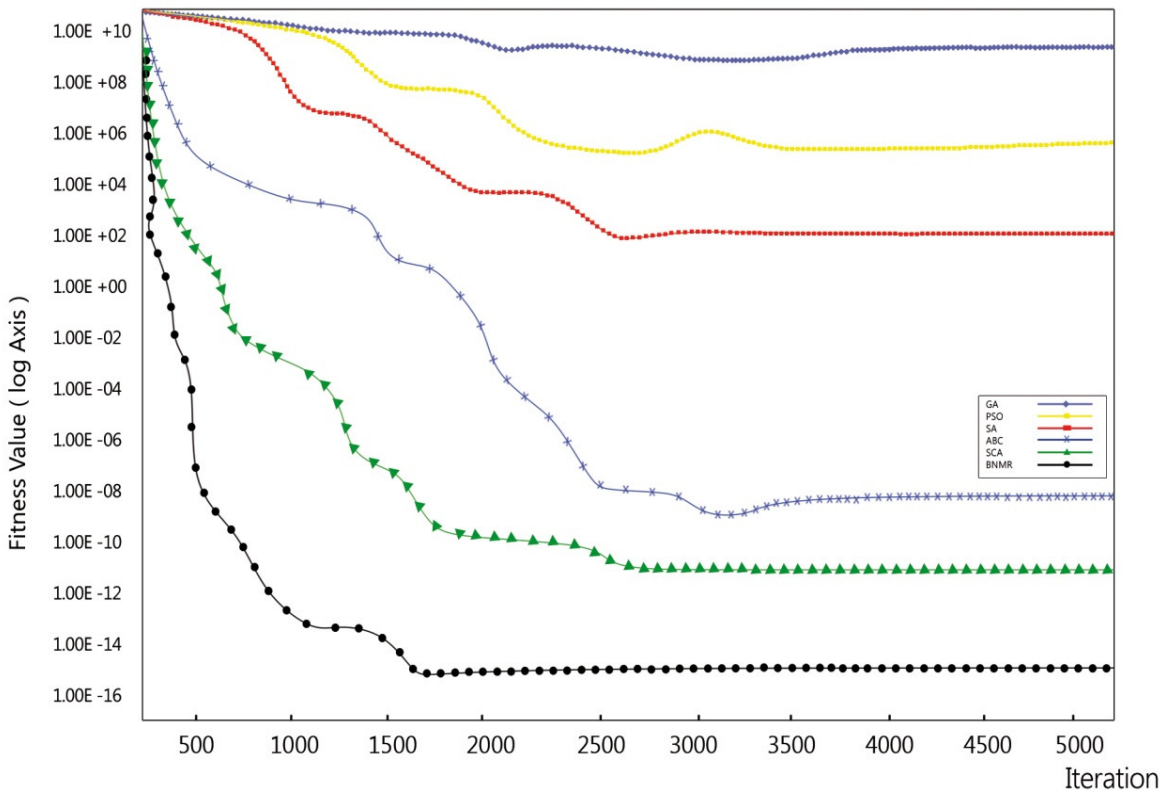

Figure 10. The results of applying different algorithms on Rosenbrock function.

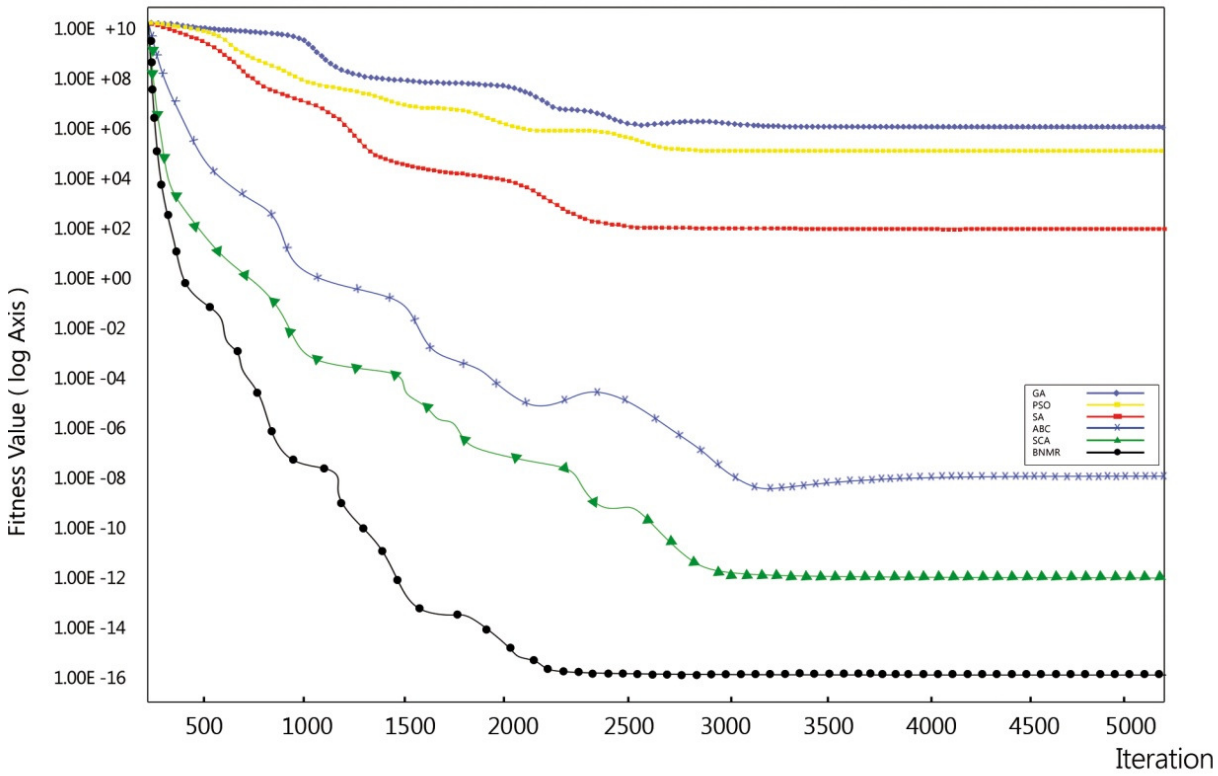

Figure 11. The results of applying different algorithms on Schwefel function.

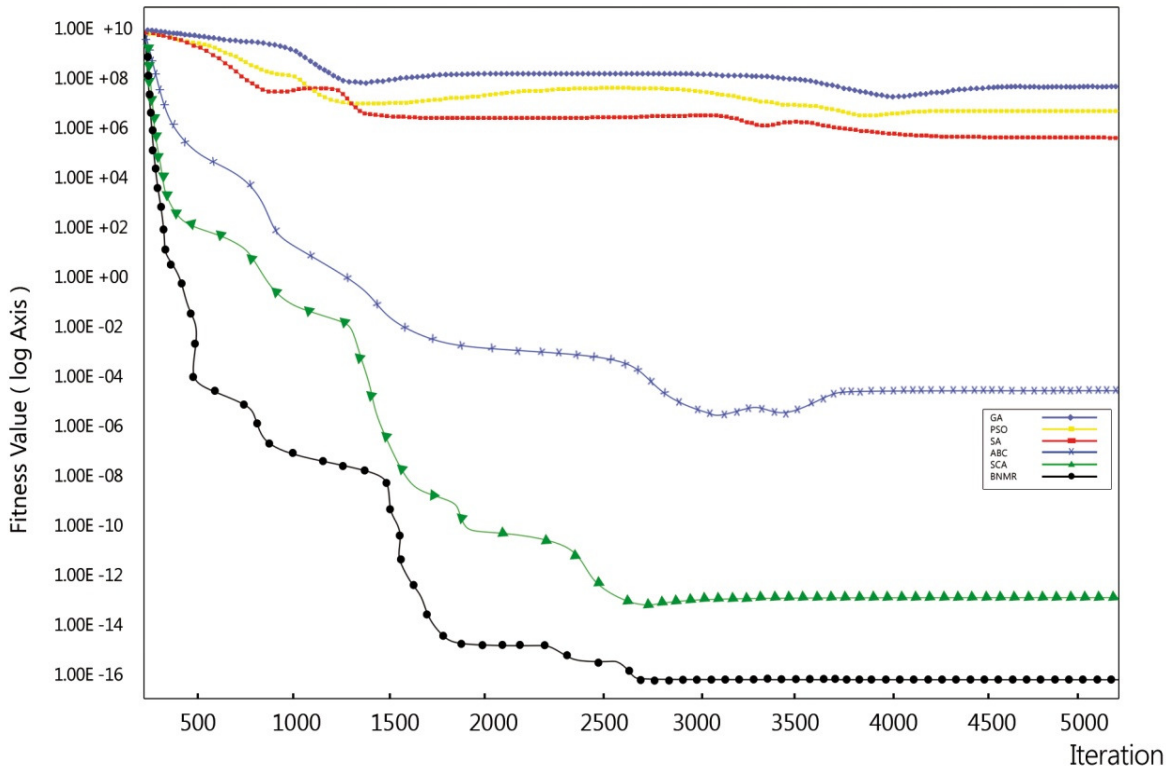

Figure 12. The results of applying different algorithms on Weierstrass function.

Table 3. The final results of mean and standard derivations (SD) of different algorithms for F_{21} function.

Dimensions		10	20	30	40
GA	Mean	6.015	9.182	12.077	19.016
	SD	3.088	7.087	11.247	14.085
PSO	Mean	1.58E-04	2.03E-04	7.46E-01	9.42E+02
	SD	2.67E-04	2.36E-03	1.84E-00	3.18E-01
SA	Mean	4.637	7.537	13.544	22.007
	SD	3.019	6.648	12.638	18.374
SCA	Mean	1.67E-11	4.36E-09	6.87E-07	9.24E-07
	SD	0.78E-12	4.98E-11	7.86E-08	8.21E-07
ABC	Mean	9.87E-11	1.26E-10	3.77E-09	4.65E-08
	SD	4.21E-12	7.98E-10	8.36E-09	3.81E-07
BNMR	Mean	2.12E-15	2.29E-14	4.27E-13	6.84E-11
	SD	2.56E-14	2.78E-13	5.36E-13	7.11E-13

Table 4. The final results of Mean and Standard Derivations (SD) of different algorithms for F_{22} function.

Dimensions		10	20	30	40
GA	Mean	17.049	29.018	41.297	53.709
	SD	11.03	15.025	32.123	38.283
PSO	Mean	6.465	11.064	14.837	26.024
	SD	7.0037	9.0026	16.048	21.074
SA	Mean	4.036	14.076	25.086	31.097
	SD	3.019	8.024	17.028	26.842
SCA	Mean	1.17E-03	2.23E-01	6.11E+01	1.16E+02
	SD	2.5E-02	7.21E-01	1.12E+01	1.27E+02
ABC	Mean	1.02E-00	1.27E+01	1.67E+02	6.34E+02
	SD	3.12E-00	1.38E+01	1.89E+02	2.88E+02
BNMR	Mean	0.16E-04	1.28E-03	1.78E-03	3.18E-01
	SD	2.56E-05	2.33E-04	1.24E-04	1.22E-02

ability of getting out of a local minimum in the problem space and achieving the global minimum. So, we get better performance using BNMR algorithm in optimizing multimodal and multivariable functions.

For different dimensions such as 10, 20, 30 and 40 the mean and standard derivations (SD) function values of the best solution found by the algorithms have been saved by GA, PSO, SA, SCA, ABC and BNMR that are shown in Tables 3 to 10.

Meanwhile, we have studied the effect of scalability on the computational complexity of the BNMR algorithm for Weierstrass function as described in (Suganthan et al., 2005). We also have computed the CPU Time; the time when the algorithm achieves its best results, of each algorithm on different mentioned Benchmark functions. Tables 11 to 18 show these times. The characteristics of

Table 5. The final results of Mean and Standard Derivations (SD) of different algorithms for F_{23} function.

Dimensions		10	20	30	40
GA	Mean	4.632	9.018	16.871	23.097
	SD	3.673	7.762	12.406	20.742
PSO	Mean	0.072	0.274	1.527	3.036
	SD	0.024	0.089	0.178	1.365
SA	Mean	3.627	6.979	11.046	27.187
	SD	1.496	4.867	9.956	25.243
SCA	Mean	1.23E-14	1.28E-13	7.86E-12	1.22E-09
	SD	3.45E-14	2.66E-13	1.29E-13	3.89E-11
ABC	Mean	3.67E-09	2.11E-12	6.87E-09	2.45E-06
	SD	2.77E-10	1.86E-14	7.78E-09	1.77E-07
BNMR	Mean	1.87E-15	1.88E-14	7.78E-13	1.77E-12
	SD	2.85E-16	3.67E-16	8.66E-14	2.56E-13

Table 6. The final results of Mean and Standard Derivations (SD) of different algorithms for F_{24} function.

Dimensions		10	20	30	40
GA	Mean	21.192	32.976	41.338	47.638
	SD	11.764	14.275	19.588	28.872
PSO	Mean	0.0978	0.1763	0.6012	0.927
	SD	0.0082	0.0312	0.165	0.214
SA	Mean	4.762	6.436	22.427	31.004
	SD	3.826	5.982	16.536	19.739
SCA	Mean	1.17E-12	1.76E-11	8.47E-11	2.08E-10
	SD	1.05E-13	8.16E-13	0.43E-11	9.83E-11
ABC	Mean	3.68E-11	1.72E-10	1.84E-08	2.67E-06
	SD	1.88E-11	2.76E-09	4.56E-09	3.01E-05
BNMR	Mean	0.12E-14	1.36E-14	2.17E-14	0.02E-13
	SD	1.07E-16	3.64E-14	0.65E-13	9.89E-13

the computer on which the algorithms were run are: Macintosh OS, Two 2.93 GHz 6-Core Intel, 64 GB Ram, and the graphic card is ATI Radeon HD 5870 1 GB.

Moreover, code execution time (T_0), execution time of Weierstrass function for 200,000 evaluations (T_1) and for five runs, mean of the BNMR algorithm execution times on Weierstrass function for 200,000 evaluations ($\hat{T}_2$) were computed. The complexity of algorithm was computed by $\{(\hat{T}_2 - T_1)/T_0\}$ and given in Table 19.

Most of the mentioned optimization algorithms in this paper achieved good results but not as well as the proposed method (BNMR algorithm) since the other algorithms are intensively depended on the adjustment of their parameters. For instance, in ABC algorithm, we

Table 7. The final results of Mean and Standard Derivations (SD) of different algorithms for Ackley function.

Dimensions		10	20	30	40
GA	Mean	0.63	0.89	1.13	3.78
	SD	0.32	0.33	0.37	0.57
PSO	Mean	6.8E-12	4.6E-07	3.6E-06	3.2E-03
	SD	7.2E-13	6.2E-08	5.7E-06	6.1E-02
SA	Mean	0.28	0.56	0.63	0.97
	SD	0.29	0.31	0.42	0.73
SCA	Mean	0	0	1.01E-16	1.26E-13
	SD	0	0	1.07E-14	1.28E-14
ABC	Mean	0	0	8.1E-11	6.2E-09
	SD	0	0	6.1E-12	1.6E-09
BNMR	Mean	0.6E-16	2.56E-16	4.66E-16	3.78E-15
	SD	0.3E-18	0.46E-17	2.66E-17	7.86E-16

Table 8. The final results of Mean and Standard Derivations (SD) of different algorithms for Rosenbrock function.

Dimensions		10	20	30	40
GA	Mean	2.563	7.865	13.757	23.456
	SD	1.763	3.678	9.457	17.642
PSO	Mean	0.876	1.342	9.453	17.453
	SD	1.124	3.651	7.563	15.674
SA	Mean	5.753	11.456	19.453	27.567
	SD	4.436	8.456	16.456	31.456
SCA	Mean	0.06E-03	1.24E-03	4.46E-01	7.66E-00
	SD	1.02E-04	7.26E-03	3.77E-03	7.66E-01
ABC	Mean	1.25E-02	1.76E-00	7.14E+1	2.16E+3
	SD	2.36E-02	5.76E-00	1.17E+1	0.68E+2
BNMR	Mean	0.87E-07	1.13E-07	1.56E-04	6.87E-02
	SD	1.02E-08	1.45E-07	4.33E-04	6.86E-03

need to change its parameters (for example, MR, SF and etc.) for each function to obtain a good result for that function and due to this fact the algorithm cannot be an ideal one.

DISCUSSION

The genetic algorithm implements the law of survival with a focus on the existing solutions in order to reach a better solution. The genetic algorithm suffers a lot from too much dependency on such techniques such as selection, mutation, crossover, etc. with the conditions of the problem and the initial conditions and a weak selection of these constraints and parameters have got a remarkable influence on the function of genetic algorithm (GA). Of course, despite the existence of optimum methods presented to complete the GA, it is still damaged in the

Table 9. The final results of Mean and Standard Derivations (SD) of different algorithms for Schwefel function.

Dimensions		10	20	30	40
GA	Mean	10.844	22.783	41.983	78.564
	SD	9.032	17.637	36.272	47.782
PSO	Mean	2.567	6.547	11.365	21.457
	SD	1.098	3.093	7.543	18.366
SA	Mean	5.864	12.043	17.634	32.764
	SD	3.536	7.837	11.635	17.623
SCA	Mean	1.67E-05	1.69E-03	3.88E-01	7.65E-01
	SD	4.64E-05	7.35E-04	3.65E-02	2.35E-02
ABC	Mean	2.67E-03	3.65E-01	7.63E+01	8.71E+03
	SD	3.03E-03	2.11E-02	5.34E+1	6.34E+02
BNMR	Mean	2.11E-07	7.52E-06	3.65E-03	8.76E-03
	SD	1.22E-07	3.56E-06	4.89E-04	9.24E-03

Table 10. The final results of Mean and Standard Derivations (SD) of different algorithms for Weierstrass function.

Dimensions		10	20	30	40
GA	Mean	0.123	0.632	1.352	7.653
	SD	0.023	1.865	2.768	4.786
PSO	Mean	0.0002	0.0096	0.0167	1.677
	SD	0.0034	0.0082	0.0734	0.0962
SA	Mean	0.008	0.028	0.787	1.256
	SD	0.001	0.007	0.154	1.234
SCA	Mean	0	0	1.22E-08	2.66E-06
	SD	0	0	2.11E-07	4.19E-06
ABC	Mean	0	0	0	6.98E-11
	SD	0	0	0	2.66E-09
BNMR	Mean	1.22E-12	2.55E-11	8.66E-11	1.02E-10
	SD	2.09E-12	3.66E-11	4.88E-10	2.99E-09

Table 11. CPU time obtained by applying the different algorithms on F_{21} function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	17.0017	9.0827	13.9038	3.0076	4.1073	1.2705

problems with continuous and discontinuous spaces with high dimensions of early convergence or repeated interruptions. For instance, where some of weak people join the referenced set and form it and the continuation of

Table 12. CPU time obtained by applying the different algorithms on F_{22} function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	19.0086	8.1026	15.6728	3.2618	3.1704	2.0074

Table 13. CPU time obtained by applying the different algorithms on F_{23} function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	19.1703	9.1763	14.0762	2.9968	3.5603	2.3607

Table 14. CPU time obtained by applying the different algorithms on F_{24} function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	18.0936	7.9825	13.5638	2.0216	2.0063	1.1607

Table 15. CPU time obtained by applying the different algorithms on Ackley function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	9.0028	5.1004	8.9827	1.0034	1.9008	0.9162

Table 16. CPU time obtained by applying the different algorithms on Rosenbrock function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	9.1076	5.2256	9.0034	1.2607	1.7683	0.8901

Table 17. CPU time obtained by applying the different algorithms on Schwefel function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	9.2674	4.9907	8.4013	1.6003	1.4108	0.7243

Table 18. CPU time obtained by applying the different algorithms on Weierstrass function.

Method	GA	PSO	SA	SCA	ABC	BNMR
Real Time in Second	9.3267	5.1806	7.6482	1.0764	2.0002	0.5624

Table 19. Time complexity of the BNMR algorithm on Weierstrass function.

Dimension	T_0	T_1	$\hat{T}_2 = Mean(T_2)$	Complexity $\{(\hat{T}_2 - T_1)/T_0\}$
10	0.31845715929317	0.18714673563299	0.39948641607526	0.66677632
30	0.31845715929317	0.18947360835546	0.46656005091724	0.87009014
50	0.31845715929317	0.19544465678268	0.53936376385198	1.07995407
100	0.31845715929317	0.19987782534738	0.69177629442021	1.54462996

examinations in the sample space is stopped, one of the reasons could be this issue that a very appropriate chromosome is selected for reproduction. As a result, the offspring becomes similar to its parent and this is when the chromosomes are very similar to each other. Thus, before reaching the optimal solution, the early convergence is formed. Of course, through applying constraints, we can prevent it, but this is when the dimensions of the issue do not rise.

The particle swarm optimization (PSO) has got a very simple implementation, but it suffers from early convergence. Although PSO algorithm has got a more reasonable speed than other optimization algorithms (Deep and Bansal, 2009), but it cannot optimize the quality of solutions by increasing the number of repetitions. This issue is more visible when examining and optimizing the multi-model problems. The reason for its occurrence in $gbest$ PSO is that the particles become convergent in one specific point, while this point is located on one line, between the best global position and the best individual position. The other problem arises from too much dependency to regulating the PSO algorithm parameters (Kang et al., 2012b).

The simulated annealing (SA) algorithm (Oliveira et al., 2012), Initial solution and neighboring formation mechanism are important parameters of the SA algorithm that play significant roles in reaching into the optimum solution.

After selecting these two parameters, an initial temperature analogous to the kinetic energy is attributed to the system. Selecting initial value is arbitrary, but is applied depending on the behavior of the function at the start point. This SA algorithm process raises the time needed to reach the optimum solution and also increases the repetitions. SA is scrutinized in different problems and by many researchers but one of its main deficiencies is the large time in reaching into the optimum solution that is still unsolved.

The self-renewal process plays an important role in the SCA algorithm (Taherdangkoo et al., 2012a), but the presence of symmetry (symmetric and asymmetric propagation) has increased the time of obtain the optimum solution and it practically reaches the final solution by high repetitions. This is considered as one of the main problems of this algorithm.

The artificial bee colony (ABC) algorithm has got more accuracy and speed, than other optimization algorithms, but the use of roulette wheel in selecting the employed bees and the relation of employed bees and onlooker bees has not been turned into a model appropriately, and this issue does not make this algorithm distinguished from other optimization algorithms, and finally a research for modified of basic of artificial bee colony algorithm, was able to solve the main problem of ABC algorithm by using controlled of phase and magnitude and could achieve reasonable results (that should be note that in our proposed method, the same algorithm has been used for comparison with proposed methods, but they have been included within the context as ABC algorithm) (Kang et al., 2011, 2012a).

Other optimization algorithms exist such as Bat algorithm (Yang, 2010), and etc. Although these almost new algorithms have solved the problems arising from other optimization algorithms, but through implementing relatively more constraints and conditions, they have more complexity compared to them in implementation.

Although the Bat algorithm has been able to combine and use the advantages of other optimization algorithms with echolocation of the bat, but it has given much complexity to the algorithm and we should not neglect its high dependency on the initial conditions and constraints of the problem. In this paper, we have tried to cover all problems extracted from other optimization algorithms, while no additional constraint has been used in implementing the algorithm and flexibility of the algorithm is maintained. The observation of results is a good evidence of the aforesaid issue.

Conclusion

We have introduced a novel meta-heuristic algorithm for addressing the global optimization problems. The new algorithm is based on social behavior of blind naked mole-rats in finding food sources and protecting the colony against invasion. We have tested the proposed algorithm on several Benchmark functions and the obtained results compared with the other optimization algorithms have demonstrated the superior performance of the proposed algorithm.

REFERENCES

Akay B, Karaboga D (2010). A modified Artificial Bee Colony algorithm for real-parameter optimization. Inf. Sci. 192:20-142.

Bennett NC, Jarvis JUM (1988). The social structure and reproductive biology of colonies of the mole-rat. Cryptomys damarensis. J. Mammal. 69:293-302.

Brett RA (1991). The ecology of naked mole-rat colonies: Burrowing, food and limiting factors. In: The biology of the naked mole-rat. Princeton, NJ: Princeton University Press. pp. 137-184.

Chakraborty P, Das S, Roy GG, Abraham A (2010). On convergence of multi-objective particle swarm optimizers. Inf. Sci. 181:1411-1425.

Chang P-C, Huang W-H, Ting C-J (2010). Dynamic diversity control in genetic algorithm for mining unsearch solution space in TSP problems. Expert Syst. Appl. 37(3):1863-1878.

Clerc M, Kennedy J (2002). The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. 6(1):58-73.

Deep K, Bansal JC (2009). Mean particle swarm optimization for function optimization. Int. J. Comput. Intell. Stud. 1(1):72-92.

Goldberg D (1989). Genetic Algorithms in Search. Optimization and Machine Learning. Addison-Wesley. Reading. MA.

Holland JH (1975) Adaptive in Natural and Artificial Systems. University of Michigan Press, AnnArbor.

Kang F, Li J, Ma Z (2012a). An Artificial Bee Colony Algorithm for Locating the Critical Slip Surface in Slope Stability Analysis. Eng. Optim. DOI:10.1080/0305215X.2012.665451.

Kang F, Li J, Ma Z (2011). Rosenbrock Artificial Bee Colony Algorithm

for Accurate Global Optimization of Numerical Functions. Inf. Sci. 181(16):3508-3531.
Kang F, Li J, Xu Q (2012b). Damage Detection based on Improved Particle Swarm Optimization Using Vibration Data. Appl. Soft Comput. 12(8):2329-2335.
Karaboga D, Basturk B (2007). A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial bee colony (ABC) algorithm. J. Glob. Optim. 39(3):459-471.
Kirkpatrick S, Gelatt CD, Vecchi PM (1983). Optimization by simulated annealing. Sci. 220:671-680.
Oliveira Junior HAE, Ingber L, Petraglia A, Petraglia MR, Soares Machado MA (2012). Adaptive Simulated Annealing. Stochastic Global optimization and its Application with Fuzzy Adaptive Simulated Annealing. 35(2):33-62.
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, Nanyang Technological University, Singapore, 2005, <http://www.ntu.edu.sg/home/EPNSugan>.
Taherdangkoo M, Paziresh M, Bagheri M H, Yazdi M (2012b). An Efficient Algorithm for Function Optimization based on Modified Stem Cells Algorithm. Cent. Eur. J. Eng. Accepted for future publish.
Taherdangkoo M, Yazdi M, Bagheri MH (2011). Stem Cells Optimization Algorithm. Bio-Inspired Comput. Appl. 6840:394-403.
Taherdangkoo M, Yazdi M, Bagheri MH (2012a). A Powerful and Efficient Evolutionary Optimization Algorithm based on Stem Cells Algorithm for Data Clustering. Cent. Eur. J. Comput. Sci. 2(1):47-59.
Yang X-S (2010). A New Metaheuristic Bat-Inspired Algorithm. Nature Inspired Cooperative Strategies for Optimization. Stud. Comput. Intell. 284:65-74.