# Time Complexity Of A Priori And Evolutionary Algorithm For Numerical Association Rule Mining Optimization

**Imam Tahyudin, haviluddin Haviluddin, Hidetaka Nanbo**

**Abstract**: Some of the solutions for solving numerical Association rule mining problem are by discretization and optimization methods. The popular algorithms of optimization are A priori algorithms, Genetic algorithms (GA) and Particle swarm optimization (PSO). This research has aim to study time complexity of those optimization algorithms. The results show that the time complexity of evolutionary algorithms such as GA and PSO are faster than the time complexity of A priori algorithms.

**Index Terms**: time complexity, numerical association rule mining, a priori, evolutionary algorithm.

## 1. INTRODUCTION

Nowadays the numerical association rule mining problem is an interesting topic that has been studied using various approaches. Among these are conventional methods like a priori and FP growth [1]–[3] discretization approaches like partitioning and combining, clustering and fuzzy [4], [5] by optimization methods like Genetic Algorithms (GA), differential evolution and Particle Swarm Optimization (PSO) [6]–[8]. The PSO method is one of the evolutionary algorithms used for solving the ARM problem [9]. However, this method has the drawback that it may become trapped in local optima when the number of iterations goes to infinite then the particle velocity tends to 0. As such, the PSO does not have the capability to search for the optimal solution [10]. This weakness has been solved by combining PSO with Cauchy distribution [11]. This combination can do the searching process faster than traditional methods. Hence, the aim of this research is to explain the difference in time complexity between a priori methods as traditional method and evolutionary methods like either GA or PSO. This research is arranged as follows: Section 2 describes about association rules mining; Section 3 discusses time complexity of a priori and evolutionary algorithms; and finally, the conclusion is given in Section 4.

## 2 ASSOCIATION RULE MINING

Association Rule Mining (ARM) is one of the methods in data mining which finds the association of some variables in a dataset by using some algorithms in order to obtain the useful information such as pattern or rules [3]. The familiar algorithms of ARM are a priori and FP growth algorithm which effective to seek an optimal rule in sparse data [12]. Sparse data means that each item is relatively infrequent [2]. An a priori algorithm usually requires a minimum support. The item set which greater than support threshold is discovered. Furthermore, it is appropriate for categorical data type like gender or binary form. If the data type is continuous numerical type such as age, weight or length, it should to discretize to interval form or grouping form 1. In fact, this step introduces weaknesses like missing many information and needing more processing time [8], [13]. It is because a priori algorithm need one scan for every data item sets length in one database. Therefore, it makes the process slowly [3], [14], [15]. Many authors proposed some methods to overcome these drawbacks. Among of them are by using a GA approach [12], [16], [17] or the PSO approach [8], [10]. These methods solved the numerical ARM problem without a discretization process and specifying a threshold of minimum support. Using these method, the important information will retain and also the processing time is faster than an a priori algorithm.

- *Imam Tahyudin*[*,1]*, Artificial Intelligence Laboratory Graduate School of Natural Science and Technology, Division of Electrical Engineering and Computer Science, Kanazawa University, Japan. Email: imam@blitz.ec.t.kanazawa-u.ac.jp*
- *Haviluddin Haviluddin*[2]*, Universitas Mulawarman, East Kalimantan, Indonesia.Association for Scientific Computing Electronics and Engineering (ASCEE), Indonesia Section. Email: haviluddin@unmul.ac.id*
- *Hidetaka Nambo, Artificial Intelligence Laboratory Graduate School of Natural Science and Technology, Division of Electrical Engineering and Computer Science, Kanazawa University, Japan. Email: nambo@ec.t.kanazawa-u.ac.jp*

## 3 RESULT AND DISCUSSION

### 3.1 Time Complexity of a priori Algorithms and Evolutionary Algorithms

Difficulty, number of steps is correlates with the computational process. There are some factors which influence to the time complexity of an a priori algorithm. These are the minimum support threshold, the number of items, the number of transactions, the average transaction width, and the generation of frequent 1-itemsets, candidate generation and support counting. These factors will be explained in details below [18].

### 3.2 Minimum Support Threshold

The minimum support threshold often results in more item sets being declared as frequent. This has an adverse effect on the computational complexity of the algorithm because more candidate item sets must be generated and counted. The maximum size of frequent item sets also tends to increase with minimum support thresholds. Accordingly, as the maximum size of the frequent item sets increases, the algorithm will need to make more passes over the data set [18].

### 3.3 Number of Items (Dimensionality)
As the number of items increases, more space will be needed to store the support counts of items. If the number of frequent items also grows with the dimensionality of the data, the computation and I/O costs will increase because of the larger number of candidate item sets generated by the algorithm [18].

### 3.4 Number of transaction
Since the a priori algorithm makes repeated passes over the data set, the run time increase exponentially with a larger number of transactions. But to emphasize it is not a linear increase in processing time [18].

### 3.5 Average transaction width
For dense data sets, the average transaction width can be large. This affects the complexity of the a priori algorithm in two ways. First, the maximum size of frequent item sets tends to increase as the average transaction width increases. As a result, more candidate item sets must be examined during candidate generation and support counting. Second, as the transaction width increases, more item sets are contained in the transaction. This will increase the number of hash tree traversals performed during support counting [18].

### 3.6 Generation of frequent 1-item sets
For each transaction, we need to update the support count for every item present in the transaction. Assuming that w is the average transaction width, this operation requires $O(N_w)$ time, where N is the total number of transactions [18].

### 3.7 Candidate generation
To generate candidate k-item sets, pairs of frequent (k − 1)-item sets are merged to determine whether they have at least k − 2 items in common. Each merging operation requires at most k − 2 equality comparisons. In the best-case scenario, every merging step produces a viable candidate k-item set. In the worst-case scenario, the algorithm must merge every pair of frequent (k−1)-item sets found in the previous iteration [18].

### 3.8 Support counting
Each transaction of length |t| produces $\binom{|t|}{k}$ item sets of size k. This is also the effective number of hash tree traversals performed for each transaction. The cost for support counting is

$$O\left(N \sum_k \binom{\omega}{k} \propto_k\right) \qquad (1)$$

Where ω is the maximum transaction width and ∝_k is the cost for updating the support count of a candidate k-item set in the hash tree [18]. According to the previous researchers a priori based algorithm based is slow because increasing the number of attributes results in an exponential increase of the running time. As depicted in equation 1, the computation complexity of an a priori algorithm follows an exponential distribution. In this equation, d is the number of attributes and N shows the number of transactions or records in a data set [8], [12]. Time complexity = O (Finding frequent item sets) + O (Rule generation)

Time Complexity =

$$O(N * d * 2^{d}) + O\left(\sum_{k=1}^{d-1}\left[\binom{d}{k} \text{x} \sum_{j=1}^{d-k}\binom{d-k}{j}\right]\right) \qquad (2)$$

$$= O(N * d * 2^{d}) + O(3^{d} - 2^{d+1} + 1)$$

$$= O(N * d * 2^{d}) + O(3^{d})$$

$$= O(2^{d+1})$$

Because the order of the time complexity is exponential, the a priori algorithm runs slowly because as many as the number of attributes used increases, the time complexity is longer. On the other hand, the time complexity of evolutionary algorithms follows a quadratic distribution $O(n^2)$. Because of the number of iteration is fixed so that the complexity of the algorithm is equal to O (N x d) or $O(n^2)$. Lobo et al. and Oliveto et al. explained that it diminishes the relevance of a fixed mutation operator as a means of introducing diversity in the population [18]–[20].

## 4 CONCLUSION
In conclusion, we took some points which are first, the problem of numerical ARM can be solved by using conventional method like a priori algorithm or by using evolutionary algorithms such as GA and PSO. Second, the process uses the former method by using discretization step which is one of the weaknesses as it is a slow process while the later method covered this weakness because its process without discretization and minimum support determination process. Third, it is clear that the time complexity of an a priori algorithm is slower than the time complexity of evolutionary algorithm because an a priori algorithm follows the exponential form while the evolutionary algorithm follows a quadratic form. For the future research, this study can be used for developing the recent method which used for solving numerical ARM such as by GA method (MOGAR) or PSO method (MOPAR).

## ACKNOWLEDGMENT
This research supported by various parties. We would like to thank for scholarship program from Kanazawa University, Japan and Ministry of Research and Technology and Directorate of Higher Education (KEMENRISTEKDIKTI), STMIK AMIKOM Purwokerto, Indonesia, Universitas Mulawarman, Indonesia and also Kanazawa University, Japan In addition, we thank for anonymous reviewers who gave input and correction for improving this research support

## REFERENCES
[1] J. Han and Kamber Micheline, Data Mining: Concepts and Techniques (Second Edition). 2015.
[2] A. Y. Zomaya and S. Sakr, Handbook of big data technologies. 2017.
[3] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques. 2016.
[4] M. Kaya, "Multi-objective genetic algorithm based approaches for mining optimized fuzzy association rules," Soft Comput., 2006.
[5] H. WATANABE, Toshihiko TAKAHASHI, "A Study on Quantitative Association Rules Mining Algorithm Based on Clustering Algorithm," Int. J. Biomed. Soft Comput. Hum. Sci. Off. J. Biomed. Fuzzy Syst. Assoc., vol. 16, no. 2, pp. 59–67, 2011.
[6] M. G. AROTARITEI, Dragos NEGOITA, "An optimization of data mining algorithms used in fuzzy association rules," in International Conference on

Knowledge-Based and Intelligent Information and Engineering Systems, 2003, pp. 980–985.

[7] K. Gandhi et al., “An analysis of multi-criteria decision making methods An Analysis of Multi-Criteria Decision Making Methods,” Expert Syst. Appl., 2015.

[8] D. Karaboga and E. Kaya, “Adaptive network based fuzzy inference system (ANFIS) training approaches: a comprehensive survey,” Artificial Intelligence Review, 2018.

[9] R. J. Kuo, M. Gosumolo, and F. E. Zulvia, “Multi-objective particle swarm optimization algorithm using adaptive archive grid for numerical association rule mining,” Neural Computing and Applications, 2017.

[10] C. Li, Y. Liu, A. Zhou, L. Kang, and H. Wang, “A fast particle swarm optimization algorithm with cauchy mutation and natural selection strategy,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2007.

[11] I. Tahyudin and H. Nambo, “The combination of evolutionary algorithm method for numerical association rule mining optimization,” in Advances in Intelligent Systems and Computing, 2017.

[12] C. C. Aggarwal, Data Mining: The Textbook. 2015.

[13] B. Minaei-Bidgoli, R. Barmaki, and M. Nasiri, “Mining numerical association rules via multi-objective genetic algorithms,” Inf. Sci. (Ny)., 2013.

[14] R. J. Bayardo, “Efficiently mining long patterns from databases,” SIGMOD Rec., 1998.

[15] K. Gouda and M. J. Zaki, “Efficiently mining maximal frequent itemsets,” in Proceedings - IEEE International Conference on Data Mining, ICDM, 2001.

[16] H. R. Qodmanan, M. Nasiri, and B. Minaei-Bidgoli, “Multi objective association rule mining with genetic algorithm without specifying minimum support and minimum confidence,” Expert Syst. Appl., 2011.

[17] Haviluddin and R. Alfred, “A genetic-based backpropagation neural network for forecasting in time-series data,” in Proceedings - 2015 International Conference on Science in Information Technology: Big Data Spectrum for Future Information Economy, ICSITech 2015, 2016.

[18] P.-N. Tan, M. Steinbach, and V. Kumar, “Association Analysis: Basic Concepts and Algorithms,” Introd. to Data Min., 2005.

[19] F. G. Lobo, D. E. Goldberg, and M. Pelikan, “Time complexity of genetic algorithms on exponentially scaled problems,” Proc. Genet. Evol. Comput. Conf., 2000.

[20] P. S. Oliveto, J. He, and X. Yao, “Time complexity of evolutionary algorithms for combinatorial optimization: A decade of results,” Int. J. Autom. Comput., 2007.