http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 10 (8): 1536-1544, 2011
ISSN 1812-5638 / DOI: 10.3923/itj.2011.1536.1544

A New Particle Swarm Optimization with a Dynamic Inertia Weight for Solving Constrained Optimization Problems

Hui Lu and Xiao Chen
School of Electronic and Information Engineering, Beijing University of Aeronautics and Astronautics, Beijing 100191, China

Abstract: This study has presented an enhanced particle swarm optimization approach which is designed to solve constrained optimization problems. The approach incorporates a dynamic inertia weight in order to help the algorithm to find the global and overcome the problem of premature convergence to local optima. The inertia weight of every individual is dynamically controlled by the Euclidean distance between individual and global best individual. The approach was tested with a well-known benchmark. Simulation results show that the suitability of the proposed algorithm in terms of effectiveness and robustness.

Key words: Constrained optimization, particle swarm optimization, inertia weight, euclidean distance, evolutionary algorithms

INTRODUCTION

In the last few years, using evolutionary algorithms to solve constrained optimization problems has attracted a lot of researches, since such problems are very common in real-world applications. Particle Swarm Optimization (PSO) (Eberhart and Kennedy, 1995; Kennedy and Eberhart, 1995; Kennedy *et al*., 2001; Hu *et al*., 2003b) is one of the evolutionary algorithms that have been adopted to solve such problems. PSO has been widely used to solve several types of optimization problems. Kennedy and Spears (1998) proposed a matching algorithm for the multimodal problem generator. Hu *et al*. (2003a) used PSO in n-queens problem and Cagnina *et al*. (2004) focus on sequencing problems. Hooshmand (2008) proposed an optimal design using PSO in power system. Shakiba *et al*. (2008) researched short-time prediction of traffic rate. Wang *et al*. (2008) researched structure learning of product unit neural networks. Ren *et al*. (2010) presented a new global optimization algorithm for mixed-integer-discrete-continuous variables based on PSO. Hernane *et al*. (2010a, b) focused on scheduling problem. Nevertheless, they are unconstrained search technique or lack an explicit mechanism to bias the search in constrained search spaces (Parsopoulos and Vrahatis, 2002; Coath and Halgamuge, 2003; Michalewicz and Schoenauer, 1996). This has motivated the development of a considerable number of approaches to incorporate constraints into the fitness function of PSO (for example, Runarsson and Yao (2000), Mezura-Montes and Coello (2005), Liang and Suganthan (2006), Takahama and Sakai (2006), Cagnina *et al*. (2007) and Zhao *et al*. (2008).

PSO was inspired by the movement of a group of animals such as a bird flock or fish school. PSO explores the search space using a population of individuals and the best performers (either within a group or with respect to the entire population) affect the performance of the others. Each individual is named particle and represents a possible solution within a multidimensional search space. The particles have their own position and velocity which are constantly updated. They record their past behavior and use it to move towards promising regions of the search space (Kennedy and Eberhart, 1999).

A new PSO algorithm is presented in this study which is designed to solve constrained optimization problems. For that sake, our approach contains a constraint-handling technique as well as a mechanism to update the velocity and position of particle which is extended by adding to it a dynamic inertia weight as a way to avoid premature convergence.

STATEMENT OF THE PROBLEM

Without loss of generality, this study consider the general nonlinear programming problem (Mezura-Montes and Coello, 2005) in which the destination is:

$$\text{Find } x \text{ which optimizes } f(x) \quad (1)$$

Subject to:

Corresponding Author: Hui Lu, 204 Lab, Beijing University of Aeronautics and Astronautics, Beijing, 100191, China
Tel: +8601082316487 Fax: +8601082316486

$$g_i(x) \leq 0,\ i = 1,\ldots, p \quad (2)$$

$$h_j(x) = 0,\ j = 1,\ldots, p \quad (3)$$

Where x is the vector of solutions $x = [x_1, x_2,\ldots, x_D]$, n is the number of inequality constraints and p is the number of equality constraints (in both cases, constraints could be linear or nonlinear). All satisfying all inequality and equality constraint functions determine the feasible solution.

If F denotes the feasible region and with S to the whole search space which is a D-dimensional rectangle defined by the lower and upper bounds of each variable x_d, then it should be clear that $F \subseteq S$.

PREVIOUS RELATED WORK

There exist many studies on solving unconstrained optimization problems using particle swarm optimization. However, the proposals of constraint-handling mechanisms for PSO are relatively scarce. Next, this study will review the most representative work down in this area.

Takahama and Sakai (2006) presented a PSO algorithm that combines the ε constrained method for handing constrains. The ε constrained method is an algorithm transformation method which can convert algorithms for unconstrained problems to algorithms for constrained problems using the ε level comparison that compares the search points based on the constraint violation of them. In the ε PSO, the agents who satisfy the constraints move to optimize the objective function and the agents who don't satisfy the constraints move to satisfy the constraints. But sometimes the velocity of agents becomes too big and they fly away from feasible region.

Mezura-Montes and Coello (2005) added a simple multimembered evolution strategy to solve global nonlinear optimization problems which uses a simple diversity mechanism based on allowing infeasible solutions to remain in the population. The authors concluded that their results obtained are very competitive when comparing the proposed approach against other state-of-the art techniques and its computational cost is lower than the cost required by the other techniques compared.

Liang and Suganthan (2006) proposed a novel constraint-handling mechanism based on multi-swarms. The whole population is divided into a large number sub-swarms, these sub-swarms are regrouped frequently by using various regrouping schedules and information is exchanged among the particles in the whole swarm. Through combining the exploitation and exploration together, this neighborhood structure gives better performance on complex problems. The authors reported that the proposed algorithm can find reasonable solutions for all of the problems taken from Mezura-Montes and Coello (2005).

It was observed that the standard PSO presents difficulty in finding a fine-tuning of the solution based on the previous study on particle swarm optimization. In order to improve the performance of the standard PSO, a dynamic way to change inertia weight that can control the exploitation and exploration ability of PSO was proposed and some improvement of performance has been obtained. The inertia weight not only changes with the iteration time but also has a fine-tuning to the best position of PSO. The Dynamic Inertia Weight Particle Swarm Optimization (DIW-PSO) algorithm adds some feature of the algorithms cited above: the constraint-handing mechanism, the variant tolerance factor and the velocity update formula.

DYNAMIC INERTIA WEIGHT PARTICLE SWARM OPTIMIZATION ALGORITHM

Here, we describe in detail our proposed approach which is called Dynamic Inertia Weight Particle Swarm Optimization algorithm (DIW-PSO).

General model: As in the standard model, A PSO algorithm operates on a population of particles. The particles are D-dimensional real number vectors which are due to the type of problem to optimize (with D decision variables). The particles evolve using two update formulas, one for position and another one for velocity. The best position found so far for the particles (for the gbest model) or in the neighborhood (for the lbest model) is recorded. The best value reached by each particle (pbest) is stored, too.

Particular model: As it was stated in some study (Cagnina *et al*., 2004; Liang and Suganthan, 2006), the *gbest* model tends to converge to a local optimum. Motivated by this, some papers proposed a formula to update the velocity and position, using a combination of both the *gbest* and the *lbest* models (Cagnina *et al*., 2006, 2007). Such a formula is adopted here as well and is shown in Eq. 4 and 5.

$$v_{id} = \omega * v_{id} + c_1 * rand1 * (p_{id} - par_{id}) + c_2 * rand2 * (p_{id} - par_{id}) + c_3 * rand3 * (p_{gd} - par_{id}) \quad (4)$$

$$par_{id} = par_{id} + v_{id} \quad (5)$$

where, v_{id} is the velocity of the particle i at the dimension d; ω is the inertia weight whose goal is to balance global exploration and exploitation (Shi and

Eberhart, 1998). c_1 is the personal learning factor and c_2, c_3 are the social learning factors. These 3 values multiply to 3 different random numbers within the range [0..1]. p_{id} is the best position reached by the particle i, p_{id} is the best position reached by any particle in the neighborhood, p_{gd} is the best position reached by any particle in the swarm and par_{id} is the value of the particle i at the dimension d (Cagnina *et al*., 2007).

This study uses a wheel topology (Kennedy, 1999) to compute the p_{id} value, in which one individual is connected to all others and they are connected to only that one. The Wheel topology effectively isolates individuals from one another, as all information has to be communicated through the focal individual. This focal individual compares performances of all individuals in the neighborhood and adjusts its trajectory toward the very best. If adjustments result in improvement in the focal individual's performance, then that improvement is communicated out to the rest of the population. Thus the focal individual serves as a kind of buffer or filter, slowing the speed of transmission of good solutions through the population (Kennedy, 1999). This concept illustrated in Fig. 1.

Handing constraints: The typical constraint-handling scheme is Lagrange-based method (Du and Pardalos, 1995). The Lagrange-based method is a classical approach to formulate constrained optimization problems. By introducing the Lagrangian formulation, the primal problem (Eq. 1) can be written as:

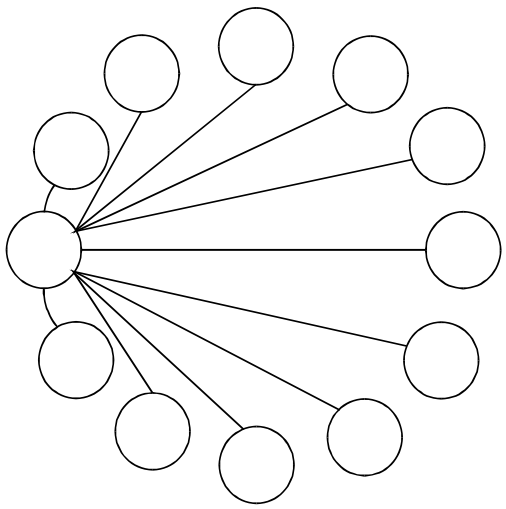

Fig. 1: Wheel topology illustration with a population of size 12

$$\min_{\mu,?} L(x,\mu,?) \tag{6}$$

Subject to:

$$\mu_i \geq 0,\ i = 1,..,\ n \tag{7}$$

$$\lambda_j \geq 0,\ j = 1,..,\ p \tag{8}$$

Where:

$$L(x,\ \mu,\ \lambda) = f(x) + \mu^T g(x) + \lambda^T h(x) \tag{9}$$

where, μ is a $n\times1$ multiplier vector for the inequality constraints. λ is a $p\times1$ multiplier vector for the equality constraints. If the problem Eq. 1 satisfies the convexity conditions over S, then the solution of the primal problem Eq. 1 is the vector x^* of the saddle point $\{x^*, \mu^*, \lambda^*\}$ of $L(x^*, \mu^*, \lambda^*)$ so that:

$$L(x^*, \mu^*, \lambda^*) \leq L(x^*, \mu^*, \lambda^*) \leq L(x^*, \mu^*, \lambda^*)$$

The saddle point can be obtained by minimizing $L(x^*, \mu^*, \lambda^*)$ with the optimal Lagrange multipliers (μ^*, λ^*) as a fixed vector of parameter. In general, the optimal values of the Lagrange multipliers are unknown a priori (Krohling and Coelho, 2006).

Dynamic inertia weight: Inertia weight is a very important parameter in standard PSO algorithm which can be used to control the exploitation and exploration ability of algorithm. Its value determines how much it succeeds current rate: the bigger the inertia weight of algorithm is, the greater the speed of particle gets and thus particle has stronger exploration ability; the smaller the inertia weight is, the stronger the exploitation ability of particle is (Tian *et al*., 2008). At present, in the study of modified PSO algorithms with inertia weight, inertia weight is usually divided into two types of static and dynamic. Since, in the process of evolution, PSO algorithm with static inertia weight always maintains a constant value, making the exploitation and exploration ability of algorithm not reach balance, thus algorithm falls into local optimization easily and, in the latter of evolution, the convergence rate greatly decreasing makes algorithm cannot converge at global optimal solution. Therefore, in order to overcome these problems, it is particularly important to research dynamic inertia weight.

This study proposes a new dynamic way of inertia weight change. During the iteration, the search space

around p_g have a high probability of global optimal solution, so a high intensity search around p_g reached by any particle in the swarm should be consider. Shi and Eberhart (1998) used a way of linear decrease to adjust inertia weight and reference (Cui and Zhu, 2007) found that with inertia weight decreasing, algorithm early had a good global search capability and late had a better convergence but the rate was relatively slow. The algorithm hopes have a good local search in the search space around p_g and particles out of the search space around p_g have a good global search capability. In that way, Dynamic inertia weight can control the exploitation and exploration ability in a good manner.

d_{ig} denotes the Euclidean distance of par_i and p_g. Inertia weight ω was the function of iteration time k and satisfied the equation:

$$\omega(i,k) = \omega_{min} + \frac{d_{ig}}{d_{max}} \times \frac{\omega_{max} - \omega_{min}}{MaxMumber} \times (MaxMumber - k) \quad (10)$$

where, d_{max} is the maximum Euclidean distance between any particle and particle p_g, MaxMumber was the maximal iteration time and the value of inertia weight dynamic decreased from ω_{max} to ω_{min}.

DIW-PSO algorithm pseudo code: Figure 2 shows the pseudo code of proposed DIW-PSO algorithm. At beginning of the algorithm, DIW-PSO randomly initialize the position and the velocity of each particle. After evaluating the particles and obtaining the best values: pbest, lbest, gbest and Euclidean distance d_{ig}, the pop begin to evolve. During the evolutionary process, the position and the velocity of each particle are updated. In the evolutionary loop, a new dynamic inertia weight mechanism is applied to improve the global search capability and convergence of PSO algorithm. Finally, the result (gbest) is taken and returned.

```
Diw-PSO:
Swarm intialization
  Intializate pop
  Intialize velocity for pop
  Evaluate fitness for pop
  Record pbest and gbest for pop
  Evaluate dig
Swarm flights through the search space
  DO
      FOR i = 1 TO number of particles Do
        Search the best leader in neighborhood
        of pari and record in lbesti
        Evaluate ω (i, k) with Eq. 10
      FOR j = 1 TO number of dimensions DO
        Update vij with Eq. 4
        Update parij with Eq. 5
      END
    END
    Evaluate dig
    Evaluate fitness (pop)
      Record pbest and gbest
  WHILE (k<MaxMumber)
  result = gbest
RETURN (result)
```

Fig. 2: Pseudo code of DIW-PSO

SIMULATION RESULTS AND STATISTICAL ANALYSIS

To evaluate the performance of the proposed approach, this paper used the 13 test functions described in Runarsson and Yao (2000). The test functions chosen contain characteristics that are representative of what can be considered "difficult" global optimization problems for an evolutionary algorithm. The detailed description of the test problems may be consulted in its original source (Runarsson and Yao, 2000).

The independent runs per problem performed 30 times. The maximum numbers of generations was set to 500. For the optimization problems studied, the population size was set to 50. The maximum inertia factor $\omega_{max} = 1.2$, the minimum inertia factor $\omega_{min} = 0.4$, personal learning factor and social learning factors for c_1, c_2 and c_3 was set to 2.05. The parameter settings such as personal learning factor, swarm size, social learning factors and maximum and minimum inertia factors were empirically derived after numerous experiments.

The simulation results using the standard PSO and DIW-PSO are shown in Table 1. Our approach found better best solution in eight problems (G2, G3, G4, G5, G6, G9, G10, G13) and a same best result in other four (G01, G7, G8, G11). Also, our approach reached better mean and worst result in all problems. A worse best result was found by DIW-PSO than PSO in problem G12.

For problem G01, the optimal solution is -15. The convergence curves of best particle using DIW-PSO and PSO are shown in Fig. 3. From Table 1, it can be seen that DIW-PSO and PSO find the optimal solution, however, the median and mean of fitness found by DIW-PSO is much better than the results obtained by PSO. Using DIW-PSO, the optimal solution is found in approximately 150 generations, while PSO finds optimal solution in approximately 300 generations. From the convergence curves of best particle, it can be seen that the solution found by DIW-PSO goes down fast than by PSO which demonstrates the robustness of the DIW-PSO algorithm.

For problem G04, G05, G06, G07, G09 and G10, the convergence curves of best particle using DIW-PSO and PSO resembles as for G01 which is shown in Fig. 4-10 in detail.

Table 1: Results Using Pso And Diw-Pso

Problem	Method	Best	Median	Mean	Worst	Std. Dev.
G01	PSO	-15.0	-14.337	-14.225	-13.725	0.145
Optimal (-15.000)	DIW-PSO	-15.0	-14.996	-14.997	-14.995	0.0014
G02	PSO	0.803623	0.803785	0.803944	0.814864	0.0078
Optimal (0.803619)	DIW-PSO	0.803621	0.803634	0.803645	0.803726	0.0006
G03	PSO	-0.9986	-0.983	-0.985	-0.921	0.053
Optimal (-1.000)	DIW-PSO	-1.0	-0.991	-0.986	-0.954	0.0012
G04	PSO	-30665.489	-30665.201	-30665.198	-30664.426	0.135
Optimal (-30665.539)	DIW-PSO	-30665.539	-30665.539	-30665.539	-30665.539	0
G05	PSO	5126.507	5126.748	5126.751	5127.035	0.345
Optimal (5126.498)	DIW-PSO	5126.502	5126.610	5126.615	5126.675	0.173
G06	PSO	-6961.810	-6961.741	-6961.736	-6960.915	0.785
Optimal (-6961.814)	DIW-PSO	-6961.814	-6961.756	-6961.750	-6961.201	0.326
G07	PSO	24.306	24.365	24.370	24.798	0.1042
Optimal (24.306)	DIW-PSO	24.306	24.308	24.340	24.518	0.0374
G08	PSO	0.095825	0.095721	0.095702	0.093124	0.037
Optimal (0.095825)	DIW-PSO	0.095825	0.095825	0.095825	0.095825	0
G09	PSO	680.647	680.659	680.658	680.845	0.0798
Optimal (680.63)	DIW-PSO	680.630	680.643	680.642	680.719	0.0155
G10	PSO	7049.29	7049.35	7049.38	7049.89	0.0144
Optimal (7049.25)	DIW-PSO	7049.25	7049.31	7049.35	7049.57	0.0056
G11	PSO	0.75	0.79	0.83	0.97	0.1054
Optimal (0.75)	DIW-PSO	0.75	0.76	0.77	0.83	0.0582
G12	PSO	1.000	0.982	0.985	0.967	0.0126
Optimal (1.000)	DIW-PSO	0.998	0.976	0.979	0.945	0.0233
G13	PSO	0.05408	0.05523	0.05571	0.05912	0.0145
Optimal (0.05395)	DIW-PSO	0.05395	0.05401	0.05403	0.05531	0.0048

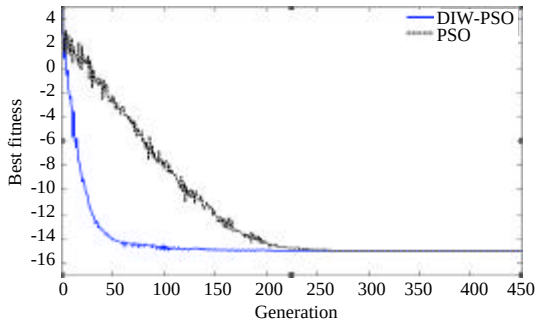

Fig. 3: Fitness of the best particle for problem G01

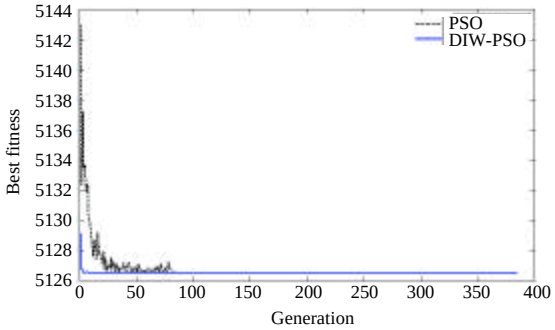

Fig. 5: Fitness of the best particle for problem G05

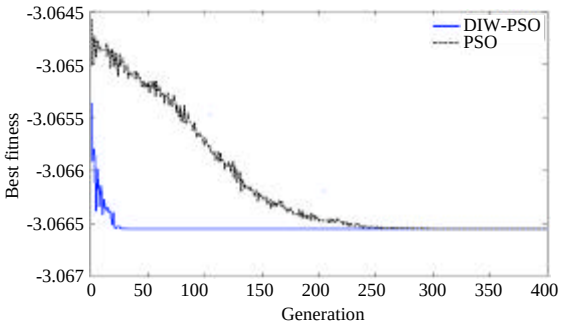

Fig. 4: Fitness of the best particle for problem G04

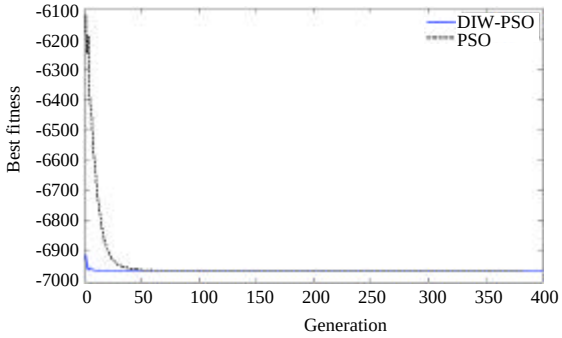

Fig. 6: Fitness of the best particle for problem G06

For problem G02, G11 and G13, the convergence curves of best fitness using DIW-PSO and PSO are shown in Fig. 10-12, respectively. From Table 1, it can be observed that DIW-PSO finds a solution very close to the

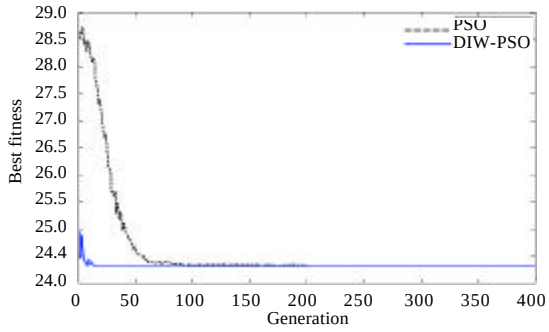

Fig. 7: Fitness of the best particle for problem G07

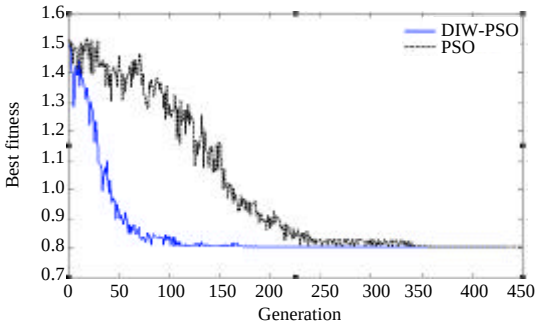

Fig. 10: Fitness of the best particle for problem G02

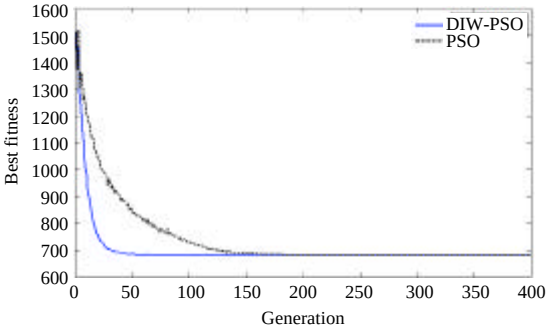

Fig. 8: Fitness of the best particle for problem G09

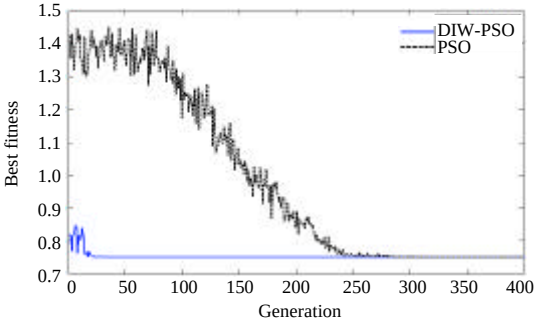

Fig. 11: Fitness of the best particle for problem G11

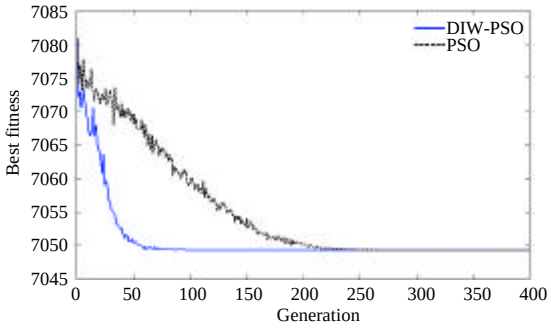

Fig. 9: Fitness of the best particle for problem G10

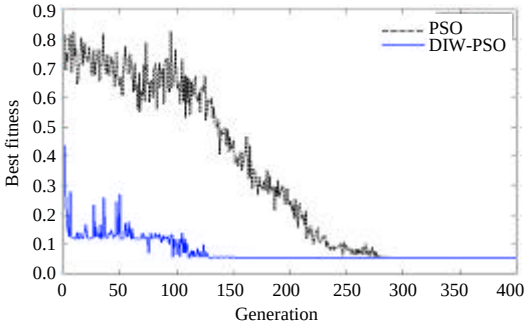

Fig. 12: Fitness of the best particle for problem G13

optimal solution which is better than the solution found using PSO. In these cases, DIW-PSO performed significantly better than PSO in terms of the median as well as the mean. In these problems, the convergence curves of PSO Shake violently. But DIW-PSO can found best solution in fewer generations.

For problem G03, the optimal solution is -1.000. The convergence curves of best fitness using DIW-PSO and PSO are shown in Fig. 13. From Table 1, it can be observed that DIW-PSO finds the optimal solution, while PSO does not. Using DIW-PSO, the best fitness curve converges in approximately 300 generations, while PSO didn't get to convergence in approximately 450 generations.

For problem G08, the optimal solution is 0.095825. The convergence curves of best fitness using DIW-PSO and PSO are shown in Fig. 14. From the Fig. 14, it can be observed that the curve of best fitness using PSO can't

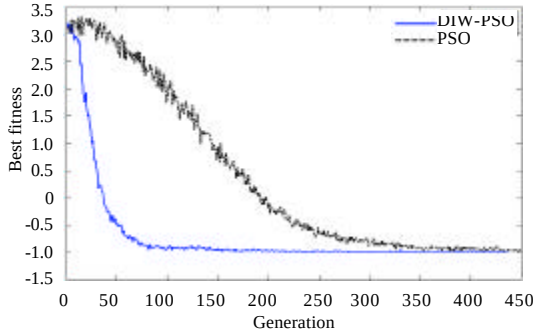

Fig. 13: Fitness of the best particle for problem G03

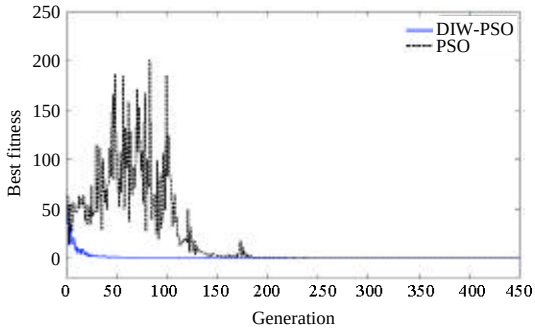

Fig. 14: Fitness of the best particle for problem G08

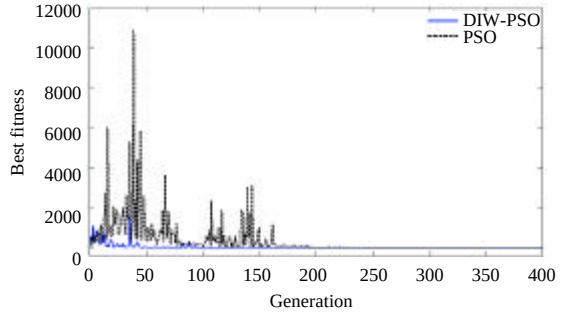

Fig. 15: Fitness of the best particle for problem G12

get convergence at first. After about 150 generations violently shake, the curve gets convergence finally. However, DIW-PSO can get convergence at first.

For the problem G12, the optimal solution is 1.000. The convergence curves of best fitness using DIW-PSO and PSO are shown in Fig. 15. In this case, according to Table 1, PSO finds the optimal solution and DIW-PSO finds a very close value to the optimal. The solution found by PSO presents a smaller standard deviation

Table 2: Statistic comparison between DIW-PSO and pso using the wilcoxon rank test

Problem	p	h	zval
G01	4.6391e-47	1	14.4075
G02	1.7483e-27	1	10.8620
G03	5.6020e-25	1	10.3220
G04	1.8653e-9	1	14.7837
G05	1.6560e-7	1	15.9838
G06	4.9548e-17	1	20.9037
G07	2.7784e-18	1	21.0408
G08	2.5587e-31	1	11.6407
G09	0.1720	0	1.3658
G10	2.6112e-017	1	8.4628
G11	6.9494e-10	1	21.2150
G12	5.5681e-34	1	-12.1524
G13	1.8794e-15	1	14.1496

(0.0126) than the results obtained by DIW-PSO (0.0233). The convergence curve of PSO shakes more horrible than DIW-PSO.

For the problems G01, G02, G03, G04, G05, G06, G07, G08, G09, G10, G11 and G13, it can be observed that DIW-PSO converges much faster than PSO and obtains solutions closer to the optimal and presents a smaller standard deviation. For the problem G12, however, PSO provides slightly better results than DIW-PSO.

For the validation of significance of results and comparison between the DIW-PSO and PSO algorithms, the Wilcoxon rank sum test using the statistical toolbox provided in Matlab was carried out. The statistic test of the results given in Table 1 is provided in Table 2. The Wilcoxon rank sum test indicates that the means of DIW-PSO and PSO are significantly different at the 95% confidence level, since h is equal to 1 in Table 2. The Wilcoxon rank sum test conforms that DIW-PSO performs better than standard PSO except for problem G12.

CONCLUSIONS

In this study, A PSO with a dynamic inertia weight (DIW-PSO) has been presented for solving constrained numerical optimization. The use of Euclidean distance for dynamic changing the inertia weight of PSO seems to provide a good compromise between the probability of having a large number of small amplitudes around the current points and a small probability of having larger amplitudes which may allow particles to move away from the current point and escape from local minima.

The DIW-PSO algorithm was compared with the standard PSO on the benchmark constrained optimization problems. The simulation results showed that the algorithm DIW-PSO outperforms the standard PSO. The DIW-PSO algorithm presents faster convergence and obtained solutions closer to the optimal.

REFERENCES

Cagnina, L., S. Esquivel and R. Gallard, 2004. Particle swarm optimization for sequencing problems: A case study. Proceeding of the Congress on Evolutionary Computation, June 19-23, Portland, pp: 536-541.

Cagnina, L.C., S.C. Esquivel and C.A. Coello Coello, 2006. A particle swarm optimizer for constrained numerical optimization. Proceedings of the 9th International Conference on Parallel Problem Solving from Nature (PPSN'06), Reykjavik, Island, pp: 910-919.

Cagnina, L.C., S.C. Esquivel and C.A. Coello, 2007. A Bi-population PSO with a shake-mechanism for solving constrained numerical optimization. Proceeding of the IEEE Congress on Evolutionary Computation, Sept. 25-28, IEEE, Singapore, pp: 670-676.

Coath, G. and S.K. Halgamuge, 2003. A comparison of constraint handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems. Congr. Evol. Computat., 4: 2419-2425.

Cui, H. and Q. Zhu, 2007. Convergence analysis and parameter selection in particle swarm optimization. Comput. Eng. Appl., 42: 89-91.

Du, D. and P.M. Pardalos, 1995. Minimax Theory and Applications. Kluwer, Norwell, MA., USA., pp: 293.

Eberhart, R.C. and J. Kennedy, 1995. A new optimizer using particle swarm theory. Proceedings of the 6th International Symposium on Micro Machine and Human Science, Oct. 4-6, Nagoya, Japan, pp: 39-43.

Hernane, S., Y. Hernane and M. Benyettou, 2010a. An asynchronous parallel particle swarm optimization algorithm for a scheduling problem. J. Applied Sci., 10: 664-669.

Hernane, S., Y. Hernane and M. Benyettou, 2010b. Migration algorithm of particle swarm optimization for a scheduling problem. J. Applied Sci., 10: 699-703.

Hooshmand, R.A., 2008. Optimal design of load shedding and generation reallocation in power systems using fuzzy particle swarm optimization algorithm. J. Applied Sci., 8: 2788-2800.

Hu, X., R.C. Eberhart and Y. Shi, 2003a. Engineering optimization with particle swarm. Proceedings of the IEEE Swarm Intelligence Symposium, April 24-26, IEEE Computer Society, Washington, DC., USA., pp: 53-57.

Hu, X., R.C. Eberhart and Y. Shi, 2003b. Swarm intelligence for permutation optimization: A case study on n-queens problem. Proceeding of the Swarm Intelligence Symposium, April 24-26, Indianapolis, Indiana, pp: 243-246.

Kennedy, J. and R. Eberhart, 1995. Particle swarm optimization. Proc. IEEE Int. Conf. Neural Networks, 4: 1942-1948.

Kennedy, J. and W. Spears, 1998. Matching algorithm to problems: An experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator. Proceeding of IEEE Conference on Evolutionary Computation. (IEEE'98) Anchorage, Alaska, pp: 78-83.

Kennedy, J., 1999. Small world and mega-minds: Effects of neighborhood topologies on particle swarm performance. Proceedings of the 1999 Congress on Evolutionary Computation, July 6-9, Washington, DC., USA., pp: 1931-1938.

Kennedy, J. and R. Eberhart, 1999. The Particle Swarm Social Adaptation in Information-Processing Systems. In: New Ideas in Optimization, Corne, D., M. Dorigo and F. Glover (Eds.). McGraw-Hill, London, UK., ISBN-13: 9780077095062..

Kennedy, J., R.C. Eberhart and Y. Shi, 2001. Swarm Intelligence. Morgan Kaufmann., San Francisco, CA., USA..

Krohling, R.A. and L.S. Coelho, 2006. Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems. Trans. Syst. Man Cybern-Part B: Cybern, 36: 1407-1416.

Liang, J.J. and P.N. Suganthan, 2006. Dynamic multi-swarm particle swarm optimizer with a novel constraint-handling mechanism. Proceeding of Congress on Evolutionary Computation, Sept. 11, Vancouver, BC., Canada, pp: 9-16.

Mezura-Montes, E. and C.A. Coello, 2005. A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Trans. Evol. Comput., 9: 1-17.

Michalewicz, Z. and M. Schoenauer, 1996. Evolutionary algorithms for constrained parameter optimization problems. Evol. Comput., 4: 1-32.

Parsopoulos, K.E. and M.N. Vrahatis, 2002. Particle Swarm Optimization Method for Constrained Optimization Problems. In: Intelligent Technologies-Theory and Applications: New Trends in Intelligent Technologies, Parsopoulos, K.E. and M.N. Vrahatis (Eds.). The IOS Press, New York, pp: 214-220.

Ren, Z., M. Pham, W. Li and C. Koh, 2010. A new global optimization algorithm for mixed-integer-discrete-continuous variables based on particles swarm optimization. Proceeding of the 14th Biennial Conference on Electromagnetic Field Computation, May 9-12, Chicago, IL., pp: 1-1.

Runarsson, T.P. and X. Yao, 2000. Stochastic ranking for constrained evolutionary optimization. IEEE Trans. Evol. Comput., 3: 284-294.

Shakiba, M., M. Teshnehlab, S. Zokaie and M. Zakermoshfegh, 2008. Short-term prediction of traffic rate interval router using hybrid training of dynamic synapse neural network structure. J. Applied Sci., 8: 1534-1540.

Shi, Y. and R. Eberhart, 1998. A modified particle swarm optimizer. Proceedings of the IEEE Congress on Evolutionary Computation, May 4-9, Piscataway, New Jersey, pp: 69-73.

Takahama, T. and S. Sakai, 2006. Solving constrained optimization problems by the e constrained particle swarm optimizer with adaptive velocity limit control. Proceeding of the IEEE Conference on Cybernetics and Intelligent Systems, June 7-9, Bangkok pp: 1-7.

Tian, Y., R. Zhu and Q. Xue, 2008. Research advances on inertia weight in particle swarm optimization. Comput. Eng. Appl., 44: 39-41.

Wang, X.H., Z. Qin, X.C. Heng and Y. Liu, 2008. Research on structure learning of product unit neural networks by particle swarm optimization. Inform. Technol. J., 7: 639-646.

Zhao, S.Z., J.J. Liang and P.N. Suganthan, 2008. Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization. Proceeding of IEEE Congress on Evolutionary Computation, June 1-6, Hong Kong, pp: 3845-3852.