# Mining Frequent Ranges of Numeric Attributes via Ant Colony Optimization for Continuous Domains without Specifying Minimum Support

**Parisa Moslehi[1], Behrouz Minaei Bidgoli[2], Mahdi Nasiri[3], Erfan Nazari Fazel [4]**

**[1] Computer Department, Islamic Azad University South Tehran Branch
Tehran, Tehran, Iran**

**[2] Computer Department, Iran University of science and Technology
Tehran, Tehran, Iran**

**[3] Computer Department, Iran University of science and Technology
Tehran, Tehran, Iran**

**[4] Computer Department, Islamic Azad University South Tehran Branch
Tehran, Tehran, Iran**

**Abstract**

Currently, all search algorithms which use discretization of numeric attributes for numeric association rule mining, work in the way that the original distribution of the numeric attributes will be lost. This issue leads to loss of information, so that the association rules which are generated through this process are not precise and accurate. Based on this fact, algorithms which can natively handle numeric attributes of a dataset would be interesting. In this paper a new approach to finding frequent intervals of numeric attributes is presented using Ant Colony Optimization for Continuous domains ($ACO_R$). The results show that this approach leads to more precise and accurate intervals in comparison with other approaches like discretizing into intervals with equal lengths.

***Keywords-*** *Numeric Association Rule Mining; Preprocessing; Ant Colony Optimization; Data Mining*

## 1. Introduction

Data mining is the most instrumental tool in discovering knowledge from transactions [1][2][3]. Also data mining is known as an integral part of knowledge discovery in databases (KDD). Transactional database refers to the collection of transaction records, which in most cases are sales records. Data mining on transactional database focuses on the mining of association rules, finding the correlation between items in transaction records. The most important application of data mining is discovering association rules. This is one of the most important methods for pattern recognition in unsupervised systems [4]. Most of the association rule algorithms are based on methods proposed by Agrawal in [5] and [6]. The rules with numeric attributes cannot be discovered by these methods. That is why numeric association rule mining algorithms have been proposed. In a numeric association rule, attributes can be Boolean, numeric or categorical.

There has been proposed many numeric association rule mining algorithms. Each of these algorithms use a method to deal with numeric attributes while mining association rules or preprocessing numeric data.
Sirkant and Agrawal in [7], proposed an algorithm for mining association rules in large relational tables containing both quantitative and categorical attributes, by partitioning numeric attributes into a number of intervals.

Fukuda et al in [8], proposed an algorithm for mining optimized association rules for numeric attributes which uses computational geometry for computing optimized ranges.

Miller and Yang in [9], proposed an algorithm for mining association rules over interval data using Birch, an adaptive clustering algorithm.

Lent and Swami in [10], proposed an algorithm for the problem of clustering two-dimensional association rules in large databases. They used a geometric-based algorithm. BitOp for clustering.

Ke et al in [11], proposed an information theoretic approach to quantitative association rule mining. They used

discretizing numeric attributes and constructing a graph based on mutual information of attributes.

David and Yanhong in [12], proposed a fuzzy weighted association rule mining algorithm by transforming numeric and categorical data into fuzzy values.

Aumannn and Lindell in [13], proposed a statistical theory for quantitative association rules, based on the distribution of values of the quantitative attributes.

These approaches except fuzzy sets may have some drawbacks. The first problem is caused by sharp boundary between intervals which is not intuitive with respect to human perception. The algorithms either ignore or over-emphasize the elements near the boundary of the intervals. Furthermore, distinguishing the degree of membership for the interval technique without a priori knowledge is not easy [14]. Similarly, partitioning by means of fuzzy sets is not an easy task because it is hard to determine the most appropriate fuzzy sets for the numeric attribute values [15][16]. Characteristics of numeric attributes are in general unknown and it is unrealistic that the most appropriate fuzzy sets can always be provided by domain experts. That is why some researchers have proposed an evolutionary algorithm for automatically obtaining the fuzzy sets [2] for numeric attributes as a pre-processing

In recent years, the swarm intelligence paradigm received widespread attention in research. Two main algorithms of which that are popular swarm intelligence metaheuristics for data mining, are Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). Swarm Intelligence is based on social behavior that can be observed in nature, such as ant colonies, flocks of birds, fish schools and bee hives, where a number of individuals with limited capabilities are able to come to intelligent solutions for complex problems [17].

An important insight of early research on ants' behavior was that most of the communication among individuals, or between individuals and the environment, is based on the use of chemicals produced by the ants. These chemicals are called pheromones. By sensing pheromone trails foragers can follow the path to food discovered by other ants. This collective trail-laying and trail-following behavior whereby an ant is influenced by a chemical trail left by other ants is the inspiring source of ACO. The first ACO algorithm for tackling combinatorial optimization problems was proposed by Dorigo et al in [18].

Recently, a continuous version of ACO metaheuristic has been proposed by Socha for tackling continuous attributes and solving continuous optimization problems in [19] and [20].

There is not any study which uses $ACO_R$ for preprocessing numeric data and finding frequent ranges of them for data mining. In this paper we describe how we find frequent ranges of numeric attributes via $ACO_R$ as the preprocessing phase of numeric association rule mining process. This leads us to have accurate and exact frequent ranges.

The rest of this paper is organized as follows. In section 2, a brief explanation of Ant Colony optimization for continuous domain ($ACO_R$) is discussed. In section 3, numeric association rule mining is discussed. The proposed algorithm for mining frequent ranges of numeric attributes via $ACO_R$ is presented in section 4. Experimental setup and results are presented in section 5. Finally we conclude with a summary in section 6.

## 2. Ant Colony Optimization for Continuous Domain ($ACO_R$)

Socha in [23] proposed an extension of ACO algorithm to continuous domain for tackling continuous optimization problems. While ACO uses a discrete probability distribution for choosing a solution, $ACO_R$ uses a probability density function (PDF) and samples it. A Gaussian function is used as PDF in $ACO_R$.

In ACO a pheromone table is used to store pheromone information. $ACO_R$ uses a solution archive of size k in order to describe the pheromone distribution over the search space. Considering the solution archive as a matrix, each entry is called $s_j^i$ where i=1,2,…,n is the number of dimensions and j=1,2,…,k is the number of complete solutions to the problem.

First the archive is initialized with k random solutions. These solutions are ranked based on their quality. The weight $\omega_j$ of solution $s_j$ is calculated according to its rank:

$$\omega_j = \frac{1}{qk\sqrt{2\pi}} e^{\frac{-(j-1)^2}{2q^2k^2}} \quad (1)$$

Where q is a parameter of the algorithm: the effect of q is that, if it is small the best-ranked solutions are strongly preferred and when it is large, the probability becomes more uniform [21].

Each ant chooses a solution from the archive probabilistically for building its own solution. The probability of choosing $s_j$ by an ant is:

$$p_j = \frac{\omega_j}{\sum_{r=1}^{k} \omega_r} \quad (2)$$

After choosing a solution $s_j^i$ in the archive, each ant samples a Gaussian function:

$$P(x) = g(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \quad (3)$$

Where μ and σ are the mean and standard deviation of the Gaussian function respectively. If an ant chooses a solution

$s_j^i$ then the value of $s_j^i$ is assigned to μ , and the standard deviation is assigned as follows:

$$\sigma \leftarrow \xi \sum_{r=1}^{k} \frac{|s_r^i - s_j^i|}{k-1} \quad (4)$$

Where ξ is a parameter of the algorithm that has the same effect as the pheromone evaporation parameter in ACO. The higher the value of ξ, the lower the convergence speed of the algorithm [21].

$ACO_R$ aims to optimize a function, called objective function. So the sampled number is given to the objective function to calculate the result of it. After all the ants finished constructing their solutions, the solution archive is ranked, based on the objective function values of the solutions. This results in having the best solution on top of the archive and the worst ones on the bottom of it. Then the m worst solutions are removed from the archive, where m is the number of ants.

## 3. Numeric Association Rule Mining

### 3.1. Frequent Range

A frequent range is an interval of a numeric attribute in which most of the values of the attribute are fallen. It can have either a short length but rather cover remarkable number of values in transactions, or long length and cover quite a few numbers of values of transactions. So there is a trade-off between the length of an interval and the number of transactions which it covers.

### 3.2. Numeric association rule mining

An example of a numeric association rule in an employee database is as follows [14]:
Age ∈ [25, 36] ⋀ Gender=male→Salary ∈ [2000-2400] ⋀ Has-Car=Yes
(Support=4%, Confidence=80%)

In this association rule, "Age ∈ [25, 36] ⋀ Gender=male" is the antecedent of the rule and "Salary ∈ [2000-2400] ⋀ Has-Car=Yes" is the consequent of it. This rule says that, "4 percent (support) of the employees are the men whose ages range between 25 and 36 and their salaries range from 2000 to 2400 and have their own cars". While it can be said that, "80 percent (confidence) of the men between 25 and 36 years old receive salaries from 2000 to 2400 dollars and have their own cars". Here the intervals [25, 36] and [2000-2400] are frequent ranges.

In a transaction database the support and confidence of a rule is calculated by following equation [23]:

$$support, s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N} \; and \quad (5)$$

$$confidence, c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} \quad (6)$$

where N is the total number of transactions. σ(X ∪ Y) and σ(X) is the frequency of occurrence of the itemset X and X ∪ Y respectively, which is called support count.

Support determines how often a rule is satisfied in the transaction, and confidence determines how often items in Y appear in transactions that contain X [22].

## 4. Proposed Work

Here we show how $ACO_R$ is used in order to preprocess continuous data for numeric association rule mining. Since, $ACO_R$ uses a Gaussian function as a PDF, for dealing with continuous variables; we got this idea and used the concept of Gaussian function described in section 2.1, in our process of generating frequent ranges. Based on this concept, each solution member in archive is assumed to be the central point of Gaussian normal distribution. The structure of the solution archive is slightly modified in order to store the standard deviation (σ) of each solution member in the archive, which is used to determine the boundaries of an interval of numeric attributes.

Each numeric attribute is given to $ACO_R$ for finding its frequent ranges, so it is considered as one dimension or column of the solution archive. The structure of the modified solution archive is shown in Fig. 1.

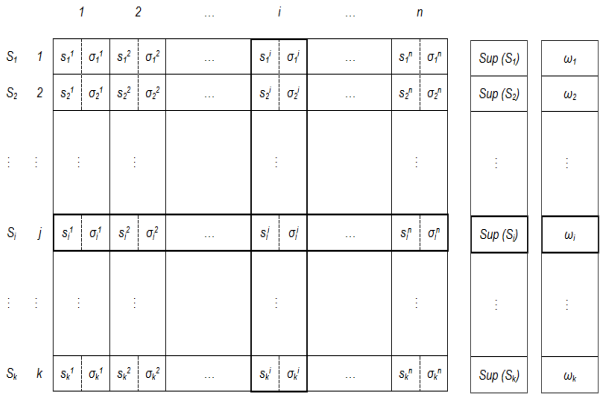


Fig.1 solution archive structure in proposed algorithm

A single dimensional solution archive is used in proposed algorithm and $ACO_R$ is applied to each numeric attribute separately in order to find its frequent ranges. The algorithm consists of some procedures, each of which is described in the following.

### 4.1. Initialization( )

First the data are loaded into memory. Then the prerequisites of the objective function are prepared.

Here the objective function is the function that calculates the support of a numeric attribute. The structure of the objective function will be discussed later.

### 4.2. SolutionArchiveInitialization( )

In this procedure, the solution archive is filled by some uniform random data [21], based on a range that is defined by user in run-time. This range is usually selected in the way that covers the upper bound and lower bound of the numeric attribute value in the database. Furthermore, as this range has a vital effect on solutions of the algorithm, it can be selected in a way to focus on some particular parts of the numeric attribute's range. Then the weights are calculated according to the equation (1) and one solution is selected probabilistically according to equation (2).

After that, the vector of standard deviations is calculated according to the equation (4), considering $s_r^i$ as a solution member and the central point of intervals, and k as the size of the solution archive. Choosing a proper value for ξ, affects the ability of the algorithm to find the correct solutions. Fig. 2 shows the effect of ξ on generated intervals that will be used by $ACO_R$.

In this work, a fixed and predefined value for ξ is used, but if its value changes dynamically, it will result in a more efficient algorithm.

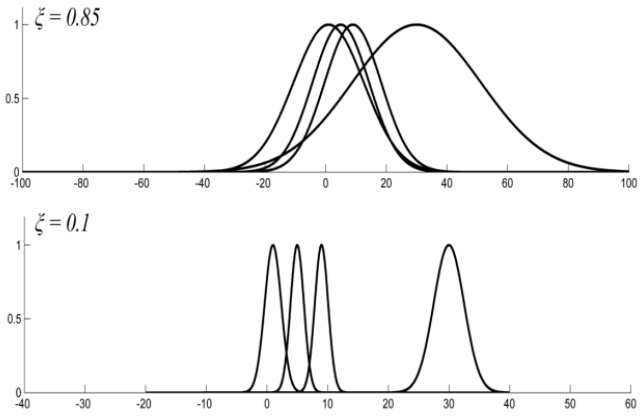


Fig. 2 The impact of ξ on generated intervals

### 4.3. AntBasedSolutionConstruction( )

This procedure consists of two parts:

- Constructing new solutions: In this part, the ants choose a solution according to the weights generated by Gaussian functions. They move through the archive and choose one row of it based on its associated weight (ω), which is calculated through equation (1). Then they construct a new solution by sampling the Gaussian function (g) of the selected solution.

- Calculating the objective function: Since the proposed algorithm aims to find some intervals of a solution which has an optimum support value, the objective function is the function of calculating support of sampled intervals. This function is calculated through the following equation:

$$Objective\ function = \frac{t}{N} \times \frac{1}{l} \quad (7)$$

Where t is the number of transactions in which the value of corresponding attribute falls in the interval, N is the total number of transactions, and l is the length of the interval. The impact of $\frac{1}{l}$ is that shorter intervals are prior to others, so we have more exact and accurate results.

### 4.4. PheromoneUpdate( )

As we said before, in $ACO_R$ pheromone table is replaced by solution archive, so here like $ACO_R$, the pheromone update procedure adds a number of new solutions, each of which is generated by one ant, and eliminates the same number of bad solutions from the archive after ranking its solutions.

### 4.5. DaemonActions( )

Local search heuristics are not used in our proposed algorithm. However, the algorithm performance can improve using these heuristics.

All of these procedures are applied to one numeric attribute of the dataset, in parallel with the others.

## 5. Experimental Setup and Results

In this section, we present the experimental setup for evaluating the accuracy and precision of the results of our proposed approach. In order to do this, the resultant intervals are compared with the intervals produced by dividing the overall range of an attribute in a database into 10 intervals of equal length.

The datasets used for applying our algorithm are Basketball and Bolts which are available from Bilkent University Function Approximation Repository [23]. Table 1 shows the specifications of the both datasets. One characteristic of the proposed algorithm is that it is stochastic so that it has fluctuations in different runs. In order to get a better result, the user may execute several trials of the algorithm to tune the parameters up and get the results with the best solutions. Table 2 and Table 3 show the parameters which resulted in finding the best solutions by $ACO_R$.

Sigma coefficient specifies the boundary of intervals. Since our goal is to find optimum frequent ranges of numeric

attributes, and Basketball dataset contains four numeric attributes and one nominal attribute, which is defined as a class attribute, we do not apply our algorithm to this one. All of the attributes of Bolts dataset are numeric. The algorithm is implemented in C# and run on a personal computer with CPU corei5 Intel and 4G main memory. After several run-time iterations of the algorithm on each attribute of the dataset, four solutions are chosen to compare with the intervals of the same number, resulted from dividing the overall range of an attribute into 10 intervals of the same length. Table 4 shows the results.

Table 1: The dataset specifications

| Basketball | | Bolts | |
|---|---|---|---|
| Number of records | 96 | Number of records | 40 |
| Number of attributes | 5 | Number of attributes | 8 |
| Number of numeric attributes | 4 | Number of numeric attributes | 8 |
| Missing values | 0 | Missing values | 0 |

Table 2: The parameters used to run the algorithm on basketball dataset

| Basketball | | | | | | |
|---|---|---|---|---|---|---|
| Parameter | m | $\xi$ | q | k | n | Sigma coefficient |
| Assists-per-minute | 2 | $5\times10^{-4}$ | 0.1 | 50 | 1 | 100 |
| Height | 2 | $2\times10^{-1}$ | 0.1 | 50 | 1 | 2 |
| Time-played | 2 | $10^{-7}$ | 0.1 | 50 | 1 | $10^{7}$ |
| Age | 2 | $10^{-6}$ | 0.1 | 50 | 1 | $2\times10^{6}$ |

Table 3: The parameters used to run the algorithm on bolts dataset

| Bolts | | | | | | |
|---|---|---|---|---|---|---|
| Parameter | m | $\xi$ | q | k | n | Sigma coefficient |
| Run | 2 | 0.1 | 0.1 | 50 | 1 | 2 |
| Speed1 | 2 | 0.09 | 0.1 | 50 | 1 | 2 |
| Total | 2 | 0.04 | 0.1 | 50 | 1 | 1 |
| Speed2 | 2 | 0.005 | 0.1 | 50 | 1 | 3 |
| Number2 | 2 | 0.009 | 0.1 | 50 | 1 | 2 |
| Sens | 2 | 0.009 | 0.1 | 50 | 1 | 2 |
| Time | 2 | 0.15 | 0.1 | 50 | 1 | 2 |
| T20Bolt | 2 | 0.5 | 0.1 | 50 | 1 | 2 |

Table 4: Frequent Ranges resuletd from $ACO_R$ in comparison with the intervals of equal length

| Basketball | | | | |
|---|---|---|---|---|
| Numeric attribute | Intervals of equal length | Count | $ACO_R$ Intervals | Count |
| Assists-per-minute | (0.07883,0.1082] | 18 | (0.0826,0.1053) | 11 |
| | (0.10826,0.1376] | 14 | (0.1057,0.1099) | 6 |
| | (0.22598,0.1965] | 11 | (0.1925,0.2561) | 26 |
| | (0.28484,0.3142] | 0 | - | - |
| Height | (172.9,177.2] | 0 | - | - |
| | (177.2,181.5] | 5 | (177.2450,183.6633) | 9 |
| | (181.5,185.8] | 22 | (181.8896,194.7184) | 63 |
| | (194.4,198.7] | 21 | (191.4370,197.7300) | 29 |
| Time-played | (13.143,16.206] | 6 | (13.2381,13.4124) | 3 |
| | (19.269,22.332] | 12 | (18.4885,20.1026) | 6 |
| | (22.332,25.395] | 9 | (19.5913,25.0038) | 21 |
| | (34.584,37.647] | 14 | (32.7565,38.8818) | 26 |
| Age | (25,26.5] | 8 | (22.3014,27.3910) | 48 |
| | (26.5,28] | 24 | (27.2348,28.5820) | 12 |
| | (28,29.5] | 4 | (28.3108,31.0539) | 21 |
| | (32.5,34] | 6 | (29.5503,34.9400) | 29 |
| Bolts | | | | |
| Numeric attribute | Intervals of equal length | Count | $ACO_R$ Intervals | Count |
| Run | (12.7,16.6] | 4 | (12.8865,41.0366) | 28 |
| | (−∞,4.9] | 4 | (0.6053,23.6242) | 23 |
| | (24.4,28.3] | 4 | (22.3889,39.5316) | 17 |
| | (16.6,20.5] | 4 | (14.4751,24.6106) | 10 |
| Speed1 | (−∞,2.4] | 16 | (1.9979,2.3316) | 16 |
| | (3.6,4) | 8 | (3.9002,4.2133] | 8 |
| | (5.2,5.6] | 0 | - | - |
| | (5.6,+∞] | 16 | (5.7842,6.1026) | 16 |
| Total | (−∞,12] | 16 | (9.9542,10.0580) | 16 |
| | (18,20] | 8 | (19.9754,20.1676) | 8 |
| | (26,28] | 0 | - | - |
| | (28,+∞] | 16 | (29.8826,30.1286) | 8 |
| Speed2 | (−∞,1.6] | 16 | (1.4908,1.5400) | 16 |
| | (1.9,2] | 8 | (1.9880,2.0171) | 8 |
| | (2.3,2.4] | 0 | - | - |
| | (2.4,+∞] | 16 | (2.4836,2.5155) | 16 |
| Number2 | (−∞,0.2] | 16 | (-0.0165,0.0393) | 16 |
| | (0.8,1] | 8 | (0.9504,1.0037) | 8 |
| | (1.6,1.8] | 0 | - | - |
| | (1.8,+∞] | 16 | (1.9659,2.0082) | 16 |
| Sens | (−∞,1] | 4 | (-0.0999,0.0104) | 4 |
| | (5,6] | 16 | (5.8632,6.0213) | 16 |
| | (7,8] | 8 | (7.9190,8.0544) | 8 |
| | (9,+∞] | 12 | (9.9749,10.1130) | 12 |
| Time | (−∞,16.947] | 18 | (3.8690,20.5400) | 23 |
| | (16.947,29.954] | 10 | (17.1604,19.4943) | 5 |
| | (29.954,42.961] | 4 | (33.1563,40.7584) | 3 |
| | (94.989,107.996] | 3 | (99.7842,11.6971) | 6 |
| T20Bolt | (−∞,15.522] | 14 | (6.6923,27.9836) | 25 |
| | (31.926,40.128) | 3 | (32.6876,35.2207] | 3 |
| | (48.33,56.532] | 0 | - | - |
| | (56.532,64.734] | 1 | (56.2287,90.0640) | 12 |

The results show that, considering the length of the intervals and the count of transactions included by them, the ones resulted from $ACO_R$ are more precise and accurate, and there is a trade-off between the intervals' precision and accuracy (or length and count). Also, since the length of the $ACO_R$ intervals are not equal, and the algorithm considers the support of them as the objective function, it doesn't converge to unpromising solutions.

Another important point is that we don't need to specify the minimum support threshold for each attribute, since $ACO_R$ handles this issue as its objective function.

Furthermore, as there are some overlaps between $ACO_R$ intervals and they are open intervals (not closed or half-bounded) because of using Gaussian function to build them, their boundaries are not sharp and separate so that we have more flexible intervals, and also the distribution of data is reflected. Another point is that sometimes $ACO_R$ frequent ranges cover 2 or more ranges of equal length with low count.

## 6. Conclusion

An extension of Ant colony optimization algorithm to continuous domain called $ACO_R$, is a new metaheuristic for tackling continuous optimization problems. This study

proposed an algorithm which uses $ACO_R$ and the notion of Gaussian functions used by it in order to mine frequent ranges of numeric attributes as a preprocessing step of numeric association rule mining process.

An interval resulted from our algorithm is open and reflects the distribution of the original data. $ACO_R$ finds frequent ranges without any need to specify minimum support threshold. The algorithm is used for mining frequent ranges of a dataset which has numeric attributes and it has given satisfactory results in its application.

Our algorithm seems to provide useful extensions for practical applications since it overcomes the problems which other algorithms proposed for preprocessing numeric data cannot handle.

Mining frequent ranges by $ACO_R$ considering other attributes in an n-dimensional solution archive may be presented as further works.

## References

[1] Chen C.-H., Hong T.-P., and Tseng V.S., *A Cluster-Based Fuzzy-Genetic Mining Approach for Association Rules and Membership Functions*, IEEE International Conference on Fuzzy Systems, pp. 1411 - 1416, 2006.

[2] Kayaa M., Alhajj R., *Genetic algorithm based framework for mining fuzzy association rules*, Fuzzy Sets and Systems , Vol. 152, No. 3, pp. 587-601, 2005.

[3] Tsay Y. J., Chiang J. Y., *CBAR: an efficient method for mining association rules*, Knowledge-Based Systems, Vol. 18, No. 2-3, pp. 99-105, 2005.

[4] Qodmanan H. R., Nasiri M., and Minaei-Bidgoli B., *Multi objective association rule mining with genetic algorithm without specifying minimum support and minimum confidence*, Expert Systems with applications, Vol. 38, No. 1, pp. 288-298, 2010.

[5] Agrawal R., Imielinski T., and Swami, A., *Mining association rules between sets of items in large databases,* In proceedings of ACM SIGMOD conference on management of data, pp. 207-206, 1993.

[6] Agrawal R., Sirkant R., *Fast algorithms for mining association rules,* In proceedings of the 20th international conference on very large databases, Santiago, Chile, 1994.

[7] Srikant R., Agrawal R., *Mining quantitative association rules in large relational tables*, In: Proceedings of ACM SIGMOD international conference on Management of data, Vol. 25, No. 2, pp. 1-12, 1996.

[8] Fukuda T., Yasuhiko M., Sinichi M., Tokuyama T. Mining *optimized association rules for numeric attributes*. In: Proceedings of ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems. New York, pp. 182–191, 1996.

[9] Miller R.J., Yang Y., *Association rules over interval data*, In: Proceedings of ACM SIGMOD international conference on management of data, vol. 29, No. 2, pp. 452–461, 1997.

[10] Lent B., Swami A., and Widom J., *Clustering association rules*, In: Proceedings of IEEE international conference on data engineering, pp. 220-231, 2002.

[11] Ke K., Cheng J., and Ng W., *An information-theoretic approach to quantitative association rule mining*, Journal Knowledge and Information Systems, Vol. 16, No. 2, pp. 112-114, 2008.

[12] David L. O., Yanhong L., *Mining Fuzzy Weighted Association Rules.* In: 40th Annual Hawaii international conference on system, sciences HICSS, pp 53–62, 2007.

[13] Aumann Y., Lindell Y., *A statistical theory for quantitative association rules*, Journal of Intelligent Information Systems, Vol. 20, No. 3, pp.255–283, 2003.

[14] Alatas B., Erhan A., *Rough particle swarm optimization and its applications in data mining,* Soft Computing – A Fusion of Foundations, Methodologies and Applications, Vol.12, No. 12, pp. 1205-1218, 2008.

[15] Alatas B., Arslan A., *Mining of fuzzy association rules with genetic algorithms (in Turkish)*, Gazi University, J Polytech, Vol. 7, No. 4, pp. 269-276, 2004.

[16] Alatas B., Arslan A., *A novel approach based on genetic algorithm and fuzzy logic for mining of association rules (in Turkish)*, Firat University, J Sci Eng, Vol. 17, No. 1, pp. 42–51, 2005.

[17] Martens D., Baesens B., and Fawcett, T., *Editorial Survey: Swarm Intelligence for Data Mining,* Machine Learning, Vol. 82, No. 1, pp. 1-42, 2010.

[18] Dorigo M., Stutzle T., *Ant Colony Optimization,* A Bradford Book, MIT Press, Cambridge, Massachusetts, London, England, 2004.

[19] Socha K., *ACO for Continuous and Mixed-Variable Optimization,* Ant Colony Optimization and Swarm Intelligence, Computer Science, Vol. 3172, pp. 53-61, 2004.

[20] Socha K., Dorigo M., *Ant colony optimization for continuous domains*, European Journal of Operational Research, Vol. 185, No. 3, pp. 1155-1173, 2006.

[21] Socha K., *Ant Colony Optimization For Continuous and Mixed-Variable Domains*, Ph.D. Thesis, Universit´e Libre de Bruxelles, Brussels, Belgium, 2008.

[22] Tan P.-N., Steinbach M., and Kumar V., *Introduction to Data Mining*, Pearson International Edition Pearson Addison Wesley, 2006.

[23] Guvenir H.A., Uysal I., Bilkent University Function Approximation Repository. http://funapp.cs.bilkent.edu.tr, 2000.