World Applied Sciences Journal 14 (10): 1507-1514, 2011
ISSN 1818-4952

Artificial Immune System Based Remainder Method for Multimodal Mathematical Function Optimization

[1]David F.W. Yap, [2]S.P. Koh and [2]S.K. Tiong

[1]Faculty of Electronic and Computer Engineering, Universiti Teknikal Malaysia Melaka (UteM),
Hang Tuah Jaya,76100 Durian Tunggal, Melaka, Malaysia
[2]College of Engineering, Universiti Tenaga Nasional (UNITEN),
Km 7, Kajang-Puchong Road, 43009 Kajang, Selangor Darul Ehsan, Malaysia

Abstract: Artificial immune system (AIS) is one of the nature-inspired algorithm for solving optimization problems. In AIS, clonal selection algorithm (CSA) is able to improve global searching ability compare to other meta-heuristic methods. However, the CSA rate of convergence and accuracy can be further improved as the hypermutation in CSA itself cannot always guarantee a better solution. Conversely, Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO) have been used efficiently in solving complex optimization problems, but they have an inclination to converge prematurely. In this work, the CSA is modified using the best solutions for each exposure (iteration) namely Single Best Remainder (SBR) - CSA. Simulation results show that the proposed algorithm is able to enhance the performance of the conventional CSA in terms of accuracy and stability for single objective functions.

Key words: Component · Clonal selection · Antigen · Affinity maturation · Antibody · Mutation

INTRODUCTION

Optimization is used in improving the performance of a process. Research on improving the optimization algorithm is ongoing and encompasses numerous field; from mathematics to engineering or even business and finance. For the past few decades, many methods have been developed, for example, GA, PSO, Ant Colony or Artificial Immune System (AIS). In this work, the improved CSA is evaluated in comparison to conventional CSA and other evolutionary algorithms such as PSO and GA. These algorithms are described in the following paragraphs.

PSO was originally proposed by Kennedy and Eberhart [1] as a simulation of social behaviour of organisms that live and interact within large groups. The essence of PSO is that particles move through the search space with velocities which are dynamically adjusted according to their historical behaviours. This mimics the social behaviour of animals such as bird flocking and fish schooling [2]. Each individual in a group imitates other groups that are better, in order to improve its own group. Genetic Algorithm (GA) is a search technique that has a representation of the problem states (represented using a set of chromosomes) and also has a set of operations to move through the search space. Each chromosome represents an individual solution (genes) to the problem. The set of individual solutions or genes forms a population. Genes in the population are improved across generation through a set of operation that GA uses during the search process. During each generation, the genes will go through the process of selection, crossover and mutation [3].

AIS is greatly reinforced by the immune system of a living organism such as human and animal. The clonal selection algorithm is a branch of AIS with principles extracted from clonal expansion and affinity maturation. The clonal selection theory explains that when an antigen (Ag) is detected, antibodies (Ab) that best recognize this Ag will proliferate by cloning. This immune response is specific to each Ag [4].

Corresponding Author: David F.W. Yap, Faculty of Electronic and Computer Engineering, Universiti Teknikal Malaysia Melaka (UteM), Hang Tuah Jaya,76100 Durian Tunggal, Melaka, Malaysia.
Tel: +6013 6100182, Fax: +606 555 2112.

Two algorithms based on CSA are proposed in this work. They are half best insertion (HBI) CSA and single best remainder (SBR) CSA. Similar to CSA, the ease of implementation is sustained in the proposed algorithms.

PSO, GA and AIS Algorithm

Particle Swarm Optimization: PSO algorithm starts with a group of random particles that searches for optima by updating each generation. Each particle is represented by a volume-less particle in the n-dimensional search space. The i_{th} particle is denoted as $X_i = (x_{i1}, x_{i2}, x_{i3}, \ldots x_{in})$. During generation updating, each particle is updated by ensuing two best values. These values are the best solution (*mbest*) and the global best value (*gbest*) that has been obtained by particles in the population at particular generation. With the inclusion and inertia factor ω, the velocity equations are shown in Eqs. (1) and (2).

$$v_{i+1} = v_i\omega + \alpha_1 \cdot rnd() \cdot (mbest_i - x_i) + \alpha_2 \cdot rnd() \cdot (gbest_i - x_i) \quad (1)$$

$$x_{i+1} = x_i + v_i \quad (2)$$

Where *rnd()* is a random number between 0 and 1, α_1 and α_2 are learning factors to control the knowledge and the neighbourhood of each individual respectively. The PSO algorithm is described in the following steps

- Generate initial random particle swarms assigned with its random position and velocity.
- Compute the fittest value of *N* particles according to fitness function.
- Update values of the best position of each particle and the swarm.
- Update the position and velocity for each particle according to equation 1 and 2.
- Repeat steps 3 and 4 until pre-defined stopping condition is achieved.

Genetic Algorithm: GA uses three main processes. They are selection, crossover and mutation to improve genes through each generation. The selection process uses the fitness function to evaluate the quality of the solution. Then, the best solutions from each generation are stored. Consequently, the function of crossover generates new genes given a set of selected members of the existing population. In the crossover process, genetic material between two single chromosome parents is exchanged. Then, mutation prompts sudden change in the

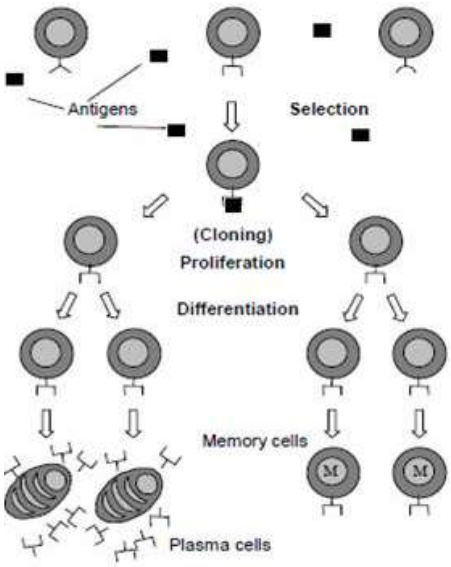


Fig. 1: Clonal Selection Principle (de Castro & Von Zuben, 2001a)

chromosomes unpredictably. However, the mutation process is important for the genes to avoid from trapping in local minima by adding random variables. The GA algorithm is described in the following steps

- Generate initial random population of individuals.
- Compute the fittest value of each individual in the current population.
- Select individuals for reproduction.
- Apply crossover and mutation operators.
- Compute the fittest value of each individual.
- Select the best individuals to generate new population.
- Repeat steps 3 to 6 until pre-defined stopping condition is achieved.

Artificial Immune System: The clonal selection theory explains t he detection of Ag by the Ab through cloning. As illustrated in Figure 1, the immune cells will reproduce against the Ags. The new cloned cells are then differentiated into plasma cells and memory cells [5]. The Abs are produced by plasma cells and go through mutation process to promote genetic variation. The memory cells are responsible for future Ags invasion. Lastly, the selection mechanism stores the Abs with the best affinity to the Ags in the next population [4]. The CSA pseudocode is described in the following steps.

- Generate an initial random population of antibodies, Abs.
- Compute the fittest value of each Ab according to fitness function.
- Generate clones by clonning all cells in the Ab population.
- Mutate the clone population to produce a mature clone population.
- Evaluate the affinitiy value for each clone population.
- Select the best Ab to compose the new Ab population.
- Repeat steps 3 to 6 until a pre-defined stopping condition is achieved.

Artificial Immune System and Particle Swarm Optimization Hybrid: AIS have the advantage to avoid the population from being trapped into local optimum. Moreover, PSO has the capability to improve itself but have a tendency to converge prematurely [6]. Therefore, the combination of AIS and PSO (AIS-PSO) is expected to improve the global search ability and prevent from being trapped in local minima even though the population size is relatively small [7]. The AIS-PSO pseudocode is described in the following steps.

- Select the best particles, N_1 (consisting of N/2 particles) from PSO.
- Generate a random initial population of Ab, N_2 (consisting of the other half, N/2 particles and regard every particle as an antibody).
- Combine N_1 with N_2 and compute the fitness of each Ab.
- Generate clones by cloning all cells in the Ab population.
- Mutate the clone population to produce a mature clone population.
- Evaluate affinity values of the clones' population.
- Select the best Ab to compose the new Ab population.
- Steps 4 to 7 are repeated until a pre-defined stopping condition is reached.

Half Best Insertion Artificial Immune System: Clonal selection of AIS adapts B-cells (and T-cells) to kill the invader through affinity maturation through hypermutation. The adaptation requires B-cells to be cloned many times [8, 9] and the hypermutation process cannot always guarantee that the next generation will provide better solution. The stochastic factor (randomization) at times can even produce worse result from previous solution. Thus, N number of the best Abs from the previous generation will be combined with the initial random Abs of the next generation to compose a new population for that next generation. This method known as Half Best Insertion (HBI) is expected to improve the convergence of the CSA algorithm. In HBI, half of the best antibodies from the previous generation are used in the next generation.

The N number of best Abs can be summarized as

$$\alpha = A_{best} / 2 \tag{3}$$

where α is the number of best Abs and A_{best} is the number of previous best Abs.

The best Abs selection, A_S, of sth antibodies is

$$A_S \begin{cases} 0 \\ A_{S+1}, \text{ otherwise} \end{cases} \tag{4}$$

Then, the new antibody population A_{new} is

$$A_{new} = A_i \cup A_S \tag{5}$$

The HBI algorithm is described in the following steps.

- Generate an initial random population of antibodies, A_i.
- Compute the fittest value of each Ab according to fitness function.
- Generate clones by cloning all cells in the Ab population.
- Mutate the clone population to produce a mature clone population.
- Evaluate the affinity value for each clone in the population and select N number of best Abs, α.
- Generate next generation of initial random Abs and include A_s.
- Repeat steps 1 to 6 until pre-defined stopping condition is achieved.

Single Best Remainder Artificial Immune System: Hypermutation of good Abs in HBI algorithm would tend to produce more bad solution due to stochastic factor. Therefore, the Single Best Remainder (SBR) algorithm tries to avoid hypermutation process. The best Abs from previous generation is kept in global memory as single

best antibody which is not affected by the next affinity maturation and hypermutation processes. The global single best antibody will be updated through each generation and used in the next generation if the hypermutation result converges prematurely in the search space. Hence, SBR is proposed in order to improve the convergence and accuracy of the CSA algorithm.

In SBR, the best antibody obtained for the clonal selection process is recorded as global solution, A_m. During each generation process, the randomize antibodies, A_r, is replaced by the best solution. The clone cell result after maturation, F_m, is evaluated based on the test function. Then, F_m is compared with the result of randomize antibodies (A_r) after the test function based evaluation, F_m. If F_m is larger or equal to F_t, the clone cell, C_{bp}, is replaced by A_r. Otherwise, the C_{bp} is maintained.

$$C_{bp} = A_r, \text{ if } F_m = F_t \tag{6}$$

where F_t = testFunction (A_r) and F_m = testFunction (C_{bp})

The SBR algorithm is described in the following steps.

- Generate an initial random population of Abs.
- Compute the fittest value of each Ab according to fitness function.
- Generate clones by cloning all cells in the Ab population.
- Mutate the clone population to produce a mature clone population.
- Evaluate the affinity value for each clone in the population.
- Select the best Ab, A_m, in 5 as global memory and repeat steps 1 to 5.
- Repeat steps 1 to 5 and compare the best Ab obtained with A_m.
- The best Ab from 7 is updated as the global memory, A_m.
- Repeat steps 1 to 8 until pre-defined stopping condition is achieved.

All methods described above are evaluated using nine mathematical test functions. The termination criteria for all methods will be met if minimum error value is achieved or maximum number of evaluation allowed is exceeded.

Experiments on Test Function: The computing platform used for the this simulation is AMD Phenom 9600B Quad-Core CPU running at 2.30 GHz, 2GB of RAM and Windows Vista Enterprise operating system. Each algorithm is evaluated based on 500 iterations, 10 dimensions and the mean of best fitness is obtained after 30 runs (with the average value taken). The minimum error is set as 1.00E-25, while the population size P_0 is set to 20. The simulation was carried out up to 500 iterations, because beyond this value, the change of the fitness value is to the power of -6 and thus it is pointless to simulate further.

In the HBI, antibodies and memory size of 50% are maintained. At first iteration, CSA is used to obtain the first solution. Then, for the next iteration, half of the population is composed of the half best antibodies after hypermutation and the other half is given by randomized Abs. The new population then goes through the affinity maturation process similar to CSA.

Subsequently, in SBR, similar to HBI, CSA is used to obtain the first solution. Then, for the next iteration, the best antibody, A_m, is kept as global memory. This A_m will never go through affinity maturation process, but will be assigned as a reference (memory) in case the hypermutation process produces worse solution.

The nine benchmark functions (objective functions) are described as follows.

Rastrigin's Function: Rastrigin's function is mathematically defined as follows.

$$f_1(x) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i) + 10 \right) \tag{7}$$

where $-5.12 \leq x_i \leq 5.12$, $i = 1. . ., n$ and global minimum is located at the origin and its function value is zero.

De Jong's Function: De Jong's function is mathematically defined as follows.

$$f_2(x) = \sum_{i=1}^{n} x_i^2 \tag{8}$$

where $-5.12 \leq x_i \leq 5.12$, $i = 1. . ., n$ and global minimum is located at the origin and its function value is zero.

Axis Parallel Hyper-ellipsoid Function: Axis parallel hyper-ellipsoid function is mathematically defined as follows.

$$f_3(x) = \sum_{i=1}^{n} i \cdot x_i^2 \tag{9}$$

where $-5.12 \leq x_i \leq 5.12$, $i = 1. . ., n$ and global minimum is located at the origin and its function value is zero.

Rosenbrock's Function: Rosenbrock's function is mathematically defined as follows.

$$f_4(x) = \sum_{i=1}^{n-1} 100 \cdot \left(x_{i+1} - x_i^2\right)^2 + \left(1 - x_i\right)^2 \quad (10)$$

where-2.048≤ x_i ≤ 2.048, I = 1. . ., n and global minimum is located at the origin and its function value is zero.

Sum of Different Power Function: Sum of different power function is mathematically defined as follows.

$$f_5(x) = \sum_{i=1}^{n} \left|x_i\right|^{(i+1)} \quad (11)$$

where -1≤ x_i ≤ 1, i = 1. . ., n and global minimum is located at the origin and its function value is zero.

Rotated Hyper-Ellipsoid Function: Rotated hyper-ellipsoid function is mathematically defined as follows.

$$f_6(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \quad (12)$$

where -65.536≤ x_i ≤ 65.536, i = 1. . ., n
and global minimum is located at the origin and its function value is zero.

Moved Axis Parallel Hyper-ellipsoid Function: Moved axis parallel hyper-ellipsoid function is mathematically defined as follows.

$$f_7(x) = \sum_{i=1}^{n} 5i \square x_i^2 \quad (13)$$

where -5.12≤ x_i ≤ 5.12, i = 1. . ., n and global minimum is located at the origin and its function value is zero.

Griewangk Function: Griewangk's function is mathematically defined as follows.

$$f_8(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \quad (14)$$

where - 600≤ x_i ≤ 600, i = 1. . ., n and global minimum is located at the origin and its function value is zero.

Ackley Function: Ackley's function is mathematically defined as follows.

$$f_9(x) = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)} \quad (15)$$

where - 32.768≤ x_i ≤ 32.768, i = 1. . ., n and global minimum is located at the origin and its function value is zero

RESULT AND DISCUSSION

The results for each of the test functions are shown in Figures 2 to 10 and Table 1. For Rastrigin's function in Figure 2, PSO suffers from premature convergence while PSO-AIS is less accurate in giving the fitness value. Alternatively, SBR gives the best fitness value followed by HBI, CSA and GA.

The GA fitness value is very close to CSA for Dejong's function as shown in Figure 3. The PSO converges rapidly up to 50 generations but perform no significant improvement beyond this which is also similar to Rastrigin's function. In contrast, GA converges very slow but seems to be able to perform even after 500 generations since there is no breaking point after that. The SBR achieved the best performance and is comparable to PSO-AIS and CSA.

In Figure 4, the Axis Parallel Hyper-ellipsoid function shows that the CSA achieved the best fitness value which is comparable to SBR and PSO-AIS. There is no significant improvement in PSO after 50 generations.

Figure 5 shows that PSO and GA perform badly among all algorithms. In contrast to Rastrigin's, Dejong's and Axis Parallel Hyper Ellipsoid, the PSO-AIS outperformed other algorithms. However, the fitness value of CSA and SBR are comparable.

Figure 6 shows similarities to Figure 3 and Figure 4. The fitness value of CSA, SBR and PSO-AIS are comparable. Alternatively, PSO shows the worst performance with a bad fitness value.

In Figure 7, the Rotated Hyper-ellipsoid function is similar in result to the Sum of Differential Power function. However, SBR is slightly better in performance than CSA and PSO-AIS.

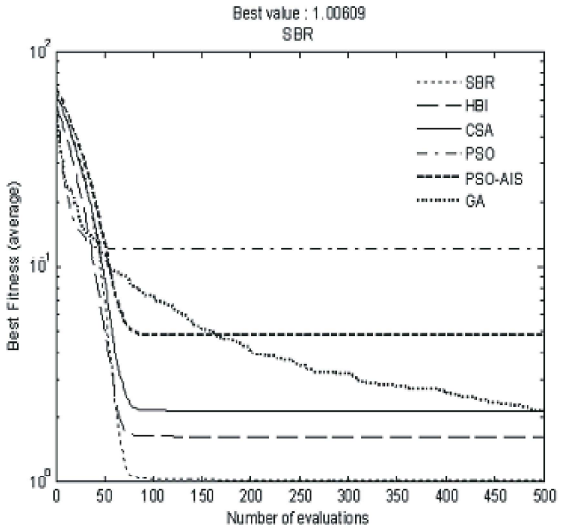
In Figure 8, the Moved Axis Parallel Hyper-ellipsoid shows similarities to Rotated Hyper-ellipsoid Function. The performance of SBR is also slightly better than CSA, followed by PSO-AIS.

The results of Griewangk's function in Figure 9 show similarities to the Moved Axis Parallel Hyper-ellipsoid result. Here, the CSA is slightly better than SBR and PSO-AIS.

The Ackley's function in Figure 10 shows that the GA performance outperformed other algorithms followed by CSA and HBI. The SBR seems to suffer from premature convergence. In contrast to previous results, the SBR is worse than HBI. However, the PSO-AIS have the worse

Table 1: Mean and Standard Deviation for each of the algorithm based on the given transfer function

	SBR		HBI		CSA		PSO		PSO-AIS		GA	
Function	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Eq. 7	1.01E+00	1.16E+00	1.62E+00	8.37E-01	2.11E+00	1.52E+00	1.21E+01	5.62E+00	4.89E+00	1.09E+00	2.12E+00	1.19E+00
Eq. 8	5.60E-07	3.30E-07	1.40E-04	3.70E-05	6.10E-07	4.80E-07	1.16E-01	7.71E-02	8.20E-07	6.30E-07	1.90E-04	1.40E-04
Eq. 9	7.60E-07	7.20E-07	1.60E-04	3.00E-05	5.90E-07	3.90E-07	1.27E-01	8.11E-02	6.90E-07	4.60E-07	1.50E-04	7.10E-05
Eq. 10	4.21E+00	7.20E-01	4.59E+00	1.86E+00	4.49E+00	1.39E+00	1.14E+01	3.76E+00	2.29E+00	1.30E+00	6.06E+00	2.19E+00
Eq. 11	5.60E-07	4.50E-07	1.30E-04	2.70E-05	4.60E-07	1.50E-07	1.36E-01	8.61E-02	7.40E-07	5.40E-07	1.70E-04	6.60E-05
Eq. 12	1.60E-05	6.30E-06	3.33E-03	6.20E-04	1.80E-05	1.30E-05	1.72E+00	2.47E+00	3.00E-05	3.30E-05	2.93E-03	1.65E-03
Eq. 13	1.00E-04	1.20E-04	3.55E-03	1.18E-03	1.10E-04	1.10E-04	2.00E+00	1.09E+00	1.70E-04	3.30E-04	2.39E-03	7.50E-04
Eq. 14	2.20E-07	2.40E-07	2.00E-05	7.00E-06	2.00E-07	1.30E-07	1.94E-02	2.26E-02	2.60E-07	2.10E-07	2.20E-05	1.60E-05
Eq. 15	4.01E-01	6.47E-01	3.57E-01	5.51E-01	1.23E-01	3.63E-01	1.33E+00	8.35E-01	1.47E+00	8.55E-01	1.73E-02	3.04E-03



Fig, 2: Algorithms evaluation on Rastrigin's function

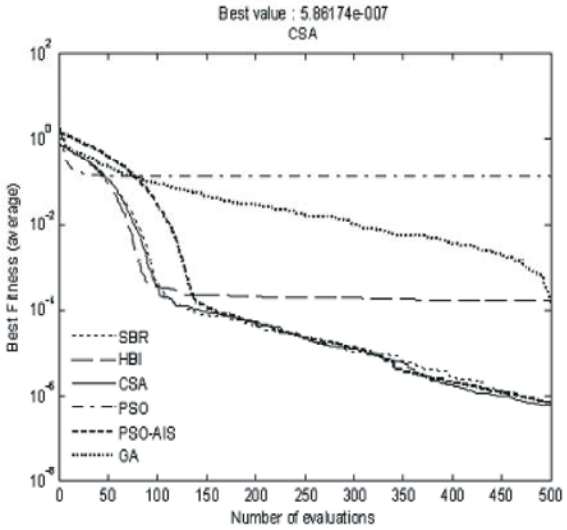


Fig. 4: Algorithms evaluation on Axis Parallel Hyper-ellipsoid function

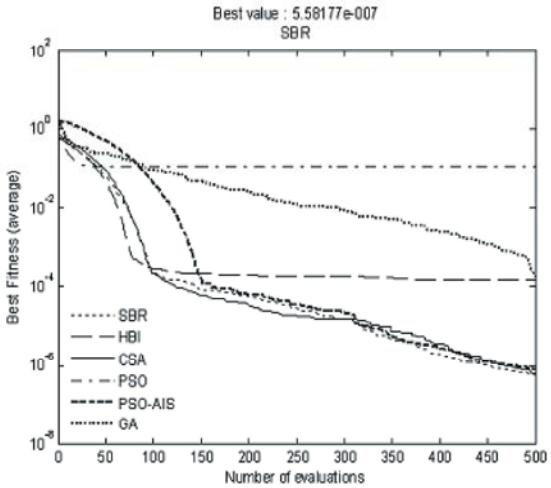


Fig. 3: Algorithms evaluation on Dejong's function

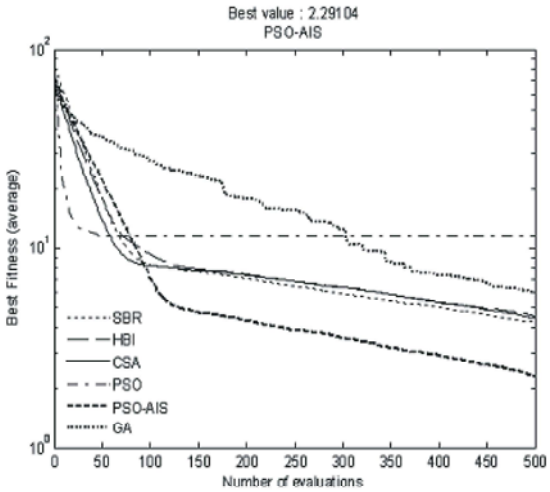


Fig. 5: Algorithms evaluation on Rosenbruck function

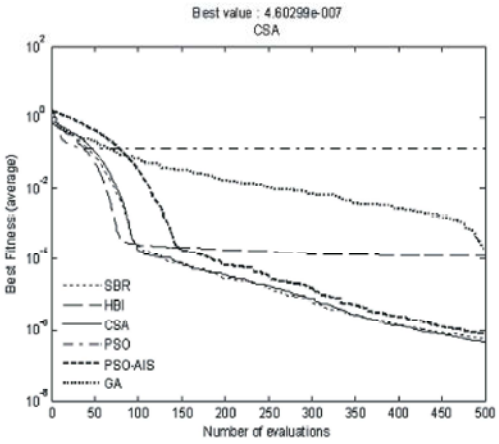


Fig. 6: Algorithms evaluation on Sum Differential Power function

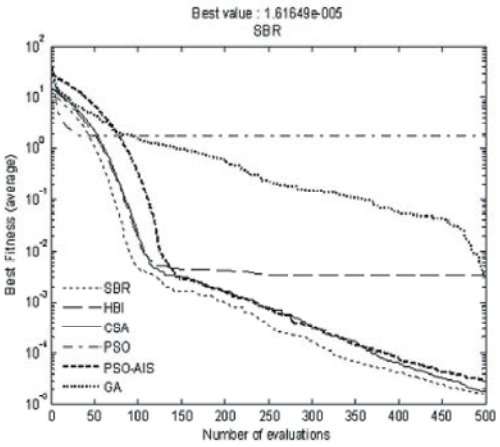


Fig. 7: Algorithms evaluation on Rotated Hyper-ellipsoid function

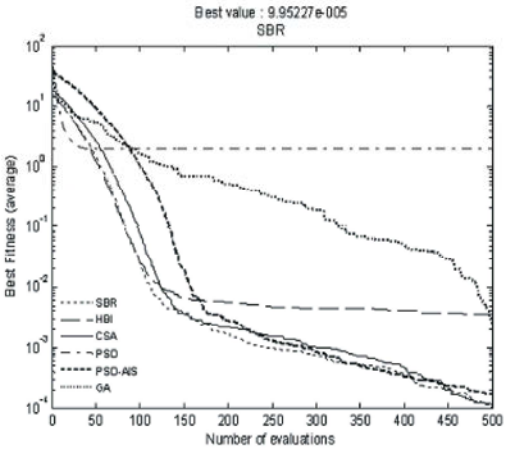


Fig. 8: Algorithms evaluation on Moved Axis Parallel Hyper-ellipsoid function\

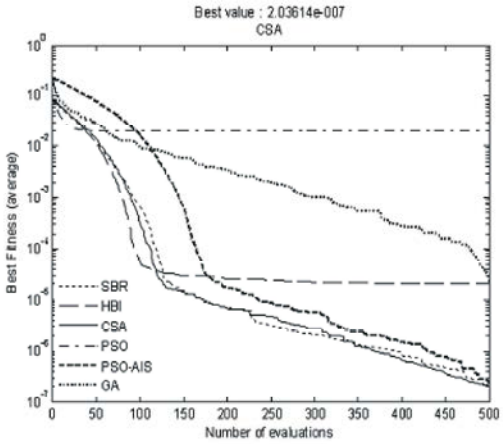


Fig. 9: Algorithms evaluation on Griewangk function

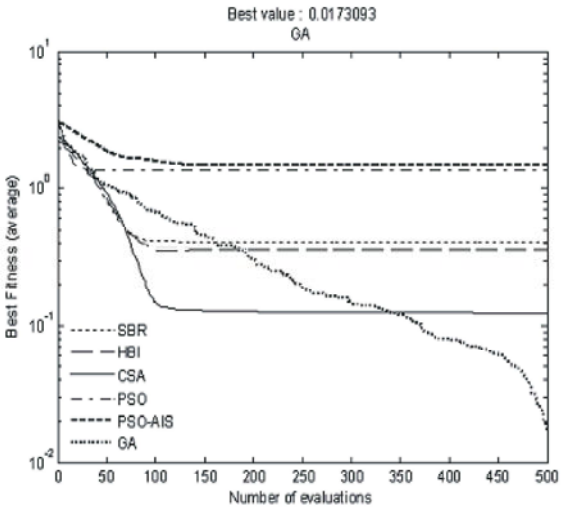


Fig. 10: Algorithms evaluation on Ackley function

performance and this is followed by PSO. Both of the algorithms have no significant improvement after 100 generations.

The mean and standard deviation value for nine test functions used to evaluate the algorithms performance is presented in Figure 8. SBR method outperformed other algorithms for test function number 1, 2, 6 and 7. The CSA is the best for test function number 6, 8 and 11. The PSO-AIS achieved the best test function for number 7 while GA performed the best for test function number 9.

The stable algorithms are given by SBR, CSA and PSO-AIS, most probably due to the very small standard deviation value in between 1e-7 and 1e-6 for most of the test functions. The PSO performed badly since the standard deviation value is large.

CONCLUSION

Two memory-based clonal selection AIS strategy using the local memory was proposed in this paper. They are known as SBR and HBI. Though PSO is fast in obtaining the fitness value, it suffers from premature convergence. On the other hand, GA converges slowly to achieve the best fitness value. The simulation work clearly showed that the best result is given by SBR. However, more work would be carried out in tweaking certain parameters in SBR such as the memory allocation factor, best memory selection criteria or the number of best memory to be taken into consideration, in order to improve the performance of the algorithm.

ACKNOWLEDGMENT

The authors would like to thank The Ministry of Higher Education Malaysia for the financial support under Fundamental Research Grant Scheme No. FRGS/2010/FKEKK/TK03/1 – F0085.

REFERENCES

1. Kennedy, J. and R. Eberhart, 1995. Particle Swarm Optimization. In the Proc. IEEE Int. Conf. Neural Networks, pp: 1942-1948.
2. Kennedy, J. and R. Eberhart, 1995. Swarm Intelligence, 1st Edition. Academic Press.
3. Emmeche, C., 1994. Garden in the Machine: The Emerging Science of Artificial Life. Princeton University Press.
4. De Castro, L.N. and J. Timmis, 2002. Artificial Immune System: A New Computational Approach. Springer-Verlag.
5. Acan, A. and A. Unveren, 2005. A Memory-Based Colonization Scheme for Particle Swarm Optimization. In the IEEE Congress on Evolutionary Computation, pp: 1965-1972.
6. Vesterstrom, J. and R. Thomsen, 2004. A Comparative Study of Differential Evolution, Particle Swarm Optimization and Evolutionary Algorithms on Numerical Benchmark Problems. In the Proceedings of Congress on Evolutionary Computation, pp: 1980-1987.
7. Yap, D.F., W.S.P. Koh, S.K. Tiong and S.K. Prajindra, 2011. Particle Swarm based Artificial Immune System for Multimodal Function Optimization and Engineering Application Problem, Trends in Applied Sciences Res., 6: 282-293.
8. Garret, S.M., 2005. How Do We Evaluate Artificial Immune System. Evoluationary Computation, 13: 145-178.
9. Burnet, F.M., 1969. The Clonal Selection Theory of Acquired Immunity. Cambridge University Press.