International Journal on Emerging Technologies **2**(1): 87-92(2011) **ISSN : 0975-8364**

Optimization of Association Rules Mining Apriori Algorithm Based on ACO

Badri Prasad Patel*, Nitesh Gupta*, Rajneesh K. Karn**and Y.K. Rana***

**Department of Computer Science, Technocrats Institute of Technology, Bhopal, (MP)*
**Department of Computer Science, Technocrats Institure of Technology, Bhopal, (MP)*
***Electrical Engineering Department, Radharaman Engineering College, Bhopal, (MP)*
****Director, Research and Development, Radharaman Engineering College, Bhopal, (MP)*

(Received 12 February, 2011 Accepted 10 March, 2011)

ABSTRACT : Discovering association rules between items in a large database of sales transactions has been described as an important database mining problem. Relationships between data are called associations. Association rule mining reveals such interesting relationships. These association rules are presented in a compact form, eliminating redundancy. Elimination of redundancy in the association rule set generated from large item is a challenging task. We show that the quality of rule sets from the Apriori algorithm for association rule mining can be improved by using Ant Colony Optimization (ACO). For this comparison we do a benchmark test using Abalone dataset from UCI machine learning repository.

Keywords : Data Mining, Association rule, Ant Colony Optimization (ACO), Apriori algorithm.

I. INTRODUCTION

Data mining also known as the knowledge discovery in database, it is a forecasting method to extract the hidden knowledge from the large-scale database or the data warehouse [9]. Data mining functions include clustering, classification, prediction, and link analysis (associations) [7]. Data mining helps the marketing analyst to construct customer behaviour profile; such profile can be used as a basis for enabling promotions and inventory management. One of the most important data mining applications is mining association rules. Association rules, first introduced by Agrawal et al. in 1993, association rules mining has received great interest by the data mining community [8]. Association rules show attributes that occur frequently together in a given dataset. These relationships are not based on inherent properties of the data themselves but rather based on co-occurrence of the data items.

Association rule mining is Applicable in various fields such as finance, stock market, medicine, manufacturing, *e*-business, intrusion detection, bioinformatics, etc. A typical and widely used example of association rule mining is market basket analysis, where the goal is to mine patterns describing the customer's purchase behavior. Generally, an association rule is an expression ($X \Rightarrow Y$), where X (antecedent) a set of items and Y (consequent) is is usually a single item. Support and confidence are the two most important quality measures for evaluating the interestingness of rules.

Mining association rules is one of the main contents of data mining research at present and emphasis particularly is finding the relation of different items in database. Apriori is the most famous and basic method in mining association rules [9]. The principle of Apriori algorithm is to find the valuable association rules whose support and confidence must satisfy the minimum support and confidence confirmed by user aforehand.

Apriori employs an iterative approach known as a level-wise search; it may repeatedly scan the dataset for pattern matching and generate a huge number of candidate itemsets in the case of large database [10]. There is an improved algorithm to resolve the disadvantage of the Apriori algorithm. In this paper we implemented an improved algorithm. The rest of the paper is organized as follows. Section 2 introduces Association Rules. Section 3 Apriori Algorithm Section 4 related work and 5 presents our improved algorithm. Section 6 presents the experimental results of the improved algorithm and analyzed its performance. Section 7 summarizes the paper and present future work.

II. ASSOCIATION RULES

In Association Rule mining find rules that will predict the occurrence of an item based on the occurrence of the other items in the transaction [11]. Table shows Market-Basket Transactions

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
2	Milk , Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Association Rules:

$\{\text{Diaper}\} \rightarrow \{\text{Beer}\}$,

$\{\text{Bread, Milk}\} \rightarrow \{\text{Egg, Coke}\}$,

$\{\text{Bread, Beer}\} \rightarrow \{\text{Milk}\}$,

Implication means co-occurrence, not causality.

Association rule is an implication expression of the form $X \rightarrow Y$, where X and Y are itemsets.

Example: $\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\}$

Rule Evaluation

Support (S): Fraction of transactions that contain both X and Y.

Confidence (C): Measures how often items in Y appear in transactions that contain X. Example:

{Milk, Diaper} $\rightarrow$ {Beer}

S = σ({Milk, Diaper, Beer}) / [T]

S = 2/5 S = 0.4

C = σ({Milk, Diaper, Beer} / σ({Milk, Diaper}

S = 2/3 S = 0.67

Itemset: A collection of one or more items. Example {Milk, Diaper, Beer}. K-itemset that contains k-items.

Frequent Itemset: An itemset whose support is greater than or equal to a min_sup threshold.

In association rule mining task from a set of transactions T, the goal of association rule mining is to find all rules having Support >= min_sup threshold and Confidence>= min_conf threshold. There are two phases in the problem of data mining association rules [12].

1. Find all frequent itemsets: i.e. all itemsets that have support s above a predetermined minimum threshold.

2. Generate strong association rules from the frequent itemsets: these association rules must have confidence c above a predetermined minimum threshold.

After the large item sets are identified, the corresponding association rules can be derived in a relatively straightforward manner. Thus the overall performance of mining association rules is determined primarily by the first step. Efficient counting of large itemsets is thus the focus of most association rules mining algorithms.

III. APRIORI ALGORITHM

The Apriori algorithm developed by Agrawal in 1994 is a great achievement in the history of mining association rules. It is by far the most well-known association rule algorithm [7]. The Apriori generates the candidate itemsets by joining the large itemsets of the previous pass and deleting those subsets which are small in the previous pass without considering the transactions in the database. By only considering large itemsets of the previous pass, the number of candidate large itemsets is significantly reduced.

Let *D* the task-relevant data, be a set of database transactions where each transaction T is a set of items, called Tid. Let *I*= {I1, I2,…, Im} be a set of items. An itemset contains *k* items is a k-itemset. If a k-itemset satisfies minimum support (Min_sup) then it is a frequent *k*-itemset, denoted by L_k. Firstly Apriori algorithm generated a set of candidates, which is candidate k-itemsets, denoted by C_k. If the candidate itemsets satisfies minimum support then it is frequent itemsets. Description of the algorithm [10]:

(1) Suppose a minimum support threshold (Min_sup) and a minimum confidence threshold (Min_conf).

(2) Scan the dataset, candidate 1-itemsets, C1, and the number of occurrences of each item is determined. The set of frequent 1-itemsets, L_1, is then determined, consisting of those candidate 1-itemsets in C_1 having minimum support. The algorithm uses $L_1 \infty L_1$ to generate candidate 2-itemsets, C_2.

(3) Scan the dataset again, frequent 2-itemsets, L_2, is then determined, consisting of those candidate 2-itemsets in C_2 having minimum support. Candidate 3-itemsets, C_3 is then generated by $L_2 \infty L_2$.

(4) Repeatedly scan the dataset, compare the support count of each candidate in C_{k-1} with Min_sup, and then generate L_{k-1}, join $L_{k-1} \infty L_{k-1}$ to generate C_k until no more candidate itemsets.

A two-step process is used to find the frequent itemsets: join and prune actions.

(*a*) The join step: To find L_k, C_k is generated by joining L_{k-1} with itself if member l1 and member l2 are joined.

(*b*) The prune step: The members of C_k may not be frequent. A scan of the database to determine the count of each candidate in C_k, and use L_{k-1} to remove a candidate k-itemset from C_k would result in the determination of L_k. Shortcomings of Apriori Algorithm;

1. Scan the database too may time. When the database storing large number of data, time scanning the data is very long, so efficiency is very low [7].

2. Increase the length of frequent itemsets, significant increase in computing time [7].

3. The Apriori algorithm will produce overfull candidates of frequent itemsets, so the algorithm needs scan database frequently when finding frequent itemsets. And it will take more resource and time to accomplish one scanning. So it must be inefficient [10].

In the early 1990s, ant colony optimization (ACO) was introduced by M. Dorigo and colleagues [13]. The ACO is a metaheuristic inspired by the behavior of real ants in their search for the shortest paths to food sources. It looks for optimal solutions by considering both local heuristics and previous knowledge. When applied the proposed algorithm achieved very promising results.

IV. RELATED WORK

A. The Optimization and Improvement of the Apriori Algorithm

This work is published in IEEE Workshop in 2008 on International Symposium on Intelligent Information Technology Application Workshops which has been done by Yiwu Xie, Yutong Li, Chunli Wang, Mingyu Lu.

Through the study of Apriori algorithm they have discover two aspects that affect the efficiency of the algorithm. One is the frequent scanning database; the other is large scale of the candidate itemsets. Therefore, IApriori algorithm is proposed that can reduce the times of scanning database, optimize the join procedure of frequent itemsets generated in order to reduce the size of the candidate itemsets. The results show that the algorithm is better than Apriori algorithm.

In this paper problem is stated as Apriori algorithm has three shortcomings:

First, because there are lots of transactions in the database, computing the frequencies of candidate item sets must scan database frequently and spend much more time. Second, C_k is generated by joining two frequent itemsets that belong to Lk_{-1}. If we can reduce the operating frequencies, we can improve the efficiency of Apriori algorithm. Third, the expending in time and space for frequent itemsets is too much. For example, if we have 104 frequent 1-itemsets, we can get 107 frequent 2-itemsets. So reducing the scale of C_k can improve the efficiency of the Apriori algorithm greatly.

Therefore, it is necessary to delete the useless transactions in the database in order to reduce the scale of database and reduce itemsets generated from C_k by the join procedure. This paper presents an improved Apriori algorithm called IApriori algorithm according to the analysis of Apriori algorithm above.

IApriori algorithm proposed in this paper only has improved from some aspects.

(1) Improvement method of optimizing the join procedure to reduce the size of candidate itemsets. When C_{k+1} is generated from $L_k \infty\ L_k$, each item of the first L_k is joining with every item in the second L_k in classical Apriori algorithm. Since the kth item of the first L_k is the same as the kth item of the second one, the kth item of the first L_k is joining with the k+1th item of the second one, when C_{k+1} is generated from $L_k \infty\ L_k$. Only in this way can we spend less time than before. For example: $L_1= \{I_1, I_2, I_3, I_4\}$, Apriori algorithm needs computing 16(4*4) times but it only needs 6(3 + 2 + 1) times in IApriori algorithm.

(2) Improvement method of reducing the scale of database. Through the process obtaining C_{k+1} from L_k if the size of transaction *t* is less than k, we can say that it is useless for generating C_{k+1}; if transaction *t* does not contain any subset of candidate itemsets C_k, mark the transaction t the deleting tag. IApriori algorithm reduces the scale of database and optimizes the join procedure, so it improves the efficiency of algorithm greatly. Moreover, the scale of L1 must be small as soon as possible to reduce the scale of C_2, C_3 and so on. Make sure the proper minsupport; also we can choose interested items for the frequent itemsets generated.

IApriori algorithm is described as follows.

```
(1) $L_1$ = {large 1-itemsets};
(2) for (k=2; $L_{k-1} \neq \emptyset$ ; k++)
(3) {
(4) $C_k$=apriori-gen(Lk-1); //generate new candidate itemsets
(5) for all transactions $t \in D$ and t.delete=0
(6) {
(7) if t.count<k then // if the size of transaction t is less than k, t is
useless for $C_k$ generated
(8) t.delete=1 //mark t the deleting tag to skip over the record in next database scanning
(9) else
(10) {
(11) Ct=subset(Ck,t); //candidate itemsets contained in transaction t
(12) if $C_t$ = Ø then // if t does not contain any subset of candidate itemsets $C_k$, mark the deleting tag
(13) t.delete=1;
(14) else
(15) {
(16) for all candidates $c \in Ct$
(17) c.count++;
(18) }
(19) }
(20) }
(21) Lk={c? Ck|c.count$\geq$minsup}
(22) }
(23) Answer=_kLk;
```

To implement the improvement, IApriori algorithm is described as follow steps:

(1) Scan the database to get C_1, make sure the proper minsupport to get frequent itemset L_1(k=1).

(2) Generate C_{k+1} though joining two frequent itemsets that belong to L_k. The mth item in the first L_k should join with the m+1th item of the second L_k.

(3) If the size of transaction *t* is less than k, we give transaction *t* to mark the deleting tag; if not, compute C_t that contains subsets of candidate itemsets C_k in transaction t.

(4) Judge whether *t* comprises any subset of C_k; if not, compute the frequencies of C_k. Otherwise, mark t the deleting tag.

(5) Add item of C_k to L_{k+1} if the support of the item is greater than minsupport.

(6) If L_{k+1}= Ø , the algorithm ceases. Otherwise, k=k+1, continue to the second step in circle until L_{k+1}= Ø.

IApriori algorithm performance test. To analyze the relative performance of the IApriori and Apriori algorithms, we use a small part data from real store database stored

10000 transactions. Figure 1 demonstrates the relative performance of these algorithms. Five experiments are carried out accomplished using the same database with different minimum support factors. The experiment is in Windows XP Professional operating system, CPU with Intel (R) 2.93GHz, memory with 512MB, the algorithm language used in C #. Experiments results show that the time needed IApriori algorithm is less than Apriori algorithm under the same support condition. So we can have the conclusion that the proposed algorithm outperforms the Apriori algorithm in computational time.

In this paper authors have discuss the problems exist in scanning database frequently and the large scale of candidate itemsets in Apriori algorithm, present an improved algorithm IApriori algorithm. It not only decrease the times of scanning database but also optimize the process that generates candidate itemsets. Experiments results show that the proposed algorithm outperforms the Apriori algorithm in computational time.

B. The Optimization and Improvement of the Apriori Algorithm

This research is published in 2008 International Workshop on Education Technology and Training & 2008 International Workshop on Geoscience and Remote Sensing by Yiwu Xie, Yutong Li, Chunli Wang, Mingyu Lu.

C. The Optimization of Apriori Algorithm Based on Directed Network

This research work is published in 2009 Third International Symposium on Intelligent Information Technology Application by Yan-hua WANG, Xia FENG, School of Computer Science and Technology, Civil Aviation University of China.

On the basis of the association rule mining and Apriori algorithm this paper proposes an improved algorithm based on the directed network. It reduces consumption and improve the efficiency of algorithms by reduce scanning datasets and improving the efficiency of the pruning step. Finally, this paper gives an experiment to analyze and compare the difference between the two algorithms and the result shows that the improved algorithm promotes the efficiency of computing.

This paper adopts algorithm based on directed network. The improved algorithm only scans the dataset D once, find the directed network G, where the set of vertices V register the itemset, namely V(G) = I; the set of edges E register the edges which are composed by the pair of items in a transaction, E(G)= <Ii, Ij>; the weight of each edge is the support count of each edge which get when scan the dataset D; the set of paths P register the longest path in each transaction and its support count. We get the frequent itemsets by the distance matrix and path searching. By convention, the improved algorithm assumes that items within a transaction or itemset are sorted in lexicographic order. Enter: dataset D, itemset I and the minimum threshold of support Min-sup. Output: all the frequent itemsets L.

Method:

```
(1) Scan the dataset D; // without frequent 1-itemset
(2) G=(V,E), Dn×n=[dij]£¬Path
(3) for each dij {
(4) if wij>=Min_sup then add dij to L2
(5) }
(6) for (i=3;|Li-1|>1;i++){//more than one itemset in Li-1
(7) Li= gen_freq(Path, Li-1, Min_sup); //get the frequent itemsets
(8)}
(9) return L= Ui Li;
Procedure gen_freq (Path, Li-1, Min_sup)   // for join and prune
(1) for each itemset $1_1$? $L_{i-1}$
(2) for each itemset $1_2$? $L_{i-1}$
(3) if $I_1 \infty I_2$ then{
(4) l = ($1_1$[1], $l_1$[2],…, $l_1$[i–1], l2[i–1]); //path
(5) if judge_frequent (l, Path) then
(6) add l to Li;
(7) else delete l ; //prune
(8) }
(9) return Li;
Procedure judge_frequent (l, Path)
(1) if (1Path)^($w_1$>=Min_sup) then //find the path 1 and its support count
(2) return true;
(3) else return false;
```

In order to evaluate the performance of the proposed algorithm, authors have program with MATLAB to realize the two algorithms. Then compared them in the same condition. The test condition is: P4 2.0 CPU; 1G DDR; 80G HD and Windows XP sp2 professional OS. The data source of our experiments is one flight data of B737-300, contains 8352 records.

The typical Apriori algorithm has performance bottleneck in the massive data processing. In this paper, improved algorithm that based on directed network is proposed to mine the association rules from database. The proposed Algorithm takes both efficiency and accuracy into account and it is proved and validated by experiment, so that we can mine association information from massive data faster and better.

D. Genetic Network Programming for Fuzzy Association Rule-Based Classification

This research work is done by Karla Taboada, Member, IEEE, Shingo Mabu, Member, IEEE, Eloy Gonzales, Non-Member, IEEE, Kaoru Shimada, Member, IEEE and Kotaro Hirasawa Member, IEEE. Authors are with the Graduate School of Information, Production and Systems, Waseda

University. Hibikino 2-7,Wakamatsu-ku, Kitakyushushi, Fukuoka 808-0135, Japan. (978-1-4244-2959-2/09,2009 IEEE).

This paper presents a novel classification approach that integrates fuzzy classification rules and Genetic Network Programming (GNP). A fuzzy discretization technique is applied to transform the dataset, particularly for dealing with quantitative attributes. GNP is an evolutionary optimization technique that uses directed graph structures as genes instead of strings and trees of Genetic Algorithms (GA) and Genetic Programming (GP), respectively. This feature contributes to creating quite compact programs and implicitly memorizing past action sequences. Therefore, in the proposed method, taking the GNP's structure into account

(1) Extraction of fuzzy classification rules is done without identifying frequent itemsets used in most Apriori-based data mining algorithms,

(2) Calculation of the support, confidence and χ^2 value is made in order to quantify the significance of the rules to be integrated into the classifier,

(3) Fuzzy membership values are used for fuzzy classification rules extraction,

(4) Fuzzy rules are mined through generations and stored in a general pool. On the other hand, parameters of the membership functions are evolved by non-uniform mutation in order to perform a more global search in the space of candidate membership functions. The performance of our algorithm has been compared with other relevant algorithms and the experimental results have shown the advantages and effectiveness of the proposed model.

In order to evaluate the performance of the proposed classification method, four public-domain data sets from UCI (University of California at Irvine) data set repository have been selected. Algorithms have been developed in a Java-based software development environment. Experiments were performed on a 1.50 GHz Pentium M with 504 MB RAM.

E. Applying Data Mining and Petri Net in Reengineering of Manufacture Management Information System

This Research Work is founded Proceedings of the 2006 IEEE Asia-Pacific Conference on Services Computing (APSCC'06) this work is done by Shaoqin Wu, Chaoan Lai, Yanming Sun, South China University of Technology Guangzhou, Guangdong 510640, China.

This paper studied association rules mining algorithm for mining process model in workflow log of manufacture management information system with XML technology, applied Petri Net in modelling, simulation and optimization of process model, and studied the method for reengineering and optimization of information system based on function units in terms of optimized process model, it was pointed that the difference between actual process model and the process model implied in information system would lead to the decrease of efficiency of information system, and the reengineering and optimization of information system based on actual optimized process model would lead to improvement of efficiency. It was proved with experiments that the association rules mining algorithm put forward in this paper is better than traditional algorithm Apriori in performance.

With experiments, it was proved that the algorithm could mine frequent pattern in XML transaction data without pre-treatment of data, moreover, the efficiency of this algorithm is higher than that of the algorithm in which Apriori and RDBMS were used. To evaluate the efficiency of the proposed method, they had implemented the XML-Apriori algorithm, along with Apriori algorithm, using JAVA on a Pentium III 600 MHz PC with 256MB of available physical memory. The data source was the workflow log, which data were put into either database provided with Microsoft SQL Server and XML document. In this experiment, the efficiency of the XML-Apriori algorithm was compared to the Apriori algorithm. The test database and XML document contain the same 10,000 log records. The performance of XML Apriori algorithm was compared to Apriori algorithm under the various minimum supports.

V. IMPROVED ALGORITHM

The optimized Algorithm for association rule mining using ant colony optimization For a given rule generation task the problem of find a better value of support and confidence in large dataset; given the original data set F of n transaction. Find subset S, which contains of m candidate generation (m < n, S < F), such that the generation of association rule is improved. The generation of rules represent following artificial ants.

1. n candidate that constitute the original dataset F= {f1,f2,f3……..fn}

2. A number of artificial ant to find the support count and candidate value na ants.

3. τ_i the intensity of pheromone trail assured with transition mi which returns the previous knowledge about the important of F_i.

4. For each ant J a list that contains the selected candidate key subset S_j = {S1, S2, S3………Sm}

5. Now here we have used general method for pheromone updating

$$USM_i^{Sj} = (\tau_i)^{\alpha}(LI_i^{Sj})^{\beta} \Sigma (\tau_g)^{\alpha}(LI_g^{Sj})^{\beta}$$

$$g\,?!Sj$$

$$0 \qquad \text{otherwise}$$

Where LI_iS_j is the local importance of Fi given the subset Sj, the parameter α and β control the effect of pheromone.

1. Initialization:

a. Set τ_i = c and $\Delta\tau_i$ = 0, (i=1,2 .....n)

where c is a constant and $\Delta\tau_i$ is the amount of change at pheromone trail generating at support and confidence

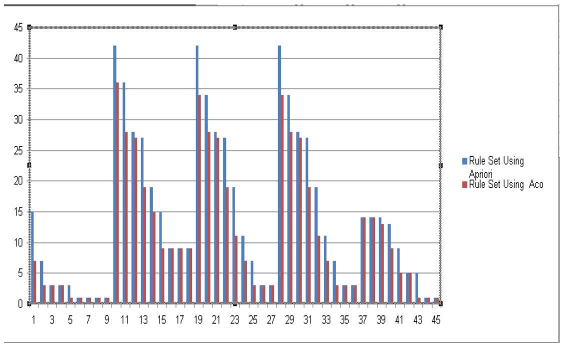
2. Determine maximum number of iteration.

3. Define p where m-p is the number of generated candidate key that each ant will start with in the next step.

```
4. If the first iteration
m=1 Random association subset m candidate to Sj
$m_k$ = {Sj|J? F^σ({Sj})≥N* Min_sup}
5. For $j$ = 1 to $n_a$
        $m = m + 1$
$F$ = Apriori generated ($M_{k-1}$)
6. If iteration are completed the updated pheromone ant value USM.
7. For each transaction n?F do
8. $n\tau_i$ = subset ($m_k$, $\tau_i$)
    end for
9. mk = {c|c? $M_k$^σ(c≥ N* Min_sup)
10. until $m_k = \varnothing$
11. Reset = $Um_k$
12. For each mkset $F_k$, K ≥ 2 do
13. C = {i|i? $F_k$}
14. call ap-gen-rules ($F_k$, $n_a$)
15. end for
16. set the termination at subset.
17. Display optimized strong rule set
```

VI. EXPERIMENTAL RESULT

In order to evaluate the performance of the proposed algorithm, we program with MATLAB to realize the two algorithms. Then we compared them in the same condition. The test condition is: P4 3.00 GHZ CPU; 256MB RAM; 80G HD and Windows XP-SP2 professional OS. The data source of our experiments is Abalone dataset obtained from UCI machine learning repository. The dataset has 4177 samples. It is composed of a discrete attribute and 8 continuous attributes. We can know the comparative result of Apriori algorithm and the improved algorithm. The improved algorithm discovers frequent itemsets and shows much greater efficiency than the Apriori algorithm. Rule set generation and optimization table with Apriori Algorithm and Using ACO.



VII. SUMMARY AND FUTURE WORK

The Apriori algorithm has performance bottleneck in the massive data processing. In this paper, our improved algorithm that based on ACO is proposed to mine the association rules from database. Our algorithm takes both efficiency and accuracy into account and it is proved and validated by experiment, so that we can mine association information from massive data better. The Apriori algorithm scan the database too may time. When the database storing large number of data, time scanning the data is very long, so efficiency is very low. Increase the length of frequent itemsets, significant increase in computing time. The Apriori algorithm will produce overfull candidates of frequent itemsets, so the algorithm needs scan database frequently when finding frequent itemsets. And it will take more resource and time to accomplish one scanning. So it must be inefficient. Therefore we have proposed an algorithm based on ACO to optimize the association rule generated by using Apriori algorithm.

REFERENCES

[1] Ashok Savasere, Edward Omiecinski, Shamkant Navathe," An Efficient Algorithm for Mining Association Rules in Large Databases", *Proceedings of the 21st VLDB Conference Zurich,* Swizerland, (1995).

[2] Ahmed Al-Ani, "Feature Subset Selection Using Ant Colony Optimization", the Faculty of Engineering, University of Technology, Sydney, Australia.

[3] Rakesh Agrawal, Ramakrishnan, "Fast Algorithms for Mining Association Rules", *Proceedings of the 20th VLDB Conference* Santiago, Chile, (1994).

[4] Charu C. Aggarwal, Philip S. Yu, "Online Generation of Association Rules", IBM T. J. Watson Research Center Yorktown Heights, NY 10598.

[5] Mohammed J. Zaki, "Generating Non-Redundant Association Rules", Computer Science Department, Rensselaer Polytechnic Institute, Troy NY 12180.

[6] Stefan Mutter, "Classification using Association Rules", Freiburg im Breisgau, Germany, 11th March (2004).

[7] Margaret H. Dunham, Yongqiao Xiao (Department of Computer Science and Engineering, Southern Methodist University, Dallas, Texas), Le Gruenwald, Zahid Hossain (Department of Computer Science, University of Oklahoma, Norman, OK), A Survey of Association Rule.

[8] Karla Taboada, , Shingo Mabu, Eloy Gonzales ,Kaoru Shimada, Kotaro Hirasawa , "Genetic Network Programming for Fuzzy Association Rule-Based Classification", *IEEE*, (2009).

[9] Yiwu Xie, Yutong Li, Chunli Wang, Mingyu Lu, "The Optimization and Improvement of the Apriori Algorithm", *International Symposium on Intelligent Information Technology Application Workshops, IEEE,* (2008).

[10] Yan-hua Wang, Xia Feng, "The Optimization of Apriori Algorithm Based on Directed Network", *Third International Symposium on Intelligent Information Technology Application, IEEE,* (2009).

[11] Tan, Steinbach, Kumar, "Data Mining Association Analysis: Basic Concepts and Algorithms", Lecture Notes (2004).

[12] K. Ming Leung, "Association Rules", (2007).

[13] Marco Dorigo, Christian Blum, "Ant colony optimization theory: A survey", Elsevier B.V., (2005).