Performance Evaluation of the Distributed Association Rule Mining Algorithms

Ferenc Kovács and Sándor Juhász
Department of Automation and Applied Informatics
Budapest University of Technology and Economics
1111, Budapest Goldmann György tér 3
HUNGARY

Abstract: - One of the best-known problems in data mining is association rule mining. It requires very large computation and I/O traffic capacity, therefore several distributed and parallel association rule mining algorithms have been developed. However the association rule mining problem is NP complete, the execution time estimation of the algorithms can be very important, especially for load balancing or for capacity and resource planning. In this paper a novel execution time prediction method is introduced and evaluated on a PC cluster environment. The average relative error of this model is less than 10 percent.

Key-Words: - Data Mining, Association Rule, Distributed Algorithms, Performance Modelling

1 Introduction

The association rule mining (ARM) is very important task within the area of data mining [1]. Many algorithms were developed to find association rules, but the Apriori is the best-known [2]. The one of the main disadvantage of the Apriori algorithm is its I/O cost. Reference [3] introduces an algorithm for I/O cost cutting, but it has higher memory requirements.

Because of the complexity of the ARM task several parallel algorithms have been developed. The main part of the distributed algorithms is based on the Apriori algorithm. The count distribution (CD) -like algorithms [4] generate the smallest network traffic, because they send only their own counters to the other nodes. The data distribution (DD)- based algorithms [4] generate higher network traffic, because the nodes send not only their local counters, but their own database as well. There are some distributed algorithms, which are not based on Apriori, for instance [5] contributes such an algorithm.

The paper [6] shows a detailed analysis of the Apriori algorithm and it is concluded that the finding the 2 frequent itemsets is the significant part of the execution time. The paper [7] introduces two modifications to avoid the bottlenecks in the Apriori algorithm.

The distributed algorithms were developed and evaluated in a supercomputer environment, but the PC cluster systems have several differences compared to traditional cluster systems. Therefore the contributed distributed association rule mining algorithms do not consider the data distribution cost and they begin the itemset counting after the data distribution phase. But the initial data distribution appears in case of PC cluster systems as the database usually is stored separately from the mining cluster.

Just few distributed association rule mining algorithm have been investigated in PC cluster environment. The paper [8] is concerned with the behaviour of HPA algorithm [9] in PC cluster environment. The node synchronization and possibility of different types of nodes can cause serious performance decrease in PC clusters. The PC cluster-based modification of CD and DD algorithms were discussed in [10]. The algorithm synchronization problem was examined in [11], and an asynchronous distributed algorithm was introduced in [12]. A PC cluster-optimised CD algorithm, which takes the initial data distribution into the consideration, was introduced in [13].

The execution time estimation is an important objective in many applications, and this is especially true for long running, resource intensive, costly data mining algorithms. Performance prediction not only allows estimating the execution time, but also helps to adjust the parameters, to which the execution time is particularly sensitive. It allows balancing the associated costs and the expected benefits of the execution. Good estimation of resource requirements is also important in distributed systems, where the balance of the running time and the amount of processing resources can be fine tuned.

This paper is organized as follows: first of all the widely spread sequential ARM algorithms are described. Afterwards the basic distributed ARM algorithms are summarized. Then the detailed analysis of the count distribution based algorithms is described and a novel performance model of these algorithms is introduced. Finally the conclusion of the algorithms analysis is described.

2 Problem Statement

First we elaborate on some basic concepts of association rules using the formalism presented in [1]. Let $I=\{i_1,i_2,...i_m\}$ be set of literals, called items. Let $D=\{t_1,t_2,...t_n\}$ be set of transactions, where each transaction t is a set of items such that $t \subseteq I$. The itemset X has support s in the transaction set D if s% of transactions contains X, here we denote s= support(X). An association rule is an implication in the form of $X \rightarrow Y$, where $X, Y \subseteq I$, and $X \cap Y = \emptyset$. Each rule has two measures of value, support and confidence. The support of the rule $X \rightarrow Y$ is support($X \cup Y$). The confidence c of the rule $X \rightarrow Y$ in the transaction set D means c% of transactions in D that contain X also contain Y, which can be written in $c = S(X \cup Y)/S(X)$ form. The problem of mining association rules is to find all the rules that satisfy a user specified minimum support and minimum confidence. If support(X) is larger than a user defined minimum support (denoted here min_sup), then the itemset X is called large itemset.

The association rule mining can be decomposed into two subproblems:

- Finding all of the large itemsets
- Generating rules from these large itemsets

The second subproblem is much easier than the first one that is the reason why the ARM algorithms are different from each other only in the method handling the first subproblem.

3 Basic Algorithms

In this section some basic association rule mining algorithms are described. First the Apriori algorithm is introduced because it is the fundamental algorithm of the investigated distributed algorithms. Then three fundamental and referential algorithms are described: count distribution [4], data distribution [4] and hash-based partition algorithm [14]. Table 1 contains the notations that are used in the detailed descriptions of the algorithms.

k itemset	An itemset having k items
L	Set of the frequent itemset
L_i	Set of frequent i itemset
C_i	Set of candidate i itemset (potentially frequent itemset)
|A|	Number of elements in set A

Table 1. Notations in the sequential algorithms

3.1 Apriori algorithm

The Apriori algorithm [2] use the following theorem to reduce the search space: if an itemset is large then all of its subsets are large as well. This means it is possible to generate the potentially large i+1 itemset using large i itemset. Each subsets of candidate i+1 itemset must be large itemset. Hereby it is possible to find all large itemset using database scan repeatedly. During the ith database scan it counts the occurrence of the i itemset and by the end of the pass i, it generates the candidates, which contain i+1 item. Figure 1 shows the pseudo code of the Apriori algorithm. The main disadvantage of this algorithm is the multiple database scan. There are many solutions to reduce the number of database scans [2][3][5][14]

```
$L_1 \leftarrow \{1 \text{ frequent itemsets}\}$
$C_2 \leftarrow GenerateCandidate(L_1)$
$i \leftarrow 2$
$while\ (L_{i-1} \neq \emptyset)$
{
  $foreach\ (t \in D)$
  {
    $IncrementCounter(C_i, t)$
  }
  $L_i \leftarrow \left\{ c \in C_i \middle| \frac{c.counter}{|D|} \geq \min\_supp \right\}$
  $i \leftarrow i+1$
  $C_i \leftarrow GenerateCandidate(L_i)$
}
$L \leftarrow \bigcup_i L_i$
```

Fig. 1. Pseudo Code of the Apriori Algorithm

3.2 Count distribution algorithm

The count distribution algorithm [4] is a fundamental distributed association rule algorithm. The basic idea of this algorithm is that each of the nodes keeps large itemsets and counters of candidates locally, which are related to the whole database. These counters are maintained in accordance with the local dataset and incoming counter values. The nodes locally execute the Apriori algorithm and after reading through the local dataset they broadcast their own counters to the other nodes. Hence each of the nodes can generate new candidates on the basis of the global counter values.

3.2 Data distribution algorithm

Data distribution algorithm [4] offers a solution for the situation, when one of the nodes does not have enough memory for all of the candidates. In this case each of the nodes is responsible for only a part of the candidates. Each of the nodes counts the occurrence of its own candidates in the whole dataset. But all of the nodes have to broadcast their own local database to the other nodes. The disadvantage of the algorithm is that it generates very large network traffic, because it does not use any kind of optimisation to reduce the network traffic.

3.3 Counting While Distributing algorithm

The Counting While Distributing (CWD) [13]algorithm is based on count distribution algorithm but it uses a triangular matrix to find the 2 frequent itemset [7]. The other modification is that the traditional CD algorithm uses all-to-all broadcast mechanism to share own local counters when the nodes finished a database scan but in PC cluster environment the all to all broadcast can be very expensive. Therefore in this algorithm there is a coordinator node that collects the local counter value of the candidates and generates new candidates [10][12].

The traditional distributed association rule mining algorithms do not consider the initial database distribution cost hence they start the itemset counting after the data distribution. Due to the increased searching capability of triangular matrix representation of the 2 itemset it is possible to count the 2 itemsets during the data distribution phase. Therefore after the initial data distribution the coordinator can immediately collect the counters of the 2 itemsets. This solution allows overlapping the initial data distribution and a significant part of the itemsets counting.

The asynchronous communication model can improve the overall efficiency of the cluster. The asynchronous communication model facilitates to overlap the communication and the data processing. During the initial data distribution the coordinator node creates small data fragments from the database and these are sent to the worker nodes. This fragmentation keeps the possibility to increase the efficiency of the data processing. Due to the asynchronous communication the worker nodes can process a data fragment while a new one is arriving through the network communication channel.

4. Performance Model

In order to develop an analytical model capturing the performance of the algorithms, a detailed investigation of its operation is needed. The main features of its execution time behaviour are to be determined. The investigated parameters are independent of each other, thus their effects on the response time can be observed separately. Execution time of each CD-based algorithm can be divided into four independent parts:

- Initial data distribution
- Itemset counting
- Generating new candidates
- Node to node communication

The aim of the performance modelling is to find a model that can estimate the execution time based on the data mining input parameters (data set, minimum support) and number of working nodes. This model independently takes into consideration the effects of data set parameters and the effects of mining and cluster parameters. Model parameters depend on the mining cluster, therefore the model parameters can be different in a different environment. Table 2 contains the notations that are used in the model

4.1 Simulation environment

The algorithms are implemented in standard C++ using the pyramid [15] asynchronous message passing library on Windows XP platform. The simulation took place in the PC laboratory of the department using 11 uniform PCs having an Intel Pentium IV processor of 2.26 GHz, 512 MB RAM, and an Intel 82801DB PRO/100 VE network adapter of 100 Mbits. The nodes were connected through a switching hub of type 3Com SuperStack 4226T.

The datasets used by the algorithms are generated by IBM synthetic data generator [2]. The main parameters of the processed dataset are the number of the transactions (D), the average size of the transactions (T), and the average length of the maximal frequent itemsets (I).The parameter T varies from 10 to 25 by 5, the parameter I was in the range of 6 to 10. Earlier experiences have showed that there is a liner relationship between the response time and number of transaction [16] hence this parameter was fixed to 500.000. The maximal number of the items that can occur in a transaction was 1000. The CD and CWD algorithms were investigated and evaluated. The introduced figures are based on dataset T20I8D500K and the mining algorithm is CWD, but the behaviour of the identified parameters is similar to the introduced ones.

p	Number of nodes
s	Minimum support
C	Const parameter
T	Average size of the market baskets
D	Number of market baskets

Table2. Notations of the model

4.2 Cost of initial data distribution

The initial data distribution is independent of minimum support; it depends on the size of the cluster. This cost can be estimated based on a parallel communication channel model [17]. Figure 2 shows the measured and estimated execution time of the initial data distribution. The parameter of the model can be identified with the least square method. Equation 1 shows the model, the identified parameters and the average relative error of the model.

$$T_{distr}(p) = \frac{C_1}{p} + C_2,$$

$$\left.\begin{array}{l} C_1 = 26.319717 \\ C_2 = 9.517830 \end{array}\right\} \rightarrow \text{Identifed by least square method}$$

$$Avg\,RelErr = 10.44\%$$

Eq. 1. Model and parameters of the data distribution phase

4.3 Cost of itemset counting

Basically the substantial part of the response time is the cost of itemset counting. Figure 3 and 4 show the itemset counting part of the execution time in function of minimum support and processing nodes. It is possible to realise that this time depends on minimum support exponentially. The scale up capability of the itemset counting is satisfactory as it depends on number of nodes in 1/p way. Equation 2 shows the model its parameters and the average relative error of the model.

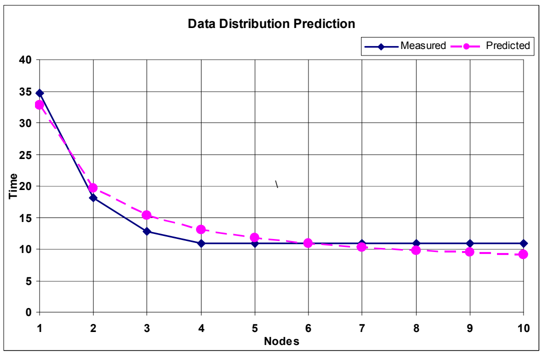


Fig. 2. Prediction of the data distribution cost

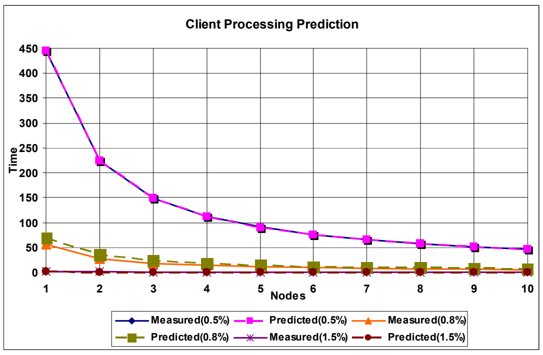


Fig. 3. Itemset counting in function of processing nodes

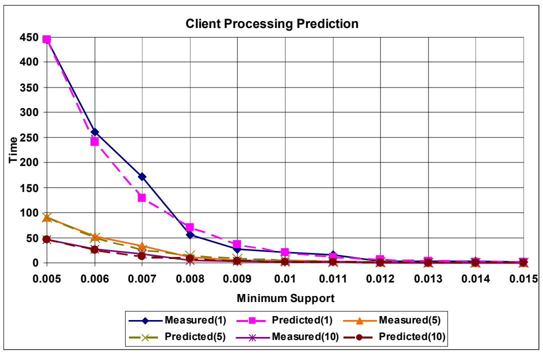


Fig. 4. Itemset counting in function of minimum support

$$T_{client}(s,p) = e^{-C_1 \cdot s} \cdot \left(\frac{C_2}{p} + C_3 \right)$$
$$C_1 = 619.910789$$
$$C_2 = 9841.926376$$
$$C_3 = 37.906202$$
$$Avg\,RelErr = 8.26\%$$

Eq. 2. Model and parameters of the itemset counting phase

4.4 Cost of new candidate generation

The cost of new candidate generation is linearly dependent on the number of working nodes as the nodes has to collect the itemset counters from the other nodes. This cost is decreasing when the minimum support threshold is increasing; experimental results show that the new candidate generation cost can be estimated by a 1/s^3 function. Figure 5 and 6 show the response time and the estimator and equation 3 shows the model and its parameters.

$$T_{coordinator}(p,s) = \left(\frac{C_1}{s^3} + C_2 \right) \cdot (C_3 \cdot p + C_4) =$$
$$= C_1 C_3 \frac{p}{s^3} + \frac{C_1 C_4}{s^3} + C_2 C_3 \cdot p + C_2 C_4$$
$$C_1 C_3 = 2.2733 \cdot 10^{-8}$$
$$C_1 C_4 = 1.2483 \cdot 10^{-7}$$
$$C_1 C_3 = 1.0508 \cdot 10^{-2}$$
$$C_1 C_3 = -8.9439 \cdot 10^{-3}$$
$$Avg\,RelErr = 11.41\%$$

Eq. 3 Model and parameters of the new candidate generation

4.5 Cost of node to node communication

The node to node communication cost depends on the architecture of the ARM algorithm. Due to the network communication capabilities the PC cluster based ARM algorithms have a centralized component, which collects the counter values and generates new candidates. Thus the network communication cost decreases and this cost can be estimated by a linear function.

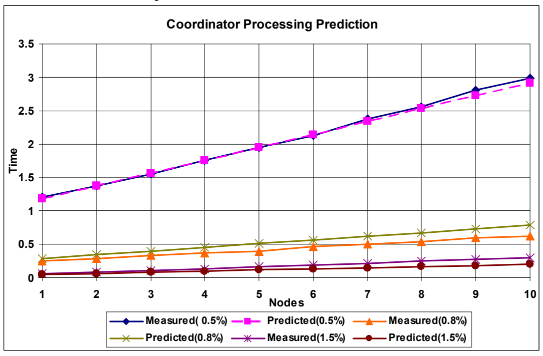


Fig. 5 New candidate generation cost in function of processing node

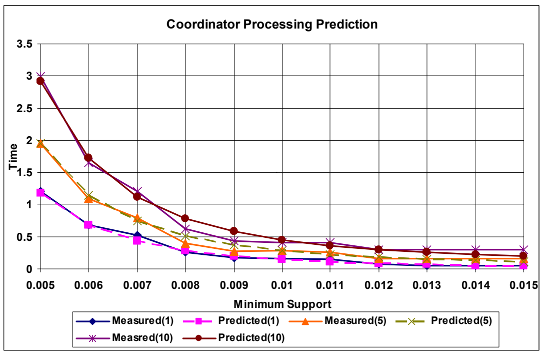


Fig. 6. New candidate generation cost in function of minimum support

Experimental results show that the node to node communication cost is independent of the minimum support threshold. This behaviour is illustrated in Figure 7. Figure 8 shows the predicted and measured communication cost and Equation 4 shows the node to node communication model and its parameters.

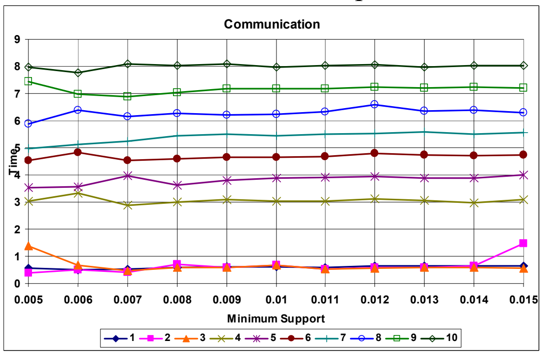


Fig.7. Communication cost in function of minimum support

$$T_{comm}(p) = C_1 \cdot p + C_2$$
$$C_1 = 0.920874$$
$$C_2 = -1.190056$$
$$Avg\,RelErr = 11{,}57\%$$

Eq. 4 Model and parameters of the node to node communication

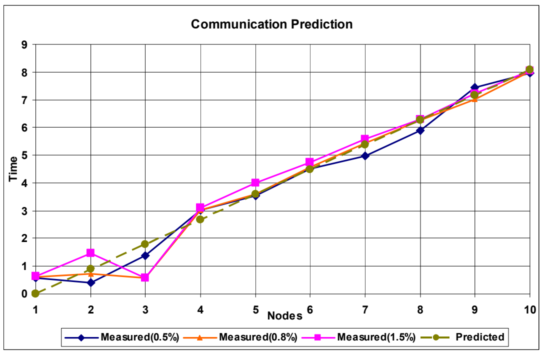


Fig. 8. Communication cost in function of processing nodes

4.6 Execution time prediction

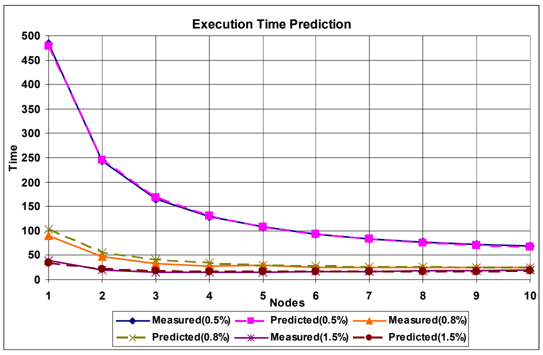
The total execution time can be estimated by the sum of the previously introduced costs. Equation 5 shows the whole model. Figure 9 and 10 show the execution time prediction; Figure 11 shows the error surface of the prediction. The average relative error of the model is 8.28%.

$$T_{exec}(s,p) = T_{distr}(p) + T_{server}(s,p) + T_{client}(s,p) + T_{comm}(p)$$
$$T_{exec}(s,p) = \left(\frac{C_1^{distr}}{p} + C_2^{distr}\right) +$$
$$+\left(\frac{C_1^{server}}{s^2} + C_2^{server}\right)\cdot\left(C_3^{server}\cdot p + C_4^{server}\right) +$$
$$+e^{-C_1^{client}\cdot s}\cdot\left(\frac{C_2^{client}}{p} + C_3^{client}\right) +$$
$$+\left(C_1^{comm}\cdot p + C_2^{comm}\right)$$

Eq. 5 Prediction model for the execution time

4.7 Effects of the dataset parameters

The dataset parameters are independent of the data mining and cluster parameters. Thus their effects can be investigated independently. The model parameters introduced above can be identified on a sample database, and the effects of the dataset parameters can be identified by some other datasets. The substantial part of the response time is finding 2 itemsets therefore the execution time dependence on the average size of the transactions can be modelled with a polynomial of the degree two, except the data distribution cost, as it depends on only the size of the dataset. The parameters of the polynomial functions can be identified with the least square method. Equation 6, 7, 8 and 9 show the modified model. Average relative error of this model is 9,87 percent.



.Fig 9. Execution time prediction in function of processing nodes

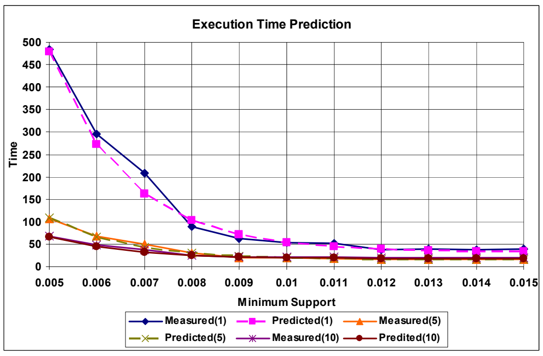


Fig 10. Execution time prediction in function of minimum support

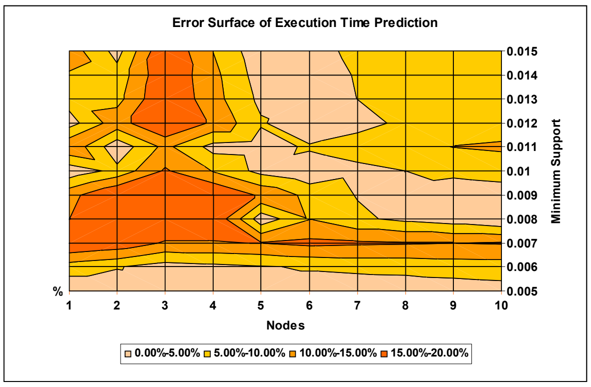


Fig 11. Error surface of the execution time prediction

$$T_{distr}(p,T,D) = f_{distr}(T,D)\cdot g_{distr}(p)$$
$$g_{distr}(p) = \frac{C_1}{p} + C_2$$
$$f_{distr}(T,D) = A_1^{distr}\cdot T\cdot D$$

Eq. 6. Effects of dataset parameters on the data distribution time

$$T_{coordinator}(p,s,T) = f_{coordinator}(T) \cdot g_{coordinator}(p,s)$$
$$g_{coordinator}(p,s) = \left(\frac{C_1}{s^3} + C_2\right) \cdot (C_3 \cdot p + C_4)$$
$$f_{coordinator}(T) = A_1^{coord}{}_1 \cdot T^2 + A_2^{coord} \cdot T + A_3^{coord}$$

Eq. 7. Effects of dataset parameters on the candidate generation time

$$T_{client}(p,s,T,D) = f_{client}(T,D) \cdot g_{client}(p,s)$$
$$g_{client}(p,s) = e^{-C_1 \cdot s}\left(\frac{C_2}{p} + C_3\right)$$
$$f_{client}(T,D) = D \cdot \left(A_1^{client} \cdot T^2 + A_2^{client} \cdot T + A_3^{client}\right)$$

Eq. 8. Effects of dataset parameters on the itemset counting time

$$T_{comm}(p,T) = f_{comm}(T) \cdot g_{comm}(p)$$
$$g_{comm}(p) = C_1 \cdot p + C_2$$
$$f_{comm}(T) = A_1^{comm} \cdot T^2 + A_2^{comm} \cdot T + A_3^{comm}$$

Eq. 9. Effects of dataset parameters on the node to node communication time

5. Conclusion

In this paper basic distributed ARM algorithms have been investigated. A new performance model has been introduced, which keeps the possibility the execution time prediction of these algorithms. A simple model was provided to anticipate the performance of the algorithm, which describes well the behaviour of the data mining algorithm for a wide range of parameters. The main contribution of the paper was predicting the behaviour of a complex probability-based data mining algorithm in a relatively simple, closed numerical form allowing a good estimation of execution times. The model was validated by comparing the calculated and measured performance. Experimental results has showed that the suggested model is reasonably accurate in a wide domain having an average error below 10 percent.

6. Acknowledgement

This work was supported by IBM Hungary and the fund of the Hungarian Academy of Sciences for control research and the Hungarian National Research Fund (grant number: T042741).

References:

[1] R. Agrawal, T. Imielinski, and A. Swami, Mining association rules between sets of items in large databases, *in Proc. of ACM-SIGMOD Conference*, 1993

[2] R. Agrawal and R. Srikant, Fast algorithms for mining association rules, *in Proc. of 20th Very Large Databases Conference, 1994*

[3] J. Han J. Pei and Y. Yin, Mining Frequent Patterns without Candidate Generation, *in Proc. of ACM SIGMOD International. Conference on Management of Data*, 2000

[4] R. Agrwal and J.C. Schafer, Parallel mining of association rules, *in IEEE Trans. Knowledge and Data Engineering*, vol. 8, no 6, 1996

[5] O. R. Zaïne, M. El-Hajj and P. Lu, Fast Parallel Association Rule Mining Without Candidacy Generation, *in Proc. of IEEE International Conference on Data Mining*, 2001

[6] Renáta Iváncsy, Ferenc Kovács and István Vajk, An Analysis of Association Rule Mining Algorithms, *in Proc. of International Conference of ICSC EIS*, 2004

[7] Ferenc Kovács, Renáta Iváncsy and István Vajk, Evaluation of the Serial Association Rule Mining Algorithms, *in Proc. of International Conference of IASTED DBA*, 2004

[8] M. Tamura and M. Kitsuregawa, Dynamic load balancing for parallel association rule mining on heterogeneous PC cluster system, *in Proc. of 25th Very Large Databases Conference*, 1999

[9] T. Shintani and M. Kitsuregawa, Hash based parallel algorithms for mining association rules, in *Proc. of Parallel and Distributed Information Systems Conference*, 1996

[10] T. Shimomura and S. Shibusawa, Performance Evaluation of Distributed Algorithms for Mining Association Rules on Workstation Cluster, *in Proc. of IEEE International Workshops on Parallel Processing (ICPP'00- Workshops)*, 2000

[11] J. Zhang, H. Shi and L. Zheng, A Method and Algorithm of Distributed Mining Association Rules in Synchronism, *in Proc. of IEEE International Conference on Machine Learning and Cybernetics*, 2002

[12] Ferenc Kovács, Renáta Iváncsy and István Vajk, Dynamic Itemset Counting in PC Cluster Based Association Rule Mining, *in Proc. of International Conference of ICSC EIS*, 2004

[13] Sándor Juhász and Ferenc Kovács, A new PC Cluster Based Distributed Association Rule Mining Algorithm, *Scientific Bulletin of Politechnica University of Timisoara Transactions on Automatic Control and Computer Science* Vol. 49 No.2 2004

[14] S.Brin, R. Motawani, J.D. Ullman and S. Tsur, Dynamic Item set counting and implication rules for market basket data, *in Proc. of ACM-SIGMOD Conference*, 1997

[15] S. Juhász et al., The Pyramid Project, Budapest University of Technology and Economics, Department of Automation and Applied Informatics, 2002. http://avalon.aut.bme.hu/~sanyo/piramis

[16] Ferenc Kovács and István Vajk, Performance Analysis of PC Cluster Based Association Rule Mining Algorithms, *in Proc. of IEEE International Conference on Intelligent Systems Design and Aplplication*, 2004

[17] Sándor Juhász, Ferenc Kovács and Hassan Charaf, Asynchronous Communication Model for Cluster Systems, *in Proc. of IASTED International Conference on Parallel and Distributed Computing and Networks*, 2004