International Journal of Computer Applications (0975 – 8887)
Volume 40– No.17, February 2012

A Hybrid PSO with Dynamic Inertia Weight and GA Approach for Discovering Classification Rule in Data Mining

S.M.Uma
Asst.Prof, Dept of CSE
Kings College of Engg
Thanjavur, Tamilnadu, India

K.Rajiv Gandhi
Asst.Prof, Dept of CSE
Kings College of Engg, Thanjavur
Tamilnadu, India

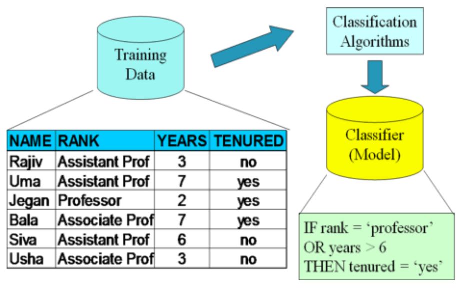
Dr.E.Kirubakaran
General Manger
out Source Unit
BHEL Trichy, India

ABSTRACT
Data Mining is the efficient knowledge discovery form database. It is also form of knowledge discovery essential for solving problem in specific domain like health care, business and other field. The proposed system is based on population based on heuristic search technique, which can used to solve combinatorial optimization problem. Our research focus on studying the hybrid algorithm that result in performance and enhancement in classification rule discovery task. In standard Particle Swarm Optimization (PSO) the non oscillatory route can quickly cause a particle to stagnate and also it may prematurely converge on suboptimal solution that is not even guaranteed to local optimal solution. In this paper we have present novel hybrid algorithm, PSO with Dynamic Inertia Weight and Genetic Algorithm (GA) approach for classification rule. The selection of inertia weight was very important to ensure the convergent behavior of particle In this hybrid algorithm approach incorporates a dynamic inertia weight in order to help the algorithm to find global and overcome the problem convergence to local optima, essentially GA can perform a global search over the entire search space with faster convergence speed. Thus the hybrid algorithm is easily implemented because of use of simple classifier it has, its computational complexity is low, are the special characteristics for the use of this hybrid algorithm.

Keywords
Genitic Algorithm; Particle Swarm Optimization; Classification;

1. INTRODUCTION
The data mining technology [1] is one of the comprehensive applications of technology item relying on the database technology, medical analysis, statistical analysis, artificial intelligence and other field. It is found in large data warehouse and extract hidden in which information is a new technology, helping decision-makers search Data between the potential links and found neglected factor.

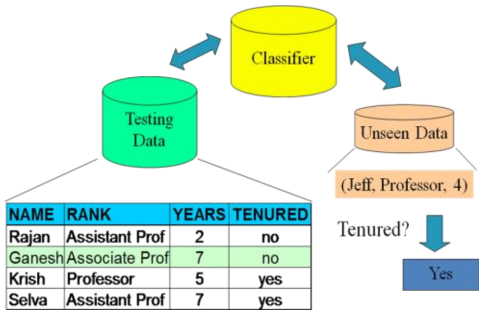
1.1 Data Mining Classification
The classification process [3] has two phases; the first phase is learning process whereby training data are analyzed by classification algorithm. Learned model or classifier is represented in the form of classification rules. The second phase is classification, and test data are used to estimate the accuracy of classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data (Fig. 1, 2). Some of the techniques that are used for data classification are decision tree, Bayesian methods, Bayesian network, rule-based algorithms, neural network, support vector machine, association rule mining, k-nearest-neighbor, case-based reasoning, genetic algorithms, rough sets, fuzzy logic. In this study, our discussion focuses on classification techniques i.e. heuristic search. However, decision tree and neural network are found useful in developing predictive models in many fields.



NAME	RANK	YEARS	TENURED
Rajiv	Assistant Prof	3	no
Uma	Assistant Prof	7	yes
Jegan	Professor	2	yes
Bala	Associate Prof	7	yes
Siva	Assistant Prof	6	no
Usha	Associate Prof	3	no

Fig 1: Classification (Training Data Set)

1.2 Rule Discovery
Rules [2] can express general knowledge about actions or conclusions in given circumstances. In the if-then format, rules are an easy way to represent cognitive processes in psychology and a useful means to encode expert knowledge. Here we define the *domain rules* as the set of rules that bear domain semantics and can explain all the domain instances.



NAME	RANK	YEARS	TENURED
Rajan	Assistant Prof	2	no
Ganesh	Associate Prof	7	no
Krish	Professor	5	yes
Selva	Assistant Prof	7	yes

Fig.1.Classification (Training Data Set)

The task of rule learning is to learn or discover a set of rules from given instances called training instances. Ideally, the learned rules as a whole should explain not just the training instances but also unseen instances in the domain—an issue known as generalization. This forms the basis of evaluating the performance of a learning system.

2. PARTICAL SWARM OPTIMIZATION
The Particle swarm Optimization (PSO) Introduced by Kennedy and Eberhart in 1995 [4, 6], PSO is a stochastic, population-based evolutionary algorithm [5].It is based on social-psychological principles and provides insights into

social behavior, like simulates the behaviors of bird flocking. In PSO, a fitness function is designed to evaluate a particle solution for a given problem. A communication structure is also defined by connecting neighbors to each individual particle to interact with. At the beginning of evolution, an initial population of individual particles is defined by random guesses as candidate solutions. An iterative process to improve these candidate solutions using the fitness function is set in motion. The individual particles in the population are iteratively evaluated, are assigned a fitness value, and remember the locations at their best success. The individual's best solution is often called the local best. Each individual particle makes this information available to their neighbors. They are also able to see their neighbors' success (local best).

PSO algorithms make use of particles moving in an n- dimensional space to search for solutions for an n-variable function optimization problem. A particle decides where to move next, considering its own experience, which is the memory of its best past position, and the experience of its most successful neighbor. This process continues with the population finally converging on a global best solution [5]. The original PSO define the following equation

$$v[] = v[] + c1 * rand() * (pbest[] - present[]) + c2 * rand() * (gbest[] - present[]) \quad (1)$$

$$present[] = present[] + v[] \quad (2)$$

$v[]$ is the particle velocity, $present[]$ is the current particle (solution). $pbest[]$ and $gbest[]$ are defined as stated before $rand[]$ is a random number between (0, 1). c1, c2 are learning factors. Usually $c1 = c2 = 2.$ The PSO algorithm can be summarized as follows

Step1: Initialize: Initialize parameters and population with random position and velocities

Step2: Evaluation: Evaluate the fitness value for each particle.

Step3: Find the pbest: If the Fitness value of particle i is better than its best fitness value (pbest) in history, then set current fitness value as the new pbest to particle.

Step4: Find the gbest: If any pbest is updated and it is better than the current gbest, then set gbest to the current value.

Step5: Update velocity and position: Update velocity and move to next position according to equation (1) and (2)

Step6: Stopping Criterion: If the number of iteration or CPU time are met, then stop; otherwise go back to step 2.

3. PSO VARIANTS

The PSO has various variants such as Inertia Weight, Constriction Factor, Dynamic Inertia with Maximum Velocity Reduction and Discrete Optimization.

3.1 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 [8]. Inertia weight plays a key role in this process and can direct PSO to optimize algorithm. Indeed, the inertia weight determines the contribution rate of a particles previous velocity to its velocity at the current time step.

3.2 Constructor Factor

The convergence of PSO [7, 10] have introduced a constriction factor indicates that the use of a constriction factor may be necessary to insure convergence of the particle swarm algorithm. He had established some mathematical foundation to explain the behavior of a simplified PSO model in its search for an optimal solution.

$$v_{id}^{t+1} = k\left[v_{id}^{t} + c_1 r_1 \left(P_{id} - x_{id}^{t}\right) + c_2 r_2 \left(P_{gd} - x_{id}^{t}\right)\right] \quad (3)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (4)$$

Where the construction factor

$$k = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}$$

Where $\phi = c_1 + c_2$

3.3 Discrete PSO

A Discrete Particle Swarm Optimization (DPSO) proposed in [9] does not consider any velocity since, from the lack of continuity of the movement in a discrete space, the notion of velocity loses sense; however they kept the attraction of the best positions. They interpret the weights of the updating equation as probabilities that, at each iteration, each particle has a random behavior, or acts in a way guided by the effect of an attraction. The moves in a discrete or combinatorial space are jumps from one solution to another.

4. GENETIC ALGORITHM

John Holland, from the University of Michigan initiated his work on genetic algorithms at the beginning of the 1960s. His first achievement was the publication of Adaptation in Natural and Artificial System [11] in 1975.

The basic idea is as follows: the genetic pool of a given population potentially contains the solution, or a better solution, to a given adaptive problem. This solution is not "active" because the genetic combination on which it relies is split between several subjects. Only the association of different genomes can lead to the solution. The Simple Genetic Algorithm can be expressed in pseudo code with the following cycle:

```
Step1: Generate the initial population of Individuals randomly P (0).
Step2: While (number generations <= maximum _
              Numbers _ generations)
              Do:
                     {
                     Evaluati
                     on;
                     Selectio
                     n;
```

```
        Reproduction;
        Generation ++;
        }
Step3: Show result.
Step4: End of the Generation
```

Crossover: The idea behind a crossover operation is as follows: it takes as input 2 expressions, selects a random point, and exchanges the sub expressions behind this point. In general, however, not all attributes will be involved in an expression. This may have some undesired effects for a crossover. First, a crossover may produce individuals in which an attribute is involved more than once. Second, a crossover may result in an offspring that is exactly the same as the parents with probability 1.0. To prevent the above-mentioned effects, we apply the following technique to perform crossover. If two individual *p*1 and *p*2 have been selected for crossover, the crossover operator is considered as follow:

(1) If there are some same attributes between individual *p*1 and *p*2, then select one randomly among these attributes and exchange the corresponding term between the two individuals.

(2) If there aren't same attributes between individual *p*1 and *p*2, then select one attribute randomly from individual *p*1 and *p*2 respectively and exchange the corresponding term between the two individuals.

Mutation: Mutation is an important operator that acts a single individual at a time. This operator maintains the diversity of gene in the population and guarantees that the search is done in the whole solution space. Through mutation operator cannot always produce a better result, it play an important role for the global optimization. Considering an individual *p* with a length of *n*, the mutation operator is defined as:

(1) When *n*=*L*, select one attribute randomly from individual *p* and delete the corresponding term.

(2) When *L*>*n*>1, select one attribute randomly from individual *p* and calculate the fitness of individual *p*. If *F* (*p*)≠0, then delete the corresponding term or replace the attribute value by another on the mutation probability of *pm*. There the two operators are select irrespectively on the probability of 50%. If *F* (*p*) =0, then delete the corresponding term.

(3)When *n*=1, replace the present attribute value by another on the probability of *pm*.

Selection: For simplifying the implementation of a genetic algorithm, the mechanism to select individuals for a new generation is based on the technique of elitist recombination. According to this technique, the individuals in a population are randomly shuffled. Then, the cross-over operation is applied on each mating pair, resulting into two offspring. The parent and the offspring with the highest fitness value are selected to be mutated with a probability. Then they are added to the new generation. In this way, there is a direct competition between the offspring and their own parents and the offspring provides the possibility to explore different parts of a search space.

5. DYNAMIC INERTIA WEIGHT WITH PSO OPTIMIZER

Inertia Weight plays a key role in the process of providing balance between exploration and exploitation process. The Inertia Weight determines the contribution rate of a particle's previous velocity to its velocity at the current time step. The different characteristic of inertia weight as shown in the table I.

5.1 Discrete PSO

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. The following formula is the dynamic inertia and maximum velocity reduction

$$v_{k+1}^{i} = w_k v_k^i + c_1 r_1\left(p_k^i - x_k^i\right) + c_2 r_2\left(p_k^g - x_k^i\right) \tag{5}$$

Maximum velocity limited

$$v^{\max} = \gamma\left(x_{UB} - x_{LB}\right)$$

$$\textit{If}\quad f\left(p_k^g\right) \geq \left(p_{k-b}^g\right), \qquad \textit{then}\quad \omega_{k+1} = \alpha\omega_k, v_k^{\max} = \beta v_k^{\max},$$

$$\textit{with}\quad 0 < \alpha, \beta < 1$$

During the iteration, the search space p_g has a high probability of global optimal solution, so a high intensity search around p_g reached by any particle in the swarm should be considered. It hopes have a good local search space around p_g and particle out of search space around p_g have good global search capability.

6. HYBRIC DYNAMIC INERTIA WEIGHT WITH PSO AND GA

The hybrid algorithm combines the stand velocity and position update rules of PSO with idea of selection, cross over and mutation from GA. at beginning of algorithm randomly initialize the position and the velocity of each particle after evaluating particle and obtaining the best value: best, lbest and gbest. During the evaluation process, the position and the velocity of each particle are updated. In the evolutionary loop process, the position and the velocity of each particle are updated.

In the evaluation new dynamic inertia weight mechanism is applied to improve the global search capability and convergence of PSO algorithm. To avoid the premature convergence of the swarm particle, we use a reproduction mechanism using cross over and mutation when stuck at the local maximum.

The Dynamic inertia weight to control the convergence of the algorithm described bellow weighting function

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} * iter \qquad (6)$$

Where $w_{\max}$: Initial weight,

$w_{\min}$: Final weight,

$iter_{\max}$: Maximum Iteration number,

$iter$: Current Iteration Number.

Form the equation 6, which gradualist moves the current searching point close to pbest and gbest can be calculated. The pseudo code for proposed hybrid approach method.

```
Initialization the population
Initialize the velocity for population
Evaluate the fitness for population
Evaluate the pbest and gbest for population
Swarms fly through the search space
    Do
        For i=1 to number of particles do
            Search the best leader in neighborhood
            of particle and record in lbest
        Calculate $w(i,k)$ with the equation (6)
        For j= 1 to number of dimension do
        Update velocity with equation (1)
        Update the particle with equation (2)
    End
    End
  Evaluate the fitness by GA
  Evaluate pbest and gbest
 While (K>maxNumber)
 Result=gbest
 Return (result)
```

7. EXPERIMENTAL RESULT AND DISCUSSION

In this section, we study the accuracy, simplicity and computational cost of the new algorithm in comparison to PSO. Four datasets from UCI data repository such as Cleveland Heart Disease, Statlog-Heart, Hepatitis and Dermatology were used for this study. The data sets were reduced using dynamic inertia PSO optimizer criterion and the new reduced data sets are shown in Table II. We have evaluated comparative performance of the proposed hybrid algorithm and PSO using a 10-fold cross validation procedure. In this procedure each data set is divided into ten partitions, each method is run ten times, using a different partition as test set and the other nine partitions as the training set each time. We run the classifiers 10 times using a different random seed to initialize the search each time for each cross-validation fold.

Data Set	Total No. of Attributes	R/I/N	INSTANCES	CLASSES
Cleveland	13	13/0/0	297	5
Statlog-Heart	13	1/12/0	270	2
Hepatitis	19	2/17/0	80	2
Dermatology	34	0/34/0	358	6

Table II. Data set description

Data Set	Total No. of Attributes	R/I/N	INSTANCES	CLASSES
Cleveland	5	5/0/0	279	5
Statlog-Heart	5	1/4/0	256	2
Hepatitis	9	1/8/0	80	2
Dermatology	20	0/20/0	358	6

Table III. Reduced data sets

R-Real, I-Integer, N-Nominal Attributes

The comparison was carried out across three criteria, namely 1) the predictive accuracy of the discovered rule lists, 2) Their simplicity and 3) computational cost. In the first step of our two step approach, we applied the GA feature selection criterion to reduce the number of attributes and removed the duplicated examples (examples with the same values for all attributes) from the resulting reduced data set to avoid the possibility that a test set contains an example that is the same as a training example.

We made experiments with several reduced data sets for all of the four original data sets and we present in Table II the best ones. In the second step of our hybrid approach we run the PSO on the new reduced data sets. For all four data sets we performed experiments using the java with myra software for comparing accuracy, simplicity and computational cost between the hybrid algorithm and PSO. Due to space limitations we demonstrate part of these results in Figures 3, 4 and 5 which are the most representative ones. However we note our conclusions for all the cases tested. On the Statlog - Heart data set, it obtained clearly better accuracy in figure 4 and Compared to the original PSO with simpler rule list (Figures 3, 5) and less computational cost.

The results obtained in the experiments are summarized as follows. Overall, the proposed hybrid approach is a more accurate and simpler classifier method than the PSO. The results show that it is able to obtain improvements over PSO on basis of accuracy, simplicity and computational cost in heat data set as seen in Figures 3, 4 and 5.

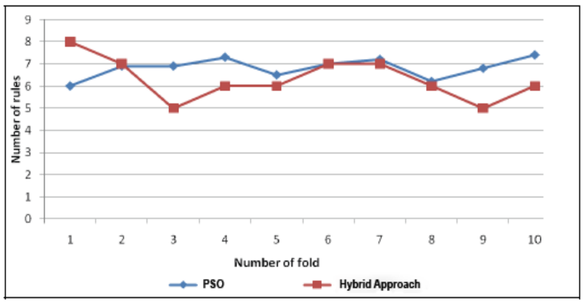


Fig 3: Number of Rules on Heart Data Set

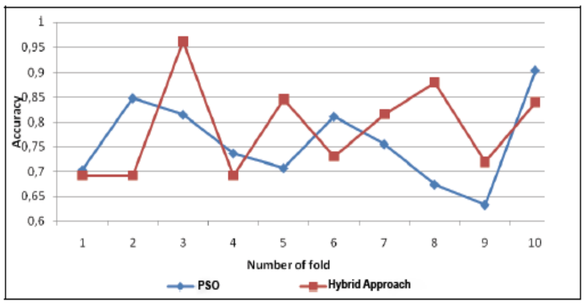


Fig 4: Accuracy on Heart Data Set

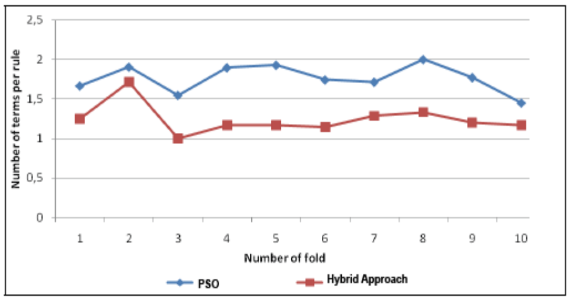


Fig 5: Number of terms per rule on heart data set

8. CONCLUSION

In this paper we present hybrid approach using GA and PSO with Dynamic Inertia Weight as modern optimal tool for discovering classifications rule. We have proposed the dynamic inertia weight with PSO is used to balance the global and local search capability. Dynamic Inertia Wight is to find the global and overcome the problem of premature we have discussed the design of genetic operator and fitness function for classification rule mining. Rule quality can be viewed in terms of accuracy and comprehensibility. A rule will be usable by medical practitioner if it is accuracy and easily understood. To balance the efficiency and exploration capability, extensive experiment need to be conducted with setting of parameter to arrive at the optimal value for these algorithms. We have explored the use of PSO with Dynamic Inertia Weight and GA algorithm for discovering predictive, comprehensible and interesting rules. The hybrid algorithm presented fast convergence and we obtained solution closer to optimal.

Name of Inertia Weight	**Formula of Inertia Weight**
Constant Inertia Weight	$w = 0$ $c = 0.7$ (Consider for Experiments) [12]
Constant Inertia Weight	$w = 0.5 + \frac{Rand(\ )}{2}$ [13]
Random Inertia Weight	$w_i(t+1) = w(0) + (w(n_t) - w(0))\frac{e^{m_i(t)-1}}{e^{m_i(t)+1}}$ [14]
	$m_i(t) = \frac{gbest - current}{gbest + current}$
Adaptive Inertia Weight	$w_k = \frac{(w_{start} - w_{end})}{(1 + e^{u*(k-n*gen)})} + w_{end}$, $u = 10^{(\log(gen)-2)}$ [15]
Sigmoid Increasing Inertia Weight	$w_k = \frac{(w_{start} - w_{end})}{(1 + e^{-u*(k-n*gen)})} + w_{end}$, $u = 10^{(\log(gen)-2)}$ [15]
Sigmoid Decreasing Inertia Weight	$w_k = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} + w_{end}$ [16]
Linear Decreasing Inertia Weight	$z = 4 * z * (1 - z)$ $w = (w_1 - w_2) * \frac{MAXiter - iter}{MAXiter} + w_2 * z$ [17]
The Chaotic Inertia Weight	$z = 4 * z * (1 - z)$, $w = 0.5 * rand() + 0.5 * z$ [17]
Chaotic Random Inertia Weight	$w(t) = \frac{w_{\min} + w_{\max}}{2} + \frac{w_{\max} - w_{\min}}{2} \cos\left(\frac{2\pi t}{T}\right)$, $T = \frac{2S_1}{3 + 2k}$ [18]
Oscillating Inertia Weight	$Inertiaweightw_i = \left(1.1 - \frac{gbest_i}{pbest_i}\right)$ [19]
Global-Local Best Inertia Weight	$w_k = w_{\min} + (w_{\max} - w_{\min}) * \lambda^{(k-1)}$, $\lambda = 0.95$ [20]
Simulated Annealing Inertia Weight	$w(t) = w_{\min} + (w_{\max -} w_{\min}).e^{-\frac{\frac{t}{MAXITER}}{10}}$ [21]
Natural Exponent Inertia Weight Strategy (e1-PSO)	$w(t) = w_{\min} + (w_{\max -} w_{\min}).e^{-[\frac{t}{MAXITER/4}]^2}$ [21]
Natural Exponent Inertia Weight Strategy (e2 - PSO)	$w = w_{\max} + (w_{\min} - w_{\max}) * \log_{10}\left(a + \frac{10t}{T_{\max}}\right)$ [22]

Exponent Decreasing Inertia Weight	$w = w_{\max} + \left(w_{\min} - w_{\max} - d_1\right)^* \exp\left(\dfrac{1}{1 + d_2 t / t_{\max}}\right)$	[23]

9. ACKNOWLEDGMENTS
The authors thankful to Kings College of Engineering, Thanjavur and BHEL Trichy, for providing necessary support to carry research work.

10. REFERENCES

[1] Zhang Haiyang A Short Introduction to Data Mining and Its Applications Aug. 2011 IEEE

[2] Li Min Fu, *Senior Member, IEEE* and Edward H. Shortliffe The Application of Certainty Factors to Neural Computing for Rule Discovery MAY 2000 in IEEE

[3] J. Han and M. Kamber, Data Mining: Concepts and Techniques. San Francisco: Morgan Kaufmann Publisher, 2006.

[4] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Neural Networks, 1995. Proceedings. IEEE International Conference on, vol. 4 1995, pp. 1942–1948 vol.4.

[5] Particle Swarm Optimization, Wikipedia. http://en.wikipedia.org/wiki/Particle swarm optimization.

[6] Particle Swarm Optimization, http://www.swarmintelligence.org/tutorials.php.

[7] LI Ming*, JI Xue-Ling, LI Wei An Improved Constriction Factor Particle Swarm Optimization Algorithm to Overcome the Local Optimum,

[8] Proceedings of the 30th Chinese Control Conference July 22-24, 2011, Yantai, China.

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

[10] Moreno-Perez JA, Castro-Gutierrez JP, Martnez-Garca FJ, Melian B, Moreno-Vega JM, Ramos J (2007) Discrete Particle Swarm Optimization for the p-median problem. In: Proceedings of the 7th Metaheuristics International Conference, Montreal, Canada.

[11] M.Clerc, J.Kennedy. The particle swarm-explosion, stability, and convergence in a multidimensional complex space[J]. *IEEE Trans. Evolutionary Computation.* 2002.6(1):58-73.

[12] Y. Shi and R. Eberhart., “A modified particle swarm optimizer”, In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on, pages 69–73. IEEE, 2002.

[13] R.C. Eberhart and Y. Shi., “Tracking and optimizing dynamic systems with particle swarms”, In Evolutionary Computation, 2001. Proceedings of the 2001 Congress on, volume 1, pages 94–100 IEEE, 2002.

[14] A.Nikabadi, M.Ebadzadeh , “Particle swarm optimization algorithms with adaptive Inertia Weight : A survey of the state of the art and a Novel method”, IEEE journal of evolutionary computation , 2008.

[15] R.F. Malik, T.A. Rahman, S.Z.M. Hashim, and R. Ngah, “New Particle Swarm Optimizer with Sigmoid Increasing Inertia Weight”, International Journal of Computer Science and Security (IJCSS), 1(2):35, 2007.

[16] J. Xin, G. Chen, and Y. Hai., “A Particle Swarm Optimizer with Multistage Linearly-Decreasing Inertia Weight”, In Computational Sciences and Optimization, 2009. CSO 2009. International Joint Conference on, volume 1, pages 505–508. IEEE, 2009.

[17] Y. Feng, G.F. Teng, A.X. Wang, and Y.M. Yao., “Chaotic Inertia Weight in article Swarm Optimization”, In Innovative Computing, Information and Control, 2007. ICICIC’07. Second International Conference on, page 475. IEEE, 2008.

[18] K. Kentzoglanakis and M. Poole., “Particle swarm optimization with an oscillating Inertia Weight”, In Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pages 1749–1750. ACM, 2009.

[19] M.S. Arumugam and MVC Rao., “On the performance of the particle swarm optimization algorithm with various Inertia Weight variants for computing optimal control of a class of hybrid systems”, Discrete Dynamics in Nature and Society, 2006, 2006.

[20] W. Al-Hassan, MB Fayek, and SI Shaheen, “Psosa: An optimized particle swarm technique for solving the urban planning problem”, In Computer Engineering and Systems, The 2006 International Conference on, pages 401–405. IEEE, 2007.

[21] G. Chen, X. Huang, J. Jia, and Z. Min., “Natural exponential Inertia Weight strategy in particle swarm optimization”, In Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on, volume 1, pages 3672–3675. IEEE, 2006.

[22] Y. Gao, X. An, and J. Liu., “A Particle Swarm Optimization Algorithm with Logarithm Decreasing Inertia Weight and Chaos Mutation”, In Computational Intelligence and Security, 2008. CIS’08. International Conference on, volume 1, pages 61–65. IEEE, 2008.

[23] H.R. Li and Y.L. Gao., “Particle Swarm Optimization Algorithm with Exponent Decreasing Inertia Weight and Stochastic Mutation”, In —2009 Second International Conference on Information and Computing Science, pages 66–69. IEEE, 2009.