Association Rule Mining using Self Adaptive Particle Swarm Optimization

Indira.K
Research Scholar
Department of CSE
Pondicherry Engineering College
Puducherry, India

Kanmani.S
Professor
Department of IT
Pondicherry Engineering College
Puducherry, India

ABSTRACT
Particle swarm optimization (PSO) algorithm is a simple and powerful population based stochastic search algorithm for solving optimization problems in the continuous search domain. However, the general PSO is more likely to get stuck at a local optimum and thereby leading to premature convergence when solving practical problems. One solution to avoid premature convergence is adjusting the control parameters, inertia weight and acceleration coefficients. This paper proposes two adaptive mechanisms for adjusting the inertia weights namely self adaptive PSO1 (SAPSO1) and self adaptive PSO2 (SAPSO2) for mining association rules. The accuracy of the mined rules by these two algorithms when compared to weighted PSO shows that the self adaptive PSO produces better results when compared to weighted PSO.

Keywords
Particle Swarm optimization, Association Rules, Inertia Weight, SAPSO1, SAPSO2.

1. INTRODUCTION
Data mining extracts implicit, previously unknown, and potentially useful information from large databases. Data mining consists of several tasks depending on application domain and user interest. Association rule mining is one among the most widely used task in data mining [1, 14]. Association rule mining is a discovery of interesting patterns or relations between variables in large databases. These relationships can be represented as IF–THEN statement. IF <some conditions are satisfied > THEN <predict some values of other attribute(s) >. The conditions associated in the IF part is termed as antecedent and those with the THEN part is called the consequent.

Apriori and FP growth tree have been the standard algorithms for mining association rules. The increased input/output overhead and inability to mine rules from huge databases made researchers to seek for other methods. Evolutionary algorithms provide robust and efficient approach in exploring large search space. Evolutionary algorithms are applicable for problems where no good method is available and most suitable in problems where multiple solutions are required. Genetic algorithm (GA) and Particle Swarm Optimization (PSO), both being population based search methods are more suitable for association rule mining. This study explores the application of self adaptive PSO for mining association rules.

The concept of particle swarm optimization was put forth by Kennedy and Eberhart [6, 9, 10]. It has been proved to be efficient at solving engineering problems [6]. The advantages of PSO over many other optimization algorithms are its implementation simplicity and ability to reasonable convergence.

The study is divided into six sections including the introduction. In Section 2, a brief description of basic PSO is given; Section 3 gives the literature review of self adaptive PSO algorithms. In Section 4, we give the outline of the self adaptive PSO algorithm proposed for the purpose of mining association rules. In Section 5 the experimental settings and result analysis of the algorithm are discussed. Finally the conclusions based on the present study are drawn in Section 6.

2. PARTICLE SWARM OPTIMIZATION
PSO has emerged as one of the most promising optimizing technique for solving global optimization problems. Its mechanism is inspired by the social and cooperative behavior displayed by various species like birds, fish etc including human beings. The PSO system consists of a population (swarm) of potential solutions called particles.

In the past several years, PSO has been successfully applied in many research and application areas. It has been demonstrated that PSO gets better results in a faster and cheaper way in comparison to other methods like GA, simulated annealing (SA) etc.

The particles move through the search domain with a specified velocity in search of optimal solution. For D-dimensional search space the velocity is represented as

$$V_i = (v_{i1}, v_{i2}, \dots v_{id}, \dots v_{iD}) \quad (1)$$

Each particle maintains a memory which helps it in keeping track of its previous best position. The position of the ith particle is represented as

$$X_i = (x_{i1}, x_{i2}, \dots x_{id}, \dots x_{iD}) \quad (2)$$

The positions of the particles are distinguished as personal best (pBest) and global best (gBest). The personal best and global best is represented as

$$P_i = (p_{i1}, p_{i2}, \dots p_{id}, \dots p_{iD}) \quad (3)$$

$$P_g = (p_{i1}, p_{i2}, \dots p_{id}, \dots p_{iD}) \quad (4)$$

The particles or members of the swarm fly through a multidimensional search space looking for a potential solution as shown in figure 1.

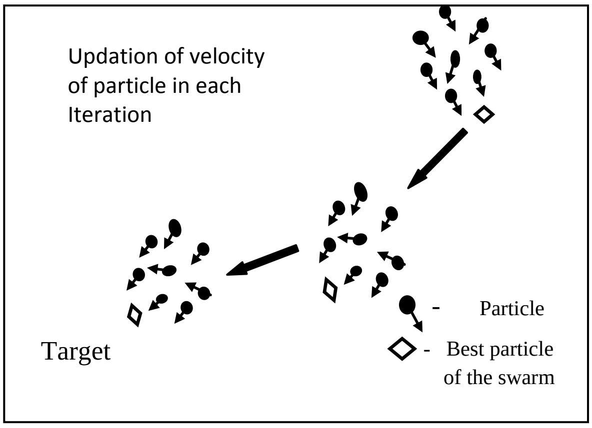

Figure 1. Swarm following the best particle to move to the goal

Each particle adjusts its position in the search space from time to time according to the flying experience of its own and of its neighbors. Each particle updates its corresponding velocity and position with Equations 5 and 6 as follows

$$v_i^{new} = \omega * v_i^{old} + c_1\, rand(\ \ )(pbest - x_i) + c_2 rand(\ \ )\ (gbest - x_i) \quad (5)$$

$$x_i^{new} = x_i^{old} + v_i^{new} \quad (6)$$

Where v_i^{old} is the particle velocity of the i^{th} particle, x_i is the current particle, I is the particle number, rand ( ) is a random number in the (0,1), c1 the individual factor and c_2 the societal factor.

Both c_1 and c_2 are usually set to be 2 in all literature works analyzed and hence the same adopted here. The velocity V_i of each particle is clamped to a maximum velocity V_{max} which is specified by the user. V_{max} determines the resolution with which regions between the present position and the target position are searched. Large values of V_{max} facilitate global exploration, while smaller values encourage local exploitation. If V_{max} is too small, the swarm may not explore sufficiently beyond locally good regions.

The algorithm for PSO is depicted below

Step 1. Initialize the population - locations and velocities

Step 2. Evaluate the fitness of the individual particle (lBest)

Step 3. Keep track of the individual highest fitness (gBest)

Step 4. Modify velocities based on velocity updation function

Step 5. Update the particles position

Step 6. Terminate if the condition is met

Step 7. Go to Step 2

3. REVIEW ON PSO ALGORITHMS

PSO is a population-based, stochastic optimization algorithm based on the idea of a swarm moving over a given landscape. The algorithm adaptively updates the velocities and positions of the members of the swarm by learning from the good experiences. The velocity update equation plays a major role in enhancing the performance of the PSO. However, similar to other evolutionary computation algorithms, the PSO is also a population-based iterative algorithm and the standard PSO algorithm can easily get trapped in the local optima when solving complex multimodal problems [12]. These weaknesses have restricted wider applications of the PSO [6]. Therefore, accelerating convergence speed and avoiding the local optima have become the two most important and appealing goals in PSO research.

To balance the global search and local search, inertia weight (ω) was introduced. It can be a positive constant or even a positive linear or nonlinear function of time [17]. 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. Eberhart and Shi [5] proposed a Random Inertia Weight strategy and experimentally found that this strategy increases the convergence of PSO in early iterations of the algorithm. In Global-Local Best Inertia Weight [2], the Inertia Weight is based on the function of local best and global best of the particles in each generation. It neither takes a constant value nor a linearly decreasing time-varying value. Using the merits of chaotic optimization, Chaotic Inertia Weight has been proposed by Feng et al. [7].

Gao et al. [8] proposed a new PSO algorithm which combined the Logarithm Decreasing Inertia Weight with Chaos mutation operator. Adaptive parameter control strategies can be developed based on the identified evolutionary state and by making use of existing research results on inertia weight [4, 18, 19] and acceleration coefficients [15, 16, 20, 22]. Some strategies adjust the parameters with a fuzzy system using fitness feedback [15, 19]. Some use a self-adaptive method by encoding the parameters into the particles and optimizing them together with the position during run time [20, 22].

Zhi-Hui Zhan et al. [24] propose an adaptive particle swarm optimization (APSO) consisting of two main steps, evaluating the population distribution and particle fitness in first step followed by elitist learning strategy when the evolutionary state is classified as convergence state to improve the search efficiency and convergence speed. [3] studies fifteen relatively recent and popular Inertia Weight strategies and compares their performance on five optimization test problems.

Self learning based PSO (SLPSO) [23] simultaneously adopts four PSO based search strategies and the generates better quality from past generation based on self adaptive method. Chaotic operators generated from chaotic maps [21] substitute random numbers in standard PSO. This improves the global convergence and to prevent to trap into local optima. Besides, the adaptive lattice search to enhance the accuracy of local solution. Both contribute to a more accurate global solution.

4. SELF ADAPTIVE PARTICLE SWARM OPTIMIZATION (SAPSO) ALGORITHM FOR AR MINING

This section briefly discusses association rules and its related factors. Two novel approaches for making the inertia weight self adaptive is proposed. The measurement for analyzing the efficiency of the rules mined is also discussed briefly.

4.1 Association Rules

Association rule mining finds interesting associations and/or correlation relationships among large set of data items. Association rules show attributes value conditions that occur frequently together in a given dataset.

The two major factors related to association rules are support and confidence. Support implies frequency of occurring patterns, and confidence means the strength of implication they are defined as follows:

An itemset, X, in a transaction database, D, has a support, denoted as sup(X) or simply p(X), that is the ratio of transactions in D containing X. Or

$$\sup(x) = \frac{No.of\ transactions\ \ containing\ \ X}{Total\ \ No.of transactions}$$

(7)

The confidence of a rule X → Y, written as conf(X→Y), is defined as

$$conf\ (X\ \rightarrow Y) = \frac{\sup\ (X \cup Y)}{\sup\ (X)} \tag{8}$$

4.2 Self Adaptive Particle Swarm Optimization (SAPSO)

The original PSO has pretty good convergence ability, but also suffers the demerit of premature convergence, due to the loss of diversity. Improving the exploration ability of PSO has been an active research topic in recent years. Thus, the proposed algorithm introduces the concept of selfadaptation as the primary key to tune the two basic rules velocity and position. By improving the inertia weight formulae in PSO the diversity of population could be achieved. The basic PSO, presented by Eberhart and Kennedy in 1995 [3], has no inertia weight. In 1998, first time Shi and Eberhart [7] presented the concept of constant inertia weight.

By looking at equation (5) more closely, it can be seen that the maximum velocity allowed actually serves as a constraint that controls the maximum global exploration ability PSO can have. By setting a too small maximum for the velocity allowed, the maximum global exploration ability is limited, and PSO will always favor a local search no matter what the inertia weight is. By setting a large maximum velocity allowed, then the PSO can have a large range of exploration ability to select by selecting the inertia weight. Since the maximum velocity allowed affects global exploration ability indirectly and the inertia weight affects it directly, it is better to control global exploration ability through inertia weight only. A way to do that is to allow inertia weight itself to control exploration ability. Thus the inertia weight is made to change automatically (self adaptive). Two self adaptive inertia weights are introduced for mining association rules in this paper.

In order to linearly decrease the inertia weight as iteration progresses the inertia weight is made adaptive through the equation 9 in SAPSO1.

$$\omega = \omega_{max} - (\omega_{max} - \omega_{min})\frac{g}{G}$$

(9)

Where ω_{max} and ω_{min} are the maximum and minimum inertia weights, g is the generation index and G is the predefined maximum number of generation.

In SAPSO2 the inertia weight adaptation is made to depend upon the values from previous generation so as to linearly decrease its value with increasing iterations as shown in equation 10.

$$\omega(t+1) = \omega(t) - \frac{\omega_{\max} - \omega_{\min}}{G}$$

(10)

Where $\omega(t+1)$ is the inertia weight for the current generation, $\omega(t)$ inertia weight for the previous generation, ω_{max} and ω_{min} are the maximum and minimum inertia weights and G is the predefined maximum number of generation.

The steps in self adaptive PSO1 and PSO 2 are as follows.

Step1: Initialize the position and velocity of particles.

Step 2: The importance of each particle is studied utilizing fitness function. Fitness value is evaluated using the fitness function. The objective of the fitness function is maximization. Equation 11 describes the fitness function.

$$Fitness(x) = conf(x) \times \log(\sup(x) \times length\ (x) + 1)$$

(11)

where fitness(x) is the fitness value of the association rule type x, sup(*x*) and conf(*x*) are as described in equation 1 and 2 and length(x) is the length of association rule type *x.* If the support and confidence factors are larger then, greater is the strength of the rule with more importance.

Step 3: Get the particle best and global best for the swarm. The particle best is the best fitness attained by the individual particle till present iteration and the overall best fitness attained by all the particles so far is the global best value.

Step 4: Set $\boldsymbol{\omega}_{\max}$ as 0.9 and $\boldsymbol{\omega}_{\min}$ as 0.4 and find the adaptive weights for both SAPSO1 and SAPSO2. Update velocity of the particles using equation 5.

Step 5: Update position of the particles using equation 6.

Step 6: Terminate if the condition is met.

Step 7: Go to step 2.

4.3 Predictive Accuracy

Predictive accuracy measures the effectiveness of the rules mined. The mined rules must have high predictive accuracy.

$$Predictive\ accuracy = \frac{|X\ \&\ Y|}{|X|} \quad (12)$$

where |X&Y| is the number of records that satisfy both antecedent X and consequent Y, |X| is the number of rules satisfying the antecedent X.

5. EXPERIMENTAL SETTING AND RESULT ANALYSIS

Three datasets from University of California Irvine Machine Learning Repository namely Car Evaluation, Haberman's Survival and Lenses are taken up for evaluating the performance of self adaptive particle swarm optimization algorithms SAPSO1 and SAPSO2. The experiments were carried out in Java on windows platform. The datasets considered for the experiments and initial parameter values set is listed in Table 1.

The initial velocity set was 0 for all the datasets and the learning factors c_1 and c_2 is 2. Maximum number of iterations carried out is 50.

Table 1. Datasets Information

Dataset	Attributes	Instances	Attribute characteristics	Swarm Size
Lenses	4	24	Categorical	24
Car Evaluation	6	1728	Categorical, Integer	700
Haberman's Survival	3	310	Integer	300

The effects of varying inertia weights on predictive accuracy are evaluated by varying the inertia weights. The results recorded are potted as shown in figure 2.

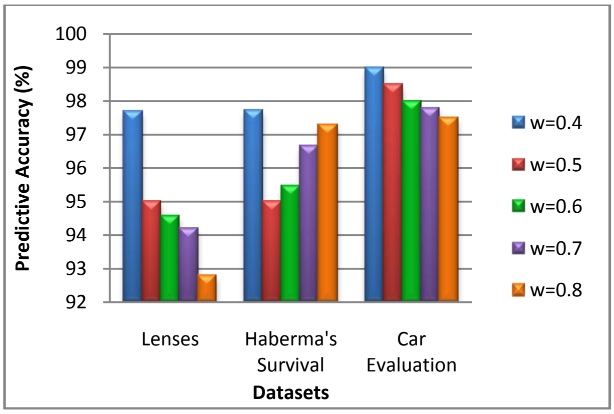

Figure 2. Comparison of Predictive Accuracy for different inertia weights

It is observed from the figure that the predictive accuracy is high when inertia weight is low for all three datasets. For Lenses and Car Evaluation datasets the predictive accuracy decreases as the inertia weight decreases. But for Haberman's survival dataset the predictive accuracy starts increasing after initial stages. The age attribute containing values with broad range results in the non linearity in predictive accuracy plot.

The inertia weight (ω) factor in velocity updation function is made self adaptive by using the two equations given in 9 and 10. The predictive accuracy for both SAPSO1 and SAPSO2 are plotted in figure3. SAPSO1 results in enhanced accuracy for all the three datasets. In case of SAPSO2 lenses and car evaluation dataset works better when compared to Haberman's survival dataset. The ω(t) used in SAPSO2 when applied for Haberman's survival dataset produces inconsistent velocity due to the age attribute values.

Introduction of adpativeness in inertia weight enhances the accuracy of the system considerably. The inertia weight controls the impact of previous flying experience, which is utilized to keep the balance between exploration and exploitation. The particle adjusts its position according to its best experience and learns the information of its neighbors. In addition, the inertia weight is also an important convergence factor; the smaller the inertia weight, the faster the convergence of PSO performances and the easier to be trapped into local best. Conversely, it improves the diversity of the solutions avoiding convergence at local optima.

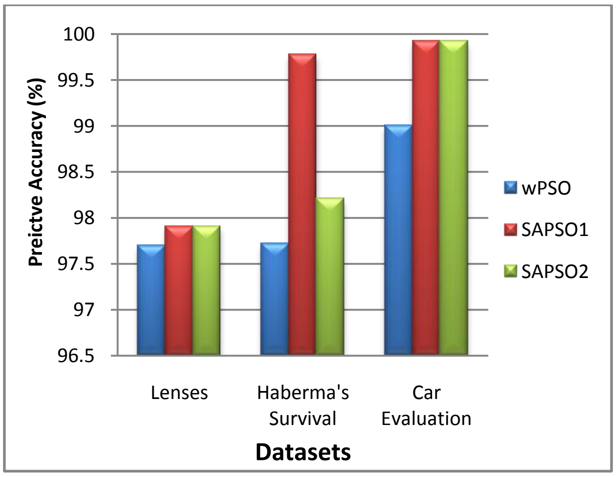

Figure 3. Predictive Accuracy Comparison for SAPSO Algorihms

According to comparison analysis in figure 3 it could be concluded that self adaptivess of the control parameters results in enhancement of the accuracy thereby increasing the ability of PSO for mining association rules.

6. CONCLUSION

In order to enhance the capability of particle swarm optimization, this paper proposed adaptive inertia weight in velocity updation. Two adaptive methods SAPSO1 and SAPSO2 are introduced for inertia weight factor. SAPSO1 is based on maximum and minimum inertia weight, while in SAPSO2 the current inertia weight relies on inertia weight factor of previous generation.

Based on the significantly encouraging results obtained from the experiments, it can be concluded that SAPSO1 And SAPSO2 significantly improves the PSO's performance and gives the best performance when compared with traditional

PSO. Another attractive property of the SAPSO is that it does not introduce any complex operations to the original simple PSO framework. The only difference from the original PSO is the introduction of self-adaptive strategy. The SAPSO is also simple and easy to implement like the original PSO. There are still several problems remaining to be investigated, such as adapting the acceleration coefficients and testing with more datasets of other kinds.

REFERENCES

[1] Agrawal, R., Imielinski, T., and Swami, A. 1993 Mining association rules between sets of items in large databases, Proceedings of the ACM SIGMOD International Conference on Management of Data. New York, ACM Press, pp 207–216.

[2] Arumugam, M.S., and Rao, M.V. "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. Discrete Dynamics in Nature and Society,Vol.2006, Article ID 79295,pp.1–17, 2006.

[3] Bansal, J.C., Singh, P.K., Mukesh Saraswat, Abhishek Verma, Shimpi Singh Jadon, and Ajith Abraham. 2011 Inertia Weight Strategies in Particle Swarm Optimization, Third World Congress on Nature and Biologically Inspired Computing , pp. 633 – 640.

[4] Chatterjee, A., and Siarry, P. "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization", Computational Operation Research, vol. 33, no. 3, pp. 859–871, Mar. 2004.

[5] Eberhart, R.C., and Shi,Y. 2001 Tracking and optimizing dynamic systems with particle swarms, Proceedings of the Congress on Evolutionary Computation, IEEE, volume 1, pp 94–100.

[6] Engelbrecht, A.P. 2006 Fundamentals of Computational Swarm Intelligence, John Wiley & Sons.

[7] Feng, Y., Teng, G.F., Wang, A.X., and Yao, Y.M. 2008 Chaotic Inertia Weight in Particle Swarm Optimization, Second International Conference on Innovative Computing, Information and Control, pp 475-501, IEEE.

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

[9] Kennedy, J., and Eberhart, R. C. 1995 Particle Swarm Optimization, IEEE International Conference on Neural Networks. pp. 1942-1948.

[10] Kennedy, J., and Eberhart, R.C. 2001 Swarm intelligence, Morgan Kaufmann.

[11] Krohling, A., and Dos Santos Coelho, L. "Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems" , IEEE Transactions on System, Man, Cybernnatics B, vol. 36, no. 6, pp. 1407–1416, Dec. 2006.

[12] Liang, J.J., Qin, A.K., Suganthan,P.N., and Baskar, S. "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions", IEEE Transactions on Evolutionary Computation, vol. 10, no. 3, pp. 281–295, Jun. 2006.

[13] Parsopoulos, K.E., Plagianakos, V.P., Magoulas, G.D., and Vrahatis, M.N. 2001 Stretching Technique for Obtaining Global Minimizers Through Particle Swarm Optimization, Proceedings of Particle Swarm Optimization Workshop, pp. 22–29.

[14] Piatetsky-Shapiro, G. 1991 Discovery, Analysis, and Presentation of Strong Rules, Piatetsky-Shapiro G, Frawley WJ, eds. Knowledge Discovery in Databases. Cambridge, MA: AAAI/MIT Press; 1991.

[15] Ratnaweera, A., Halgamuge, S., and Watson, H. 2003 Particle swarm optimization with self-adaptive acceleration coefficients, Proceedings of first International Conference on Fuzzy Systems and Knowledge Discovery, pp. 264–268.

[16] Ratnaweera, A., Halgamuge, S., and Watson, H. "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients", IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, pp. 240–255, 2004.

[17] Shi,Y., and Eberhart, R.C. 2002 A modified particle swarm optimizer, Evolutionary Computation Proceedings, The International Conference World Congress on Computational Intelligence., IEEE, pp 69–73.

[18] Shi, Y., and Eberhart, R.C. 1999 Empirical study of particle swarm optimization, Proceedings of IEEE Congress on Evolutionary Computation, pp. 1945–1950.

[19] Shi, Y., and Eberhart, R.C. 2001 Fuzzy adaptive particle swarm optimization, Proceedings on IEEE Congress on Evolutionary Computation, vol. 1, pp. 101–106.

[20] Tripathi, P.K., Bandyopadhyay, K.S., and Pal, S.K. 2007 Adaptive multi-objective particle swarm optimization algorithm, Proceedings of IEEE Congress on Evolutionary Computation, pp. 2281–2288.

[21] Weijian Cheng, Jinliang Ding, Weijian Kong, Tianyou Chai, and Joe Qin.S. 2011 An Adaptive Chaotic PSO for Parameter Optimization and Feature Extraction of LS-SVM Based Modelling, American Control Conference, pp 3263-3268.

[22] Yamaguchi, T., and Yasuda, K. 2006 Adaptive particle swarm optimization: Self-coordinating mechanism with updating information, Proceedings of IEEE International Conference on System, Man, Cybernatics, Taipei, Taiwan, pp. 2303–2308.

[23] Yu Wang, Bin Li, Thomas Weise, Jianyu Wang, Bo Yuan, and Qiongjie Tian. "Self-adaptive learning based particle swarm optimization, Information Sciences", Vol.181, pp 4515-4538.

[24] Zhan, Z.H., Zhang, J., Li, Y., and Chung. "Adaptive particle swarm optimization. IEEE Transactions on Systems Man, and Cybernetics — Part B: Cybernetics", Vol.39 (6). pp. 1362-1381. ISSN 0018- 9472, 2009.