International Journal of Applied Operational Research
Vol. 1, No. 2, pp. 37-48, Autumn 2011

Journal homepage: www.ijorlu.ir

Numeric Multi-Objective Rule Mining Using Simulated Annealing Algorithm

M. Nasiri*, L. Sadat Taghavi, B. Minaee

Received: May 22, 2011 **; Accepted:** September 10, 2011

Abstract Association rule mining process can be visualized as a multi-objective problem rather than as a single objective one. Measures like support, confidence and other interestingness criteria which are used for evaluating a rule, can be thought of as different objectives of association rule mining problem. Support count is the number of records, which satisfies all the conditions that exist in the rule. This objective represents the accuracy of the rules extracted from the database. Confidence represents the proportion of records for which the prediction of the rule (or model in the case of a complete classification) is correct, and it is one of the most widely quoted measures of quality, especially in the context of complete classification. Interestingness measures how much interesting the rule is. Using these three measures as the objectives of rule mining problem, this article uses a Simulated Annealing algorithm to extract some useful and interesting rules from any Market-basket type databases. The experimental results show that the algorithm may be suitable for large and noisy datasets but don't stay in local minimum.

Keywords Simulated Annealing, Numeric Association Rule Mining.

Introduction

Association rule mining is one of the important tasks of data mining that intends towards the decision support. Basically, it is the process of finding some relations among the attributes/attribute values of a large database. Inside the large collection of data stored in a database, some kind of relationships among various attributes may exist. Discovery of such relationships can help in taking appropriate decisions. Finding these relationships within a vast amount of data is not a simple job. The process of extracting these relationships is termed as association rule mining. These relationships can be represented as an 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. In this article, we will refer them as A and C, respectively. So, symbolically we can represent this relation as $A \rightarrow C$ and each of such relationship that holds between the attributes of records in a database fulfilling some criteria is termed as an

* **Corresponding Author. (✉)**
E-mail: Nasiri_m@iust.ac.ir (M. Nasiri)

M. Nasiri
IUST university, Tehran, Iran.

L. Sadat Taghavi
Qom university, Qom, Iran.

B. Minaee
IUST university, Tehran, Iran.

association rule. Existing algorithms, try to measure the quality of generated rule through considering only one evaluation criterion, i.e., confidence factor or prediction accuracy. This criterion evaluates a rule depending on the number of occurrences of that rule in the entire database. More the number of occurrences, the better is the rule. Most of the association rule algorithms are based on methods proposed by [1], Apriori [2], SETM [3], Pincer search [4] etc. Neither the rules with numeric attribute nor the rules in the form of I8I10I12→I4I5I9 can be discovered by these methods. We must appoint minimum support and minimum confidence in previous method. This particular makes these methods depended on datasets, and it must be executed several times. We do not require to appointment support and confidence in our method and extract best rules in once executed. Previous methods mine association rule in two stages. First they find frequent itemsets and then extract association rules from frequent itemsets. This paper proposed a method based on simulated annealing algorithm without considering minimum support, confidence and interestingness, and extract the best rule with the best of them. If we don't require using support, we can't use the base method for rule mining, whereas in our method we can unuse support with change fitness. Our method is very flexible on changing fitness, so the user can define any normal multi-objective fitness with support, confidence, etc. and obtain his interesting rules. Moreover, he can define the fitness function so that the order of items is considered based on importance of rules.

The generated rule may have a large number of attributes involved in the rule thereby making it difficult to understand [5]. If the generated rules are not understandable to the user, the user will never use them. Again, since more importance is given to those rules, satisfying number of records, these algorithms may extract some rules from the data that can be easily predicted by the user. It would have been better for the user, if the algorithms can generate some of those rules that are actually hidden inside the data. These algorithms do not give any importance towards the rare events, i.e., interesting rules [6, 7]. Using these three measures-support, interestingness, is measured by the number of attributes involved in the rule and tries to quantify the understandability of the rule, and the confidence, interesting rules can be generated. Hence, the rule-mining problem is not a single objective problem; rather we visualize them as a multi-objective problem. A number of evolutionary algorithms have been developed for searching these rules [8-10]. Moreover, the process is too resource consuming, especially when there is not enough available physical memory for the whole database. A solution to encounter this problem is to use Simulated Annealing algorithm, which reduces both cost and time of rule discovery. Simulated Annealing, Genetic algorithm, colony algorithm, evolutionary algorithm and particle swarm algorithm are instances of single objective association rule mining algorithms. A few of these algorithms have been used for multiple objectives. Kaya proposed genetic clustering method [11]. Hong proposed a cluster-based method for mining generalized fuzzy association rules [12]. Ghun proposed a cluster-based fuzzy genetic mining method for association rules and membership functions [13]. Dehuri et al. proposed a rule mining method using multi objectives called multi objective genetic algorithm (MOGA) [14]. Later, they improved the performance by parallel association rule [15]. Nasiri et al. optimized the previous method using clustering which causes the database to be scanned just once at all [16]. They have proposed a method for rule mining with PSO [17] and multi-objective association rule mining with genetic algorithm without specifying minimum support and minimum confidence [18]. Alatas et al. proposed a multi objective differential evolution algorithm for mining numeric association rules [19]. Later, they proposed another numeric association rule mining method using rough particle swarm

algorithm which had some improvements in performance and precision compared to the previous one [20]. They also proposed another numeric association rule mining method chaos rough particle swarm algorithm [21]. Yan proposed a method based on genetic algorithm without considering minimum support [22]. The method uses an extension of elaborate encoding while relative confidence is the fitness function. A public search is performed based on simulated annealing algorithm. As the method does not use minimum support, a system automation procedure is used instead.

In this lecture, we tried to visualize association rule mining as a multi-objective [23, 24] problem rather than single objective one and tried to solve it using Simulated Annealing algorithm [25]. So, these three measures, confidence, support and interestingness are used as three objectives for rule mining problem. The rest of this article is organized as follows. Section 2 is an overview of Simulated Annealing algorithms. An overview of association rule mining and related work is stated in Section 3. Section 4 covers the details of the proposed work. Implementation and Analysis of results is expressed in Section 5. Finally, Section 6 includes the concluding remarks.

2 Simulated annealing

Simulated Annealing (SA) is motivated by an analogy to annealing in solids. The idea of SA comes from a paper published by Metropolis et al. [25]. Metropolis algorithm simulated the material as a system of particles. The algorithm simulates the cooling process by gradually lowering the temperature of the system until it converges to a steady, frozen state. In 1982, Kirkpatrick et al. [26] took the idea of the Metropolis algorithm and applied it to combinatorial (and other) optimization problems. The Table 1 shows how physical annealing can be mapped to Simulated Annealing. Using these mappings, any combinatorial optimization problem can be converted into an annealing algorithm. SA's major advantage over other methods is an ability to avoid becoming trapped at local minima. The algorithm employs a random search, which not only accepts changes that decrease objective function, f, but also some changes that increase it. The latter are accepted with a probability:

$$p = \exp\ (-\frac{\delta f}{T}), \tag{1}$$

where δf is the difference in f and T is a control parameter.
And,

δ = the change in objective function
T = the current temperature

2.1 Acceptance criteria

In each step of Metropolis algorithm, a particle is given a small random displacement and the resulting change, δf, in the energy of the system is computed. If $\delta f \leq 0$, the displacement is accepted. The case $\delta f > 0$ is treated probabilistically. The probability that the configuration is accepted is given in Eq. (1). A certain number of iterations are carried out at each temperature and then the temperature is decreased. This is repeated until the system freezes into a steady

state. This equation is directly used in simulated annealing. The probability of accepting a worse state is given by the Eq. (2):

$$p = \exp(-\frac{\delta f}{T}) > r \quad (2)$$

(Where r is a random number uniformly distributed between 0 & 1.)

The probability of accepting a worse move is a function of both the temperature of the system and of the change in the objective function. As the temperature of the system decreases, the probability of accepting a worse move is decreased. If the temperature is zero, then only better moves will be accepted.

Table 1 Relationship between physical annealing and simulated annealing

Thermodynamic Simulation	**Combinatorial Optimization**
System States	Solutions Feasible
Energy	Cost
Change of State	Neighboring Solutions
Temperature	Control Parameter
Frozen State	Heuristic Solution

2.2 Cooling schedule

The cooling schedule of a simulated annealing algorithm consists of four components.

2.2.1 Starting temperature

The starting temperature must be high enough to allow a move to almost any neighborhood state. If this is not done, the ending solution will be very close to the starting solution. However, if the temperature starts at too high a value, then the search can move to any neighbor and thus transforms the search (at least in the early stages) into a random search. Effectively, the search will be random until the temperature is cool enough to start acting as a simulated annealing algorithm. A method which is suggested by Dowsland is to rapidly heat the system until a certain proportion of worse solutions are accepted and then slow cooling can start [25].

2.2.2 Final temperature

It is usual to let the temperature decrease until it reaches zero. However, this can make the algorithm run for a lot longer. In practice, it is not necessary to let the Temperature reach zero because as it approaches zero, the chances of accepting a worse move are almost the same as the temperature being equal to zero. Therefore, the stopping criteria can either be a suitably low temperature or when the system is frozen at the current temperature (i.e. no better or worse moves are being accepted).

2.2.3 Temperature decrement

Once we have our starting and stopping temperatures, we need to get from one to the others. Theory states that we should allow enough iteration at each temperature so that the system stabilizes at that temperature. The theory also states that at each temperature, the number of iterations might be exponential to the problem size in order to achieve the system stability. As this is impractical, we need to compromise. We can either do this by performing a large number of iterations at a few temperatures, a small number of iterations at many temperatures or a balance between the two.

2.2.4 Iterations at each temperature

One method is to do a constant number of iterations at each temperature. An alternative is to dynamically change the number of iterations as the algorithm progresses. At lower temperatures, it is important that a large number of iterations are done so that the local optimum can be fully explored. At higher temperatures, the number of iterations can be less.

3 Numeric association rule mining

Recently, some optimization methods for association rule mining have been proposed. The process is too resource consuming, especially when there is not enough available physical memory for the whole database. A solution to encounter this problem is to use simulated annealing algorithm, which reduces both cost and time of rule discovery. Genetic algorithms and other algorithms like ant colony, evolutionary algorithm and particle swarm are instances of single objective association rule mining algorithms. A few of these algorithms have been used for multiple objectives.

4 SA in numeric association rule mining

To represent the possible rules as chromosomes, a suitable encoding/decoding scheme is required. In our scheme each chromosome represents a separate rule. With each numeric attribute we associate three extra tag bits. If these two bits are 00 then the attribute next to these two bits appears in the antecedent part, and if it is 11 then the attribute appears in the consequent part. And the other two combinations, 01 and 10 will indicate the absence of the attribute in either of these parts. The third bit is used for representing the number of domain that the values of attribute belong to it.

In this work, we discritize the domain of numeric value to arbitrary domains. We have some methods for discrete a numeric value. Divide it to equal distance, based on frequency in dataset and clustering the value [4]. We discrete the domain of numeric value to equal distance. For example, if the domain of value are set to [0, 100], it is grouped in pre-determined sets for example [0, 5], [5, 10], …, [95,100], if the value of this attribute in rule is 16, the third bit in chromosome is 3, means that this attribute value is between 15 and 20. So the rule $15 \leq A < 20 \rightarrow 0 \leq B < 5$ will look like 003 111.

In this way we can handle variable length rules with more storage efficiency, adding only an overhead of 3k bits, where k is the number of attributes in the database. The mined rules have to acquire large support, confidence and interestingness. The fitness function is has been shown in Eq. (3)

$$\text{Fitness} = \text{Support} \times \alpha_1 + \text{Confidence} \times \alpha_2 + \text{Interestingness} \times \alpha_3 \quad (3)$$

This fitness function has three parts. The first part is the support of a rule that is statistical significance of an association rule (AR). The second part is the confidence value. The third part is used for the number of attributes in the chromosome. This parameter rewards the shorter rules with a smaller number of attributes. By interestingness measure readability, comprehensibility, and ease of understanding that are important in data mining are increased. It is known that larger rules are more likely to contain redundant or unimportant information, and this can obscure the basic components that make the rule successful and efficiently process. α_1, α_2, and α_3 based on importance for user are specified for parameters and one might increase or decrease the effects of parts of fitness function by means of these parameters. For a user, it may support is very higher than confidence and the user can set α_1=.8 and α_2 =.1. if a new parameter is important for user he(she) can add to fitness.

5 Implementation and experimental result

Pseudo-code of our suggested approach works as follows:
Load a sample of records from the database that fits in the memory.
Generate a random initial rule R
Select the temperature change counter k = 0
Select an initial temperature $T = t_0 \geq 0$
Select a repetition schedule, M_k that defines the number of iterations executed at each temperature, t_k
Repeat
Set repletion counter m = 0
Repeat
Generate a Rule(R') in neighborhood of Rule(R)
Calculate $\Delta_{r,\,R'} = f(R') - f(R)$
If $\Delta_{r,\,R'} \leq 0$, then $R \leftarrow R'$
If $\Delta_{r,\,R'} > 0$, then $R \leftarrow R'$ with probability
$\exp(-(\Delta_{r,\,R'})) / T > \text{random}(0,1)$
$m \leftarrow m + 1$
Until $m = M_k$
$k \leftarrow k + 1$
$t_0 \leftarrow t_0 \times \alpha$
Until stopping criterion is met (number of desired rules is created)
Decode the rules in the final stored population, and get the generated rules.
SA algorithm was evaluated in a numeric database with a size of 1,000 records formed by 4 numeric attributes.

Table 2 Used parameter for SA

Parameters	T	α	No. of rules	No. of attributes	No. of bit changes
Values	3000	0.99	400	4	15

Table 3 Synthetically created sets

Sets
$A_1 \in [1–5] \wedge A_2 \in [15–20]$
$A_1 \in [15–20] \wedge A_3 \in [60–65]$
$A_2 \in [65–70] \wedge A_4 \in [15–20]$
$A_3 \in [80–85] \wedge A_4 \in [80–85]$

The values were uniformly distributed in attributes in such a way that they were grouped in pre-determined sets as shown in Table 3. This distribution of the values was completely arbitrary. Some intervals had small size and others have larger size. Support and confidence values for these sets were 25% and 100%, respectively. Other values outside these sets were distributed in such a way that no other better rules than these rules exist. All of the domains of values were set to [0, 100]. We suppose that, our domains are (twenty pre-determined equal sets with 5 members) [0, 4], [5, 10],.., [95, 100]. Intervals of four attribute is equal. The used parameter values for the experiments have been shown in Table 2. α_1, α_2, and α_3 that have been used in fitness values were selected as 1, 1 and 1 respectively.

Using the appropriate weights for fitness function in ARs mining task, these rules were to be mined. The goal was to find the intervals of each of the built regions more accurately. In Table 4, the ARs found by SA are shown. It has been shown that it found the comprehensible rules that have high synthetically created sets. It is notifyable that the SA is database-independent, since it does not rely upon support/confidence thresholds which are hard to choose for each database. If the support and confidence thresholds have been used and a support threshold that is higher than 25% is selected, no rules will be able to be found according to the values of the attributes in this database. However, it is known that this database contains some accurate and comprehensible rules. SA is able to automatically find all these rules without relying upon the minimum support and the minimum confidence thresholds.

Table 4 AR's found by SA

Rule	Support%	Confidence%	Interestingness%	Records%
$A_3 > \pm @ \$_1 > \pm @$	0.25	0.893	0.5	100
$A_3 > \pm @ \$_4 > \pm @$	0.24	0.889	0.5	
$A_4 > \pm @ \$_3 > \pm @$	0.24	0.96	0.5	
$A_1 > \pm @ \$_2 > \pm @$	0.25	0.862	0.5	
$A_2 > \pm @ \$_1 > \pm @$	0.25	0.962	0.5	
$A_2 > \pm @ \$_4 > \pm @$	0.25	0.862	0.5	

Another experiment has been performed to compare the efficiency of the SA with other algorithms such as RPSOA the Genetic Association Rule Mining (GAR) algorithm proposed in Mata et al.[27] and the work proposed in Alatas and Akin [19] in terms of support, confidence, number of rules and size. For performing this, we evaluated SA algorithm in five public domain databases: Basketball, Bolts, Pollution, Quake and Sleep. These databases are available from Bilkent University Function Approximation Repository. Table 5 shows the number of records and the number of numeric attributes for each database as well as the number of intervals and the parameter values which are used for experiments. We establish the intervals of all numeric attributes of each database manually, before the mining process.

Table 5 Used parameter and database properties for comparison

Database	No. of records	No. of attributes	No. of intervals	No. of bit changes	Temperature	No. or rules
Basketball	96	5	5	5	3000	100
Bolts	40	8	5	5	3000	100
Pollution	60	16	5	5	3000	100
Quake	2178	4	5	5	3000	100
Sleep	62	8	5	5	3000	100

Table 6 Comparisons of the results

Database	No. of Rules %			Size			
	SA	RPSOA	Alatas and Akin (2006)	SA	RPSOA	Alatas and Akin (2006)	GAR
Basketball	32	33.8	33.8	2.3	3.21	3.21	3.38
Bolts	17	39.0	39.0	2.64	5.14	5.14	5.29
Pollution	23	41.2	41.2	2.88	6.46	6.21	7.32
Quake	35	43.8	43.8	2.4	2.22	2.10	2.33
Sleep	29	32.8	32.8	2.56	4.19	4.19	4.21

Table 7 Comparisons of the results

Database	Support (%)				Confidence %		
	SA	RPSOA	Alatas and Akin (2006)	GAR	SA	RPSOA	Alatas and Akin (2006)
Basketball	25	36.44	32.21	36.69	81	60±2.8	60±1.2
Bolts	41	28.48	27.04	25.97	70	64±2.0	65±1.
Pollution	54	43.85	38.95	46.55	80	66±4.7	68±4.8
Quake	36	38.74	36.96	38.65	66	63±2.8	62±5.1
Sleep	40	36.52	37.25	35.91	70	64±2.8	64±2.3

In order to select the number of intervals parameter for each database, several trials of the algorithm is executed, to get the result with the best solution. Because of stochastic characteristic of the SA algorithm, it had fluctuations in different runs. In order to get a better result, the user may execute several trials of the algorithm to get the result with the most efficient solutions. Algorithm was executed ten times and the average values of such execution were presented. To compare the efficiency of the SA with RPSOA algorithm [20],

the experimental comparison in terms of number of rules, support and confidence and size values has been performed. The comparison results have been shown in Table 6 and Table 7.

The value of the column “Support (%)” indicates the mean of support, and the column “Confidence (%)” indicates the mean of confidence, while the value of the column “Size” shows the mean number of attributes contained in the rules. Comparing the SA algorithm with the RPSOA is difficult; because we establish the intervals of numeric attributes (in the SA algorithm) manually before mining process, but the RPSOA [20] discretizes numerical attributes while searching for ARs.

SA algorithm has found rules with high values of support in three out of five databases. The size obtained from the SA algorithm is smaller than the values obtained from the RPSOA in four out of five databases. The confidence values obtained from SA are higher than the values obtained from the RPSOA in five out of five databases. Furthermore, the number of attributes in the rules is small. Thus, the discovered rules within these databases by SA are accurate, readable, and comprehensible.

We tested our algorithm on a large data set about Iranian social station that has 10 numeric attributes and more than one million records. Some attributes are Age, earning etc.

This dataset has noise and we test it on a noisy dataset. In table 8, it shows that the algorithm test with 2%, 4%, 6% and 8% of noise level of data set. We concluded that the noise of 8% in this dataset is good. We set T=100 and 100 iteration in each temperature.

Table 8 Algorithm test with 2%, 4%, 6%, 8% of noise level of data set

Extraction Rule	(%)Support	(%)Confidence
noise level=4%		
$A_4 \in [80-98] \Rightarrow A_3 \in [80-100]$	22	100
$A_3 \in [80-99] \Rightarrow A_4 \in [80-99]$	22	92
$A_4 \in [15-45] \Rightarrow A_2 \in [65-90]$	24	100
$A_2 \in [65-90] \Rightarrow A_4 \in [15-45]$	24	96
$A_3 \in [60-75] \Rightarrow A_1 \in [15-45]$	24	100
$A_1 \in [15-45] \Rightarrow A_3 \in [60-75]$	24	96
$A_2 \in [15-30] \Rightarrow A_1 \in [1-10]$	24	100
$A_1 \in [1-10] \Rightarrow A_2 \in [15-30]$	24	96
noise level= = 6%		
$A_4 \in [80-100] \Rightarrow A_3 \in [80-100]$	5.23	100
$A_3 \in [80-100] \Rightarrow A_4 \in [80-98]$	22	88
$A_4 \in [15-45] \Rightarrow A_2 \in [65-90]$	5.23	100
$A_2 \in [65-90] \Rightarrow A_4 \in [15-45]$	5.23	94
$A_3 \in [60-75] \Rightarrow A_1 \in [15-45]$	5.23	100
$A_1 \in [15-45] \Rightarrow A_3 \in [60-75]$	5.23	94
$A_2 \in [12-30] \Rightarrow A_1 \in [1-10]$	5.23	100
$A_1 \in [1-10] \Rightarrow A_3 \in [15-30]$	5.23	96
noise level= = 8%		
$A_4 \in [80-100] \Rightarrow A_3 \in [80-100]$	22	100
$A_3 \in [80-100] \Rightarrow A_4 \in [80-100]$	23	92
$A_4 \in [15-45] \Rightarrow A_2 \in [65-90]$	23	100
$A_2 \in [65-90] \Rightarrow A_4 \in [15-45]$	23	92
$A_3 \in [60-75] \Rightarrow A_1 \in [15-44]$	23	98
$A_1 \in [15-45] \Rightarrow A_3 \in [60-75]$	23	92
$A_2 \in [15-30] \Rightarrow A_1 \in [1-10]$	23	100
$A_1 \in [1-10] \Rightarrow A_3 \in [15-30]$	23	92

6 Conclusion

Association rule mining is viewed as a multi-objective problem rather than a single objective one as assumed by the most of the existing algorithms. This article uses the SA algorithm to solve the multi-objective rule mining problem using three measured- interesting, support and confidence. We tested our approach only with the numerical valued attributes. For association rule mining within databases that have numeric attributes, the intervals of attributes established manually. Moreover, the SA algorithm has been compared with RPSOA [4], and we have shown that it is more efficient than the RPSOA [5], and the SA algorithm has provided useful extension for practical applications. Our experimental results on datasets with

numeric values show that our proposed technique is useful and helpful for discovering numerical association rules. It is proved that SA converges and doesn't stay in local minimum and; therefore, we hope that our algorithm converges. Whereas PSO is stayed in local minimum and can't find all good association rules. It mines about random initial rule that algorithm must creates. Moreover, our algorithm is useful for noisy and large data which we ran on a noisy dataset containing more than one million records. Automatic discretization of numeric attributes, using these methods for classifier rules and some refinement such as sampling techniques for the present work have been considered as our further work.

References

1. Agrawal, R., Imielinski, T., Swami, A., (1993). Mining association rules between sets of items in large databases. In Proceedings of ACM SIGMOD conference on management of data, 207–216.
2. Agrawal, R., Srikant, R., (1994). Fast algorithms for mining association rules. In Proceedings of the 20th international conference on very large databases, Santiago, Chile.
3. Houtsma, A., Swami, M., (1993). Set-oriented mining of association rules. Research Report.
4. Lin, D. I., Kedem, Z. M., (1998). Pincer-search: An efficient algorithm for discovering the maximal frequent set. In Proceedings of sixth European conference on extending database technology.
5. Fedelis, M. V., (2000). Discovering comprehensible classification rules with a genetic algorithm, in: Proceedings of Congress on Evolutionary Computation.
6. Freitas, A. A., (2002). Data Mining and Knowledge Discovery with Evolutionary Algorithms. Springer-Verlag, New York.
7. Noda, E., (1999). Discovering interesting prediction rules with a genetic algorithm, Proceedings of the Congress on Evolutionary Computation, 1322–1329.
8. Ayad, A. M., (2000). A new algorithm for incremental mining of constrained association rules. Master Thesis, Department of Computer Sciences and Automatic Control, Alexandria University.
9. Hipp, J., (2000). Algorithms for association rule mining: a general survey and comparison, SIGKDD Explorations 2(1).
10. Freitas, A. A., (2003). A survey of evolutionary algorithms for data mining and knowledge discovery. in: A. Ghosh, S. Tsutsui (Eds.), Advances in Evolutionary Computing, Springer-Verlag, New York, 819–845.
11. Nasiri, M., Tagavi, L., Minaei-Bidgoli, B., (2010). Numeric association rule mining using simulated anealing algorithm. Journal of Convergence Information Technology, 5(1), 60–68.
12. Chiu, H. P., Tang, Y. T., Hsieh, K. L., (2008). A cluster-based method for mining generalized fuzzy association rules. IEEE Transactions on Fuzzy Sets and Systems, 16(1), 249–262.
13. Chen, C. H., Hong, T. P., Tseng, Vincent S., (2006). A cluster-based fuzzy-genetic mining approach for association rules and membership functions. In IEEE.
14. Dehuri, B., Ghosh, A., (2004). Multi objective association rule mining using genetic algorithm. Information Sciences, 163, 123–133.
15. Dehuri, G., (2006). Multi-Objective Genetic Algorithm for Assiciation Rule Mining Using A Homogeneos Dedicated Cluster of Work Station. American Journal of Applied Sciences, 3(11), 2086-2095.
16. Hadian, A., Nasiri, M., Minaei-Bidgoli, B., (2009). Clustering based multi-objective rule mining using genetic algorithm. International Journal of Digital Content Technology and its Applications, 4(1), 37–42.
17. Esmaeli, A., Nasiri, M., Minaei, B., Mozayani, N., (2008). A method for association rule mining with PSO based on confodence. In 17th Conference of Electronic (ICEE), 120–124.
18. Qodmanan, H. R., Nasiri, M., Minaei, B., (2011). Multi objective association rule mining with genetic algorithm without specifying minimum support and minimum confidence. Expert Systems with Applications, 288–298.
19. Alatas, G., Akin, E., (2007). Multi-objective differential evolution algorithm for mining numeric association rules. Applied Soft Computing, 8(1), 646–656.
20. Alatas, G., Akin, E., (2008). Rough particle swarm optimization and its applications. Soft Computing, 12, 1205–1218.
21. Alatas, G., Akin, E., (2008). Chaos rough particle swarm optimization and its applications. Information Sciences, 163, 194–203.

22. Tsay, Y. J., Chiang, J. Y., (2005). CBAR: An efficient method for mining association rules. Knowledge-Based Systems, 99–105.
23. Vafaie, H., DeJong, K., (2004). Improving the performance of a rule induction system using genetic algorithms. in: R. Michalski, G. Tecuci (Eds.), Machine Learning-A Multistrategy Approach, Morgan Kaufmann Publishers, San Francisco, 4, 453–470.
24. Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P., (1983). Optimization by Simulated Annealing. Science, 220(4598), 671-680.
25. http://www.cs.nott.ac.uk/~gxk/aim/notes/simulatedannealing.doc.
26. Rutenber, R. A., (1989). Simulated Annealing Algorithms: An Overview, IEEE Circuits and Devices, Jan.19-26.
27. Mata, J., Alvarez, J. L., Riquelme, J. C., (2002). Discovering numeric associationrules via evolutionary algorithm. In: Sixth Pacific-Asia conference on knowledge discovery and data mining PAKDD-02 (LNAI), Taiwan, 2336, 40–51.
28. Davis, L., (Ed.), (1991). Handbook of genetic algorithm. New York: Van Nostrand Reinhold.
29. Eiben, A. E., Smith, J. E., (2003). Introduction to evolutionary computing. Springer.
30. Kayaa, M., Alhajj, R., (2005). Genetic algorithm based framework for mining fuzzy association rules. Fuzzy Sets and Systems, 152(3), 587–601.
31. Yan, X., Zhang, C., Zhang, Sh., (2008). Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support. Expert Systems with Applications, 36(2), 3066–3076.
32. Alatas, B., Akin, E., (2006). An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules. Soft Comput, 10(3), 230–237.