American Journal of Applied Sciences 3 (11): 2086-2095, 2006
ISSN 1546-9239


# Multi-objective Genetic Algorithm for Association Rule Mining Using a Homogeneous Dedicated Cluster of Workstations

[1]S. Dehuri, [2]A. K. Jagadev, [3]A. Ghosh and [4]R. Mall
[1]P.G. Department of Information and Communication Technology, Fakir Mohan University, India
[2]Department of Computer Science and Engineering, Krupajal Engineering College
Bhubaneswar-751002, India
[3]Machine Intelligence Unit and Center for Soft Computing Research, Indian Statistical Institute, India
[4]Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, India

**Abstract:** This study presents a fast and scalable multi-objective association rule mining technique using genetic algorithm from large database. The objective functions such as *confidence factor*, *comprehensibility* and *interestingness* can be thought of as different objectives of our association rule-mining problem and is treated as the basic input to the genetic algorithm. The outcomes of our algorithm are the set of non-dominated solutions. However, in data mining the quantity of data is growing rapidly both in size and dimensions. Furthermore, the multi-objective genetic algorithm (MOGA) tends to be slow in comparison with most classical rule mining methods. Hence, to overcome these difficulties we propose a fast and scalability technique using the inherent parallel processing nature of genetic algorithm and a homogeneous dedicated network of workstations (NOWs). Our algorithm exploit both data and control parallelism by distributing the data being mined and the population of individuals across all available processors. The experimental result shows that the algorithm has been found suitable for large database with an encouraging speed up.

**Keywords**: Data mining, association rule mining, network of workstations, MOGA Parallel MOGA

## INTRODUCTION

Association rule mining is an important problem in the rapidly growing field called data mining and knowledge discovery in databases (KDD)[1]. The task of association rule mining is to mine a set of highly correlated attributes/features shared among a large number of records in a given database. For example, consider the sales database of a bookstore, where the records represent customers and the attributes represent books. The mined patterns are the set of books most frequently bought together by the customer. An example could be that, 60% of the people who buy Design and Analysis of Algorithms also buy Data Structure. The store can use this knowledge for promotions, self-placement etc. There are many application areas for association rule mining techniques, which include catalog design, store layout, customer segmentation, telecommunication alarm diagnosis and so on.

The task of mining all frequent associations in very large datasets is quite challenging. The search space is exponential in the number of attributes and with millions of records of dataset. However, most current approaches are iterative in nature, requiring multiple database scans, which is clearly an expensive solution. Some of the methods, especially those using some form of sampling can be sensitive to the data skew, which can adversely affect performance. Furthermore, most approaches use very complicated internal data structures, which have poor locality and add additional space and computation overheads. Although a number of parallel algorithms have already been developed for scalability but these algorithms have their limitations. In this work, we tried to visualize association rule mining as a multi-objective problem rather than as a single objective one. The objective functions like confidence factor[2]; comprehensibility[3] and interestingness[4] can be thought of as different criterion of association rule mining problem. Confidence factor is defined as the ratio between the samples satisfies all the conditions present in the rule and the samples satisfies the conditions present in the antecedent part of the rule. This objective gives the confidence/strengthen of the rules extracted from the database. Comprehensibility is measured by the number of attributes involved in the rule and tries to quantify the understandability of the rule. Interestingness measures how much interesting the rule is?

These three objectives is used in our rule-mining problem. This article uses a parallel multi-objective genetic algorithm (PMOGA) to extract some useful and

**Corresponding Author:** S. Dehuri, P.G. Department of Information and Communication Technology, Fakir Mohan University, India

interesting rules from any market-basket type database. Since MOGA tend to be slow, in comparison with most rule generation methods, the design of parallel MOGA for association rule mining is an important research area[5,6]. Recently there has been considerable research in designing fast algorithm for this task, but none are considering these three objectives simultaneously.

**A brief survey on non-parallel and parallel association rule mining:** This portion is divided into two parts. Part 1 provides a brief survey on non-parallel association rule mining and their limitation. In addition, the requirement of multi-objective genetic algorithm for solving association rule mining problem is given. In part 2 we have discussed a brief overview of parallel association rule mining algorithms.

**Non-parallel association rule mining:** The existing algorithms reported in the literature for mining association rules are based on the approach suggested by Agrawal *et al*.[7,8]. Apriori[8], SET-oriented Mining of association rules (SETM)[9], mining association rules between sets of items in large databases (AIS)[8], Princer search[10], Dyanamic Itemset Counting (DIC)[11] etc. are some of the popular algorithms based on this approach. These algorithms work on a binary database, termed as market basket database. On preparing the market basket database, every record of the original database is represented as a binary record where the fields are defined by a unique value of each attribute in the original database. The fields of this binary database are often termed as an item. For a database having a huge number of attributes and each attribute containing a lot of distinct values, the total number of items will be huge. Storing of this binary database, to be used by the rule mining algorithms, is one of the limitations of the existing algorithms.

Another aspect of these algorithms is that they work in two phases[7]. The first phase is for frequent item set generation. Frequent item-sets are detected from all possible item sets by using a measure called support count (SUP) and a user defined parameter called minimum support. Support count of an item set is defined by the number of records in the database that contains all the items of that set. If the value of minimum support is too high, then less number of rules may be generated. Similarly, if the value is too small, a huge number of rules may be generated. Selecting better rules from them may be another problem.

After detecting the frequent item-sets in the first phase, the second phase generates the rules using another user-defined parameter called minimum confidence (which again affects the generation of rules).

Another limitation of these algorithms is the encoding scheme where separate symbols are used for each possible value of an attribute. This encoding scheme may be suitable for encoding the categorical valued attributes, but not for encoding the numerical valued attributes as they may have different values in every record. To avoid this situation, some ranges of values may be defined. For each range of values an item is defined. This approach is also not suitable for all situations. Defining the ranges will create yet another problem, as the range of different attributes may be different. Apart from these, another problem of these algorithms is that while generating the rules, the orders of the items also play an important role[12].

Existing algorithms, try to measure the quality of generated rule by considering one evaluation criterion, i.e., confidence factor or predictive accuracy. This criterion evaluates the rule depending on the number of occurrence of the rule in the entire database. More the number of occurrences better is the rule. The generated rule may have a large number of attributes involved in the rule thereby making it difficult to understand[13]. If the generated rules are not understandable to the user, the user will never use them. Again, since more importance is given to those rules, satisfying number of records, these algorithms may extract some rules from the data that can be easily predicted by the user. It would have been better for the user, if the algorithms can generate some of those rules that are actually hidden inside the data. These algorithms do not give any importance towards the rare events, i.e., interesting rules[4,14].

Keeping these limitations of existing algorithms in mind, we are motivated to use MOGA for association rule mining problem. Section 3 provides how MOGA can helps to generates association rule. However, MOGA itself tends to be slow and as the data size is growing and hence the computation of fitness is very expensive, so we expect parallelism is the technique to overcome the sequential bottleneck of MOGA based association rule mining method and provide scalability to massive data sets and improving response time.

**Parallel association rule mining:** Andreas Mueller[15] proposed some of the first parallel association rule mining methods, built on the top of his sequential methods, which were based on apriori and partition. Partitioned Parallel Association Rules (PPAR) is based on Spear. In fact, PPAR is the parallelization suggested, but not implemented, by Partition's authors, with the exception that PPAR uses the horizontal data format. The authors reported experiments on a 16-node IBM SP2 DMM showed that PEAR always outperformed PPAR.

The parallel data mining (PDM) algorithm by Park *et al.*[16] is based on DHP. Park and his colleagues presented only simulation results on an IBM-SP2-type distributed-memory machine, so assessing the practical impact of their optimizations is difficult.

Many parallel algorithms use Apriori as the base method, because of its success in the sequential setting. Agrawal and Shafer[17], from the group that developed

Apriori, have proposed three parallel algorithms. Their target machine was a 32-node IBM SP2 DMM. Independently, Shintani and Kitsuregawa[18] proposed four Apriori based parallel algorithms, which are very similar to the Rakesh Agrawal and John Shafer's three parallel algorithms.

Han *et al.*[19] have proposed two ARM methods based on data distribution. They observe that data distribution uses an expensive all-to-all broadcast to send local database portions to every other processor. Furthermore, although data distribution divides the candidates equally among the processors. It fails to divide the work done on each transaction. That is, it still generates a subset of the transaction and determines whether the hash tree contains that subset. Similarly, in intelligent data distribution, Han and his colleagues use a linear-time, ring based, all-to- all broadcast for communication. Second, they switch to count distribution once the candidates fit in memory. Third, instead of a round-robin candidate partitioning, they perform a single-item, prefix based partitioning. Before processing a transaction, they make sure that it contains the relevant prefixes. If not, the transaction can be discarded. The entire database is still communicated, but a transaction might not be processed if it does not contain relevant items. The hybrid distribution combines count distribution and intelligent data distribution. It partitions the 'P' processors into 'G' equal-sized groups, where each group is considered a super processor.

David Cheung and his colleagues proposed the Fast Distributed Mining (FDM) algorithm for ARM. The main difference between parallel and distributed data mining is the inter connection network latency and bandwidth. In distributed mining, we assume that the network is much slower. Apart from this distinction, the difference between the two is becoming blurred. For a slow network, any variants of data distribution, which essentially communicate the entire database in each iteration, are not practical, given the communication costs. Because count distribution has the lowest communication cost, it is an ideal base method to build upon in a distributed environment. David Cheung and Yongqiao Xiao recently proposed a parallel version of FDM, called Fast Parallel Mining.

Zaki *et al.*[20] proposed four algorithms-*ParEclat*, *ParMaxEclat*, *ParClique* and *ParMaxClique*-that target hierarchical system. All four are based on their sequential counterparts. The four algorithms differ depending on the decomposition and search strategy used. *PareClat* and *ParMaxEclat* use prefix based classes, but they use bottom up and hybrid search, respectively. They have experimented on a 32-processor Digital Alpha cluster, with eight four-way SMP hosts connected by the fast Digital Memory Channel network. Comparisons with a hierarchical implementation of count distribution/CCPD showed orders of magnitude improvements of *ParMaxClique* over count distribution.

Existing parallel algorithms, try to measure the quality of generated rule by considering one evaluation criterion. Since because all are based on apriori and its variants. So, for better scalability and viewing it as a multi-objective problem parallel MOGA based rule mining is the natural solution.

**MOGA for association rule mining:** As the association rule-mining algorithm involves many criteria like comprehensibility, confidence factor and interestingness[21], therefore we treated it as a multi-objective problem rather than single objective one. A typical example, shown in Fig. 1, where one wants to maximize both the confidence factor and comprehensibility of an association rule. To cope with this multi-objective problem one can reviews three different approaches, namely: i) weighted sum approach ii) the lexicographical approach, where the objectives are ranked in order of priority and iii) the Pareto approach which consists of as many non-dominated solutions as possible and returning the set of Pareto front to the user. One can conclude that the weighted sum approach-which is so far the most frequently used in the data mining literature-is to a large extent an ad-hoc approach for multi-objective optimization, whereas the lexicographic and the Pareto approach are more principled approaches and therefore deserve more attention from the data mining community. These approaches are discussed later.

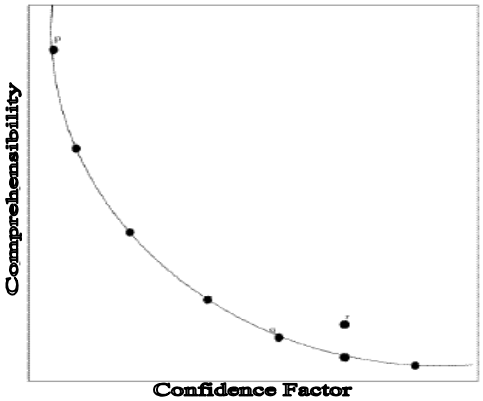


Fig. 1: Trade-off between bi-objectives

**Approaches for solving multi-objective problems:** The three broad categories to cope with multi-objective problem is as follows:

**Weighted sum approach:** Transforming a multi-objective problem into a single objective problem by far the most commonly used approach in data mining literature. Normally, this can be done by a weighted sum of objective functions. That is the fitness value 'F' of a given candidate rule is typically measured by the

formula: $F = w_1 \cdot f_1 + w_2 \cdot f_2 + ..... + w_n \cdot f_n$ where $w_i$, $i = 1,2,.....,n$ denotes the weight assigned to criteria $f_i$ and $n$ is the number of evaluation criteria.

The strength of this method is its simplicity and ease of use. However, it has the drawbacks that, the setting of the weights in these formulas is ad-hoc, either based on a somewhat vague intuition of the user about the relative importance of different quality criteria or in trial and error experimentation with different weight values (which is mostly a difficult aspect of data mining). Hence the values of these weights can be determined empirically.

Another problem with these weights is that, once a formula with precise values of weights has been defined and given to a data mining algorithm, the data mining algorithm will be effectively trying to find the best rule for that particular settings of weights, missing the opportunity to find other rules that might be actually more interesting to the user, representing a better trade-off between different quality criteria. In particular, weighted formulas involving a linear combination of different quality criteria have the limitation that they cannot find solutions in a concave region of the Pareto front.

**Lexicographic approach:** The basic idea of this approach is to assign different priorities to different objectives and then focus on optimizing the objectives in their order of priority. Hence, when two or more candidate rules are compared with each other to choose the best one, the first thing to do is to compare their performance measure for the highest priority objective. If one candidate rule is significantly better than the other with respect to that objective, the former is chosen. Otherwise the performance measure of the two candidate models is compared with respect to the next highest objective. The process is repeated until one finds a clear winner or until one has used all the criteria. In the latter case, if there was no clear winner, one can simply select the model optimizing the highest priority objective.

The lexicographic approach has important advantage over the weighted sum approach: the former avoids the problem of mixing non-commensurable criteria in the same formula. Indeed, the lexicographic approach treats each of the criteria separately, recognizing that each criterion measures a different aspect of quality of a candidate solution. As a result, the lexicographic approach avoids the drawbacks associated with the weighted sum approach such as the problem of fixing weights. In addition, although the lexicographic approach is somewhat more complex than the weighted-sum approach, the former can still be considered conceptually simple and easy to use.

The lexicographic approach usually requires one to specify a tolerance threshold for each criterion. It is not trivial how to specify these thresholds in an unbiased way. A common approach is to use a statistics oriented procedure, e.g. standard deviation-based thresholds, which allow us to reject a null hypothesis of insignificant difference between two objective values with a certain degree of confidence. This specification still has a certain degree of arbitrariness, since any high-value such as 95% or 99% could be used. Of course one can always ask the user to specify the thresholds or any other parameter, but this introduces some arbitrariness and subjectiveness in the lexicographic approach- analogous to the usually arbitrary, subjective specification of weights for different criteria in the weighted formula approach.

Hence after analyzing the strength and weakness of both these methods no one can as much suitable for our rule-mining problem associated with multiple objectives. Therefore we need an alternative method called multi-objective genetic algorithm based on Pareto approach.

**Pareto approach:** The basic idea of Pareto approach is that, instead of transforming a multi-objective problem into a single objective problem and then solving it by genetic algorithm, one should use multi-objective genetic algorithm directly. Adapt the algorithm to the problem being solved, rather than the other way around. In any case, this intuition needs to be presented in a more formal terms, which is defined in the following.

**Pareto dominance:** A solution $x_1$ is said to dominate a solution $x_2$ iff $x_1$ is strictly better than $x_2$ with respect to at least one of the criteria (Objectives) being optimized and $x_1$ is not worse than $x_2$ with respect to all the criteria being optimized.

Using the Pareto dominance, solutions are compared against each other, i.e. a solution is dominant over another only if it has better performance in at least one criterion and non-inferior performance in all criteria. A solution is said to be Pareto optimal if it cannot be dominated by any other solution in the search space. In complex search spaces, wherein exhaustive search is infeasible, it is very difficult to guarantee Pareto optimality. Therefore instead of the true set of optimal solutions (Pareto set), one usually aims to derive a set of non-dominated solutions with objective values as close as possible to the objective values (Pareto front) of the Pareto set.

**Association rule mining using pareto approach:** Association rule can be represented as an *IF A THEN C* statement. The only restriction here is that the two parts should not have a common attribute, i.e., $A \cap C = \phi$. To solve this kind of mining problem by multiobjective genetic algorithm, the first task is to represent the possible rules as individuals known as individual representation. Second task is to define the fitness function and then genetic materials.

**Individual representation:** There are two basic approaches to represent the rules, named as Pittsburgh and Michigan. In the Pittsburgh approach each chromosomes represents a set of rules and this approach is more suitable for classification rule mining[22]; as we do not have to decode the consequent part and the length of the chromosome limits the number of rules generated. The other approach is called Michigan approach where each chromosome represents a separate rule. A modified approach is currently proposed by Ghosh *et al.*[23]. In this approach each attribute is tagged with two bits and is illustrated in Fig. 2. If these two bits are 00 then attribute next to these two bits appears in the antecedent part and if it is 11 the attribute appears in the consequent part. And the other two combinations, 01 and 10 will indicate the absence of the attributes in either of these parts. For instance the rule $ACF \rightarrow BE$ is represented in the following form.

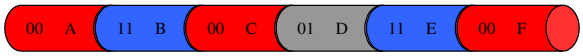


Fig. 2: Individual representation

From Fig. 2, it can be conclude that the attributes A, C and F are in antecedent part as their tag values are 00 whereas B and E are in consequent part as their tag values are 11. Other than the tag values 00 and 11 all are absent from rule.

**Fitness functions:** The fitness functions are also same as the fitness functions of classification rule mining[22] with a little modification. Let us discuss these fitness functions.

**Confidence factor:** The measure like confidence factor of association rule mining is same as classification rule mining i.e.

$$C_f(\Re) = \frac{|A \,\&\, C|}{|A|} \qquad (1)$$

where $|A \,\&\, C|$ is defined as the number of samples satisfies both antecedent and consequent part. Similarly |A| is defined as the number of samples satisfies only the antecedent part. The only modification required is in comprehensibility and interestingness measure.

**Comprehensibility:** A careful study of the association rule will infer that if the number of conditions involved in the antecedent part is less, then the rule is more comprehensible. The following expression can be used to quantify the comprehensibility of an association rule.

$$C(\Re) = \frac{\log(1+|C|)}{\log(1+|A \,\&\, C|)} \qquad (2)$$

Where $|C|$ and $|A \,\&\, C|$ are the number of attributes involved in the consequent part and the total rule respectively.

**Interestingness:** As we mentioned earlier in the classification rules[22] the measures can be defined by information theoretic[21]. This way of measuring interestingness of the association rule will become computationally inefficient. For finding interestingness, the dataset is to be divided based on each attribute present in the consequent part. Since a number of attributes can appear in the consequent part and they are not predefined, this approach may not be feasible for association rule mining. So a new expression is defined which uses only the support count of the antecedent and the consequent parts of the rules and is defined as

$$I(\Re) = \left(\frac{|A \,\&\, C|}{|A|}\right) \times \left(\frac{|A \,\&\, C|}{|C|}\right) \times \left(1 - \frac{|A \,\&\, C|}{|D|}\right) \qquad (3)$$

where $|D|$ is the total number of records in the database.

Although there are many standard MOGA[24-30] can be used for association rule mining problem but some difficulties associated with them. In case of rule mining problems, we need to store a set of better rules found from the database. If we follow the standard genetic operators only, then the final population may not contain some rules that are better and were generated at some intermediate generations. It is better to keep these rules. For this task, a separate population is used. In this population no genetic operation is performed. It will simply contain only the non-dominated chromosomes of the previous generation. The user can fix the size of this population. At the end of first generation, it will contain the non-dominated chromosomes of the first generation. After the next generation, it will contain those chromosomes, which are non-dominated among the current population as well as among the non-dominated solutions till the previous generation. The genetic materials like crossover and mutation are same as single objective association rule generation algorithm.

**Parallel MOGA for association rule mining:** There are two broad sources of parallelism in MOGA[31-33]. One can exploit parallelism in the application of genetic operators- such as selection, crossover, mutation-and/or in the computation of the fitness of the population individuals (candidate rules). In the context of mining very large databases, the later tends to be far more important. The reason is that the genetic operators are usually very simple and their application computationally cheap. Hence, the bottleneck of the algorithm is the computation of the individual's fitness, whose processing time is proportional to the size of the data being mined. In this work, we propose two models, which are illustrated in Fig. 3 and 4. Let us discuss how these models work. In model-1, the data being mined is divided and distributed across the processors. The populations that are initiated by master are also replicated to different processors. The processors then compute the fitness of each individual based on the

local data in parallel. After processors compute a partial measure of fitness for all the individuals by accessing only its local dataset, then transfer it to the master processor. As soon as the master receives the fitness of all the individuals from different sources then enter into the accumulation phase. In accumulation phase, the task of master is to add the fitness of all individuals.

Mathematically, suppose there are k numbers of processor involved in the model and P is the population pool that contains $\{I_1, I_2, ....., I_n\}$, be the set of individuals. As the entire P is given to all the available processors, the fitness collected from the different processors is defined as $\vec{P}_1 = \{\vec{f}_{11}, \vec{f}_{12}, \vec{f}_{13}, ....., \vec{f}_{1n}\}$, $\vec{P}_2 = \{\vec{f}_{21}, \vec{f}_{22}, \vec{f}_{23}, ....., \vec{f}_{2n}\}$ ,……, $\vec{P}_k = \{\vec{f}_{k1}, \vec{f}_{k2}, \vec{f}_{k3}, ....., \vec{f}_{kn}\}$. The job of master processor $P_M$ is to find out the value of $\vec{P}_M = \vec{P}_1 + \vec{P}_2 + ..... + \vec{P}_k$. After fitness computation is over then the master processors do the rest of the genetic operations. This process is repeated until we achieve a user expected set of non-dominated solutions. The most important advantage of this model, in the context of data mining is that, intuitively it is much more scalable with respect to the size of the data being mined than the control parallel approach. To put it in simply terms, more data leads to a larger degree of data parallelism to be exploited. Note that data and control parallelism address different kinds of large problems. Data parallelism addresses the problem of very large databases. Control parallelism addresses the problem of very large search spaces. Hence, it would be desirable to exploit both kinds of parallelism in a multi-objective genetic algorithm for data mining. Model-2 addresses these two kinds of parallelism.

In this model the following protocols are used: i) logically groups the processors using nearest-neighbor techniques, ii) generate population in different group based on the assigned goal and iii) distribute the population and mining domain among the group members. Let us see how this model works. Assume that there are k number of processors available in a particular group. The dataset $X = x_1, x_2, ....., x_n$ contains $n$ number of points, so divide it into equal subsets based on the available processors in a particular group. In other words, divide and distribute the datasets based on the available processors in a group. After allocating the data, then generate a population pool in any of the processors available in that group and distribute it equally to all the members of that group. After work assignment phase is over then the fitness evaluation phase is started in the following way. Now the fitness evaluation phase will start and exploits both data parallelism and control parallelism by having the individuals passing through all the processors in a kind of round-robin scheme. In this scheme the physical interconnection of processors nodes is mapped into a logical ring of processor nodes, so that each processors node has a right neighbour and left neighbour.

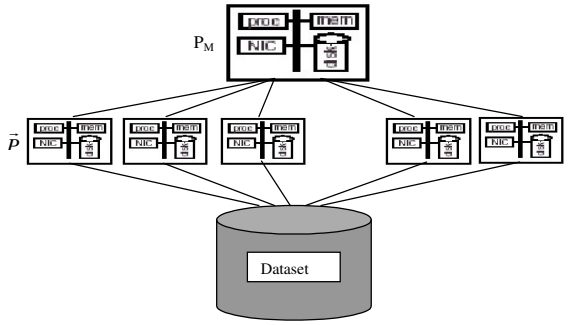


Fig. 3: Master slave model 1 for data parallelism

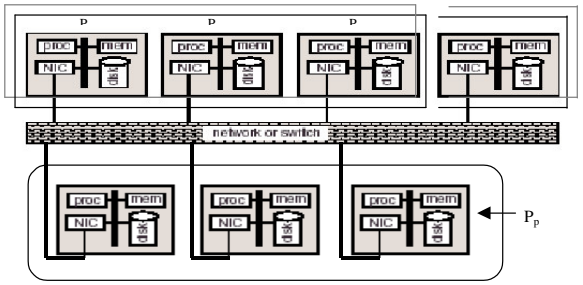


Fig. 4: Master slave shared nothing_architecture

At first each processor nodes computes a partial measure of fitness for all the individuals (rules) in its local subpopulation, by accessing only its local dataset. Then each processor transfer its entire local subpopulation of individuals, as well as the value of their partially computed fitness function, to it's right neighbour. As soon as a processor node receives a subpopulation of individuals from its left neighbour, it performs the following tasks: (*i*) it computes the partial fitness measure of the incoming individuals on its local dataset; (*ii*) it combines this partial fitness measure with the previous one of the incoming individuals to produce a new fitness measure; (*iii*) it forwards the incoming individuals, as well as their updated partial fitness measure, to its right neighbour. This process is repeated until all individuals have passed through all the processors and returned to their original processors, with their final fitness value duly computed. The aforesaid scheme is applicable to all processors groups. Note that what is being passed through the processors are only individuals and their partial fitness value, not the data being mined. This minimizes inter-process communication overhead. As allocation takes place before processing start, therefore it is called static allocation. Since no data and individual skew arises in this model so considering load balancing is not as much meaningful. The pseudocode required to implement this model is as follows. The pseudocode required for simulation studies is as follows:

**Pseudocode**

1. t=0
2. Initialize P(t) in G_Master and distribute equally to each available processors.

3. Evaluate $P_i(t)$ in each processor based on local dataset by round-robin scheme ∀i.
4. while (t <= no_of_gen)
5. Send_String ($P_i(t)$, Master)
6. P'(t)←M_Selection(P(t))
7. P"(t)←M_Recombination_and_Mutation(P'(t))
8. Distribute P"(t) to each available Processors.
9. Evaluate $P"_i(t)$ in each processor based on the local dataset.
10. t = t+1
11. $P_i(t)←P"_i(t)$
12. end while
13. Decode the individuals obtained from the masters of each group in IF-THEN rules.

Two major types of parallel programming paradigm are available to implement the proposed models like message passing and shared memory models. Message passing model is a parallel programming paradigm that requires programmers to explicitly indicate in their codes where the communication begin, who the senders and receivers are and what and how data will be sent. On the other hand, shared memory model, by making programmers see as if all processors have a single shared memory, eliminates all explicit communication required in message passing model and thus is easier for programmers to implement. However message-passing model is believed to give more speedup, since programmers are aware of parallelism and design the code accordingly to suit its parallel behaviors. Message passing model transfers data and synchronization information simultaneously at communication points by send and receive commands but in shared memory model, data are sent when page faults occurs and synchronization are performed at barriers and acquisitions of lock, resulting in a larger amount of communication.

At first, both message passing and shared memory models were implemented mostly on parallel computers and on hardware-supported distributed shared memory (DSM) clusters but as networks have increased their communication speed enormously and processors have gained higher and higher performance every year, the performance gap between parallel computers and clusters of workstations, even though still exists, is becoming closer, making clusters of workstations an excellent alternative architecture for parallel computing at a relatively low cost. Software distributed shared memory is sometimes referred to as shared virtual memory (SVM). As mentioned before, SVM suffers in terms of performance from a large amount of communication. Moreover, it also suffers from false sharing which occurs when multiple processors accesses different variables co-located on the same page and at least one access is a write. This kind of problem occurs in software DSM due to a large granularity of its virtual memory page.

In this study, PMOGA for association rule mining is implemented using MPICH, a freely available, portable implementation of MPI standard (message passing interface) for message passing runtime libraries. MPI libraries provide some additional features that can increase performance even further, such as the ability to send a block consisting of multiple data in a single message, the ability to send messages in non-blocking mode and the ability to use broadcast and multicast.

**Experimental studies:** The experiments were performed on a cluster of workstations using the following protocols: MPI (Message Passing Interface) for formulating cluster, eight 350 Mhz. Pentium III computers each with 128 MB RAM and 60 GB disk, with operating system Linux Redhat 6.5. The interconnection network was Ethernet with 10Mbps. In our implementation of MPI, we use a runtime library, MPICH, an implementation of MPI, the standard for message passing libraries. A synchronous master slave model is implemented in our PMOGA program because its programming style is easy, straightforward and also gives us opportunities to observe its characteristic more clearly than other models. With this model, only fitness evaluations are parallelized while all other functions in MOGA are done at the master node. Not only distributing individuals to slave nodes, the master node also assigns itself the same number of individuals and performs fitness evaluations of them. In MPI we try to send and receive the data by using normal *MPI-Send* and *MPI-Receive* command.

Our proposed models are validated using an artificially created dataset having 38 attributes and 8330 data points are presented here. We set some MOGA parameters to be constant for all experiments, such as chromosome length (same as the number of attributes of the dataset), the number of generations (300), probability of crossover (0.75) and probability of mutation (0.02). The results presented here are calculated from an average after 5 runs of each experiment set.

In MOGA for association rule mining, there are two parameters that can be defined as the problem size: chromosome length and population size. We fix the chromosome length equal to the number of attributes involved in the dataset and adjust the population size, ranging from 200 to 1000. Figure 3 shows the speedup of the models, running 300 hundred generations with three processors.

When we increase the population size to 1000, then accordingly the evaluation time also increases. Hence it is easier for us to observe clearly speedup by the effects of the number of parallel processor. With more parallel processors running and sharing the loads, the speedup gets higher. High computation-to-communication ratio leads to near linear speedup. Figure 5 shows the speedup when using 3 parallel processors almost

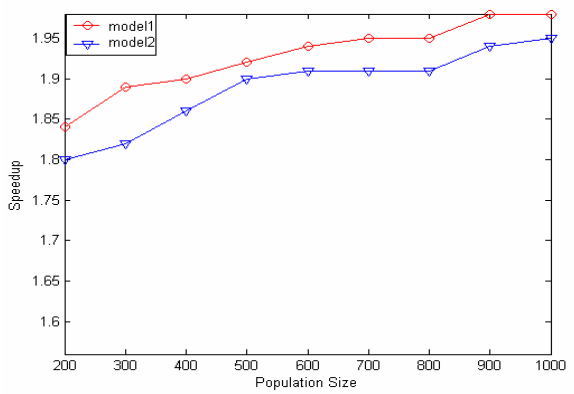


Fig. 5: Speedup obtained from the artificially created dataset

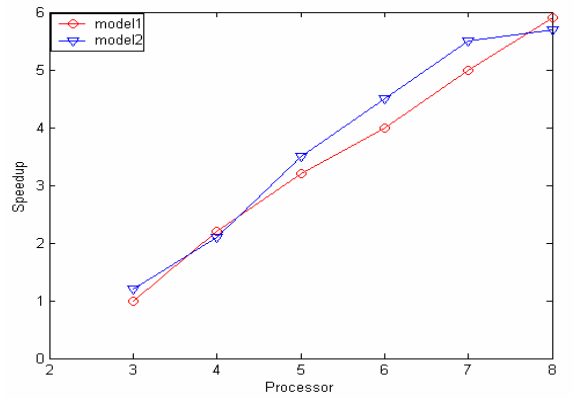


Fig. 6: Speedup with increase in the number of parallel

Table 1: Results obtained from our proposed model

| Sample Size | Number of Generations | Number of Rules Generated |
|---|---|---|
| 1000 | 100 | 24 |
| | 200 | 31 |
| | 300 | 31 |
| 1000 | 100 | 35 |
| | 200 | 40 |
| | 300 | 40 |
| 1000 | 100 | 27 |
| | 200 | 36 |
| | 300 | 37 |
| 2000 | 100 | 35 |
| | 200 | 40 |
| | 300 | 40 |
| 2000 | 100 | 35 |
| | 200 | 40 |
| | 300 | 40 |

approaches 1.2 but is around 5.5 when using 7. After that if we increase the number of processor the performance of the model 2 decrease because of more communication overhead.

Sample size and the number of rules generated by our PMOGA are put in the Table 1. The Table shows the result of parallel models by using the generation ranges from 100 to 300 and the sample size ranges from 1000 to 2000.

From the experiment it has been observed that the generated rule sets is same as the result obtained by sequential algorithm, which is proposed by Ghosh *et al.*[23]. This is because the models only parallelize the fitness computation procedure and rest operations are same as sequential algorithm. Further, the search space exploration is same in both cases.

From the rule sets generated for different samples and for different number of generations it is observed that after 200 generations it ceases to generate more rules; in other words after that number of generations the GA converges. From the results given above it can be seen that only for the third sample, it give an extra rule at the cost of 100 additional generations. Moreover, only a few numbers of attributes (3-4 attributes on both the antecedent and consequent parts) got involved in the rules, which means that all the attributes are not equally important; and the rules are simple to understand (comprehensible). The generated rules were not that much interesting (interestingness value was order of 0.005).

## CONCLUSION

In this study, we have described two models to parallelize the association rule mining using multi-objective genetic algorithm. The proposed models exploits both data and control parallelism. The results show that our proposed models can achieve considerable speed up with a limited constraint. Further, the number of rules generated from these models is provided. It is observed that the result is similar to that obtained using sequential algorithms. This is due to the fact that the parallelism is obtained only in fitness computation level and rest of the operation is same as sequential one. The future improvement includes a more extensive set of experiments with a continuous and mixed real life datasets, to further validate the results reported in this study. The use of a multi-objective evolutionary framework for association rule mining offers a tremendous flexibility to exploit in further work. In particular we are currently investigating the integration of feature selection and association rule mining.

## ACKNOWLEDGEMENTS

This work was carried out when Dr. S. Dehuri held the visiting scientist position in Center for Soft

Computing research, a National facility, located at the Indian Statistical Institute, Kolkata, on leave from P.G. Department of Information and Communication Technology, Fakir Mohan University, Vyasa Vihar, Balasore- 756019, ORISSA, India.

## REFERENCES

1. Agrawal, R. and J.C. Shafer, 1996. Parallel mining of asociation rules: Design, implementation and experience. IBM Research Report RJ 10004. To appear in IEEE Trans. Knowledge and Data Engineering
2. Imielinski, T., R. Agrawal and A. Swami, 1993. Mining association rules between sets of items in large databases. Proc. ACM SIGMOD Conf. Management of Data, pp: 207–216.
3. Hsu, W., B. Liu and S. Chen, 1997. Using general impressions to analyze discovered classification rules. Proc. of 3rd Intl. Conf. On Knowledge Discovery & Data Mining (KDD-97), pp: 31–36. AAAI Press
4. Freitas, A.A., E. Noda and H.S. Lopes, 1999. Discovering interesting prediction rules with a genetic algorithm. Proc. Conf. Evolutionary Computation, (CEC-99), pp: 1322–1329. Washington D.C., USA.
5. Bell, D.A., S.S. Anand and J.G. Huges, 1994. Parallelising the discovery of strong rules from databases. Internal Report, Faculty of Informatics, University of Ulster (Jordanstown).
6. Karypis, G., E-.H. (S.) Han and V. Kumar, 1997. Scalable parallel data mining for association rules. Proc. 1997 ACM SIGMOD Intl. Conf. Management of Data.
7. Agrawal, R. and R. Srikant, 1994. Fast algorithms for mining association rules. Proc. 20th Intl. Conf. Very Large Databases, Santiago, Chile.
8. Imielinski, T., R. Agrawal and A. Sami, 1993. Mining association ruels between sets of items in large databases. Proc. ACM SIGMOD Conf. Management of Data, pp: 207–216.
9. Houtsma, A. and M. Swami, 1993. Set-oriented mining of association rules. Research Report.
10. Lin, D.L. and Z.M. Kedem, 1998. Pincer-search: An efficient algorithm for discovering the maximal frequent set. Proc. 6th Eur. Conf. Extending Database Technology.
11. Brin, S. *et al*., 2002. Parallelism and evolutionary algorithms. IEEE Trans. Evolutionary Computation, 6: 443–461.
12. Adamo, J.M., 2001. Data mining for association rules and sequential patterns. Springer Verlag, New York.
13. Fedelis, M.V. *et al*., 2000. Discovering comprehensible classification rules with a genetic algorithm. Proc. Congr. Evolutionary Computation.
14. Freitas, A.A., 2002. Data mining and knowledge discovery with evolutionary algorithms. Springer-Verlag, New York.
15. Mueller, A., 1995. First sequential and parallel algorithms for association rule mining: A comparison. Tech. Report-CS-TR-3515, Univ. of Maryland, College Park, Md.
16. Chen, M-S., J. S. Park and P.S. Yu, 1995. An effective hash-based algorithm for mining association rules. Proc. ACM SIGMOD Conf., ACM Press, New York, pp: 175–186.
17. Agrawal, R. *et al*., 1996. Fast Discovery of Association Rules. Advances in Knowledge Discovery and Data Mining, U. Fayyad *et al*., (Eds.), AAAI Press, Menlo Park, Calif., pp: 307–328.
18. Shintani, T. and M. Kitsuregawa, 1996. Hash based parallel algorithms for mining association rules. Proc. 4th Intl. Conf. Parallel and Distributed Information Systems, IEEE Computer Soc., Press., Los Alamitos, Calif., pp: 19–30.
19. Wang, W., J. Chiang, W. Gong, K. Koperski, D. Li, Y. Lu, A. Rajan, N. Stefanovic, B. Xia, J. Han, Y. Fu and O.R. Zaiane, 1996. Dbminer: A system for mining knowledge in large relational databases. In Proc. Intl. Conf. on Data Mining and Knowledge Discovery (KDD'96), pp: 250–255.
20. Zaki, M.J. *et al*.,1997. New algorithm for fast discovery of association rules. Proc. 3rd Intl. Conf. Knowledge Discovery and Data Mining, AAAI Press, Menlo Park, Calif., pp: 283–286.
21. Freitas, A.A., 2003. A survey of evolutionary algorithms for data mining and knowledge discovery. In: A. Ghosh, S. Tsutsui (Eds.), Advances in Evolutionary Computing, Springer Verlag, New York, pp: 819–845.
22. Dehuri, S. and R. Mall, 2004. Mining predictive and comprehensible rules using a multi-objective genetic algorithm. Advance Computing and Communication (ADCOM), India.
23. Ghosh, A. and B. Nath, 2004. Multi-objective rule mining using genetic algorithm. Inform. Sci., 163:123–133.

24. Ghosh, A. and S. Dehuri, 2004. Evolutionary algorithms for multi-criterion optimization: A survey. Intl. J. Comp. Inform. Sci., 2: 38–57.
25. Deb, K., 2001. Multi-objective optimization using evolutionary algorithms. Wiley, New York.
26. Fonseca, C.M. and P.L. Fleming, 1993. Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. Fifth Intl. Conf. Genetic Algorithms, pp: 416–423.
27. Fonseca, C.M. and P.J. Fleming, 1995. An overview of evolutionary algorithms in multi-objective optimization. Evolutionary Computation, 3: 1–16.
28. Shaffer, J.D., 1985. Multiple objective optimization with vector evaluated genetic algorithm. First Intl. Conf. Genetic Algorithms, pp: 93–100.
29. Deb, K., E. Zitzler and L. Thiele, 2000. Comparison of multi-objective evolutionary algorithms: Empirical results. Evolutionary Computation, 8: 125–148.
30. Vose, M.D., 1999. The simple genetic algorithm: Foundations and theory. MIT Press, Cambridge.
31. Cantu-Paz, E., 2005. A summary of research on parallel genetic algorithms. IlliGAL Technical Report.
32. Lopes, H.S., D.L.A. Araujo and A.A. Freitas, 1999. A parallel genetic algorithm for rule discovery in large databases. Proc. IEEE Systems, Man and Cybernetics Conf., 3: 940–945.
33. Alba, E. and M. Tomassini, 2002. Parallelism and evolutionary algorithms. IEEE Trans. Evolutionary Computation, 6: 443–461.