Parallel Multi-objective Genetic Algorithm for Classification Rule Mining

SATCHIDANANDA DEHURI

Department of Information & Communication Technology, Fakir Mohan University, Vyasa Vihar, Balasore 756 019, India.
email: satchi.lapa@gmail.com

ASHISH GHOSH

Machine Intelligence Unit and Centre for Soft Computing Research, Indian Statistical Institute, 203, B T Road, Kolkata 700 108, India.
email: ash@isical.ac.in

AND

RAJIB MALL

Department of Computer Science & Engineering, Indian Institute of Technology Kharagpur, Kharagpur 721 302, India.
email: rajib@cse.iitkgp.ernet.in

Multi-objective genetic algorithms (MOGAs) are finding increasing popularity as researchers realize their potential for obtaining good solutions to mining problems in large databases. Parallel multi-objective genetic algorithms (pMOGAs) attempts to reduce the processing time needed for computing the fitness functions and to reach an acceptable solution. We propose two different master slave models of pMOGA. Our proposed models exploit both data parallelism by distributing the data being mined across various processors, and control parallelism by distributing the population of individuals across all available processors. These models are implemented through a cluster computing environment and we measure the speed up of pMOGA over its sequential counterpart.

Indexing terms: Rule mining, Multi-objective genetic algorithms, Parallel multi-objective genetic algorithms, Cluster-computing.

1. INTRODUCTION

DATA mining (DM) and knowledge discovery in databases (KDD) community is trying to address the problem of scaling up the mining algorithms with respect to the size of the databases being mined [1]. Another issue that needs urgent attention is that mining problems are often multi-objective problems rather than single objective ones. There can be several objectives that may have to be optimized. However, measures such as predictive accuracy; comprehensibility and interestingness used for evaluating a rule can be thought of as important objectives of any mining scheme [2]. Parallel processing can be regarded an attractive way to both the problem of scalability and simultaneously optimizing all the objectives involved in the mining problem [3]. In this paper we address the well-known classification task of data mining in a cluster environment. Since MOGA tend to be slow, compared to most classical rule generation methods, the design of parallel MOGA for data mining is an important research area [4,5]. MOGA for data mining tasks and its parallelization is a recent field of research. Since a great deal of literature exists describing generic parallel concepts, we focus only on exploring and analyzing possible benefits of pMOGA for our mining scheme. In the classification rule mining, the extracted rules are often expressed as a set of rules of the form:

IF < Predictive attribute> THEN < Prediction of the Goal attribute>

In our work, a pMOGA that uses Message Passing Interface (MPI) [6-9] is used to discover rules from a database. MOGA associates all individuals of a population with the same predicted class, which is never modified during the running of the algorithm [2]. Hence, if we want to discover a set of classification rules, predicting *k* different classes, we need to run the

MOGA at least k times, so that in the ith run, $i = 1, 2, 3, \ldots, k$, the algorithm discovers only the rules pertaining to the ith class [10]. This particular characteristic allows us to exploit a ground level parallelism in rule mining scheme.

The organization of this paper is as follows. Section 2, briefly describes MOGA for rule mining. Section 3, presents an overview of pMOGA. Section 4, describes pMOGA for classification rule mining. Section 5 covers experimental results and their analysis. Section 6, provides the concluding remarks and future directions of our research.

2. MOGA FOR RULE MINING

As already mentioned the rule mining problem can be considered as a multi-objective problem. The predictive accuracy, comprehensibility and interestingness used for evaluating a rule can be considered as important objectives of classification rule mining. This makes it an optimization problem that is very difficult to solve efficiently. Intuitively, MOGA appears to be the best-suggested techniques for this problem. The details of the MOGA based rule mining are available in [2]. The MOGA algorithm can be explained in pseudocode form as follows [9].

Algorithm

```
1. Select an initial population randomly.
2. Evaluate the comprehensibility, predictive accuracy and interestingness of all the individuals.
3. Rank the chromosomes depending on the non-dominance property.
4. Assign fitness to the chromosomes using the ranks.
5. Select the chromosomes for next generation, by roulette wheel selection scheme using the fitness calculated in Step 4.
6. Perform multi-point crossover and mutation on these individuals.
7. If the performance is satisfactory, then stop; otherwise, goto Step-2.
8. Decode the chromosomes in the final stored population and get the generated rules.
```

The MOGA for rule mining differs from the single objective GA in respect of the way the selection operator works. The next subsection briefly discusses the individual representation, fitness functions and genetic operators used for classification rule discovery.

2.1. Individual Representation

Each individual in the population represents a candidate rule of the form if A then C. Suppose there are N attributes in our dataset (including the class attribute). The antecedent of the rule can be formed by conjunction of at most '$N - 1$' attributes. Each condition is of the form $a_i = d_{ij}$ where a_i is the ith attribute and d_{ij} is the ith value of the ith attribute's domain. The consequent consists of a single condition of the form $C = d_l$, where, C is the goal attribute and d_l the lth value of the goal attribute's domain. Since we are considering variable length chromosomes, if an attribute is not present in the rule antecedent the corresponding value in gene is '#'. This value is represented as a flag to indicate that the attribute is absent in the rule antecedent.

2.2. Fitness Function

As discussed in Section 1, the discovered rules should have the following desirable features: (*a*) high predictive accuracy, (*b*) high comprehensibility, and (*c*) interestingness. In this subsection, we discuss each of these measures as a fitness criterion and how these can be meaningfully measured quantitatively.

2.2.1. *Comprehensibility Metric*

There are various ways to quantitatively measure rule comprehensibility [11]. The standard way of measuring comprehensibility is to count the number of rules and the number of conditions in these rules. As the number of rules increases then the comprehensibility decreases.

If a rule has at most 'M_c' conditions, the comprehensibility 'ς' of a rule '$\Re$' can be defined as:

$$\varsigma(\Re) = 1 - (N_c(\Re) / M_c) \qquad (1)$$

where '$N_c(\Re)$' is the number of conditions in the rule $\Re$.

2.2.2. Predictive Accuracy

Like comprehensibility, there are also various ways to measure rule predictive accuracy [12]. As already mentioned, our rules are of the form IF $A_1 \wedge A_2$ THEN C. The antecedent part of the rule is a conjunction of conditions. A very simple way to measure the predictive accuracy of a rule '$P(\Re)$' is

$$P(\mathfrak{R}) = \frac{|A \,\&\, C|}{|A|} \tag{2}$$

where $|A|$ is the number of examples satisfying all the conditions in the antecedent A and $|A \,\&\, C|$ is the number of examples that satisfy both the antecedent A and the consequent C. Intuitively this metric measures predictive accuracy in terms of how many cases both the antecedent and the consequent attributes are present.

2.2.3. *Interestingness*

The computation of the degree of interestingness of a rule, in turn, consists of two terms. One of them refers to the antecedent of the rule and the other to the consequent. The degree of interestingness of the rule antecedent is calculated by an information-theoretic measure, which is a normalized version of the measure proposed in [13]. Initially, the algorithm calculates the information gain of each attribute (InfoGain)[14]. The degree of interestingness of the rule antecedent (RInt) is computed by using the following expression:

$$RInt = 1 - \frac{\dfrac{\sum_{i=1}^{n-1} InfoGain(A_i)}{n-1}}{\log_2(|dom(G)|)} \tag{3}$$

where n is the number of attributes in the antecedent and $\log_2(|dom(G)|)$ is the domain cardinality (i.e. the number of possible values) of the goal attribute G occurring in the consequent. The log term is included in the formula (3) to normalize the value of RInt, so that this measure takes on a value in [0,1]. The InfoGain is given by:

$$InfoGain(A_i) = Info(G) - Info(G \mid A_i)$$

where

$$Info(G) = -\sum_{i=1}^{m_k} \left(P(g_i)\log_2(P(g_i))\right)$$

$$Info(G|A_i) = \sum_{i=1}^{n_i} \left(p(v_{ij})\left(-\sum_{j=1}^{m_k} p(g_l|v_{ij})\log_2(p(g_l|v_{ij}))\right)\right)$$

where m_k the number of possible values of the goal attribute G_k, n_i the number of possible values of the attribute A_i, $p(X)$ denotes the probability of X and $p(X \mid Y)$ denotes the conditional probability of X given Y.

2.3. Genetic Operators

The crossover operator we consider here is based on uniform multi-point crossover. There is a probability for applying crossover to a pair of individuals and another probability for swapping each gene attribute's value in the genome (rule antecedent) of two individuals. After crossover is complete, the algorithm analyses if any invalid individual is created. If so, a repair operator is used to produce valid-genotype individuals.

The mutation operator randomly transforms the value of an attribute into another value belonging to the same domain of the attribute.

Besides crossover and mutation, the insert and remove operators directly try to control the size of the rules being evolved; and thus influence the comprehensibility of the rules. These two operators randomly insert and remove, respectively, a condition in the rule antecedent. These operators are not part of a regular GA. However, we have introduced them here because of their suitability in our rule mining scheme.

3. AN OVERVIEW OF *p* MOGA

MOGA is suitable for parallelization as crossover, mutation and (in particular the time-consuming) fitness evaluation can be performed independently on different processors. The main problem is the selection operator, where global information is required to determine the relative performance of an individual with respect to all others in the current population. The three-main parallel models are Master-slave, Island and diffusion models [15].

(1) *Master slave model:* In this model the objective function evaluations are distributed among slave processors while a master processor executes the other MOGA operations to reduce the overall execution time. The search space exploration is conceptually identical to that of a serial MOGA.

(2) *Island model:* In this model, every processor runs an independent GA, using a separate sub-population. The processors co-operate by regularly exchanging migrants (good individuals). The island model is particularly suitable for computer clusters, as communication is costly and therefore needs to be limited.

(3) *Diffusion model:* Here the individuals are spatially arranged, and mate with other individuals from the local neighborhood. When

plication for this jou

parallelized, there is a lot of inter-processor communication (as every individual has to communicate with its neighbours in every iteration), but the communication is only local. Thus, this paradigm is particularly suitable for massively parallel computers with a fast local intercommunication network.

In our work, the following two models are proposed and are implemented using a cluster-computing environment. Though, originally the above models were designed for single objective genetic algorithms, they can be used even for multi-objective genetic algorithms for exploiting parallelism in fitness computation as well as distributing the objective functions to different processors. These models are based on master slaves and diffusion concepts and are described below.

Master slave Model 1 exploits data parallelism. Here, we distribute the entire population and subsets of the dataset among all the available processors. Let us see how this model works. Here, the processors send their fitness values based on their local data to the master, and the master accumulates all the fitness values and proceeds for rest of the work. Note that the processors associated with this model are responsible for calculating the fitness of all objective functions of our mining scheme. This method is more scalable. The problem of this model is that when the master is doing some work the slaves sit idle. In addition, the partitioning of the dataset also requires careful attention.

In Model 2 both data and control parallelism is exploited. Here the dataset as well as population is divided and distributed among different available processors. In this model, the individuals compute their fitness based on the local data in which the individuals are assigned the dataset available in rest of the processors through a ring structure. In addition, the processors are divided into different groups. Each group is responsible for computing different objective functions, which is highly data intensive.

The structural representation of these models is schematically shown in Figs 1 and 2.

4. *p*-MOGA FOR CLASSIFICATION RULE MINING

There are two broad sources of parallelism in MOGA. 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 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 is 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.

Let us discuss how these proposed models work for rule mining scheme. In Model 2, the data being mined is divided and distributed among the processors. The populations that are initiated by the master are also replicated in different processors. The processors then compute the fitness of each individual based on their local data in parallel. The most important advantage of this model, in the context of data mining, is that, it is much more scalable with respect to the size of the data being mined than the control parallel approach. Conceptual larger volume of data leads to a larger degree of data parallelism to be exploited.

The pseudocode representation of this model is as follows:

1. Divide the given dataset and distribute among each available processors including master.
2. Randomly initialize the population in master processor and replicate these in all available processors.
3. Compute the fitness of each individual based on the local dataset in parallel and send the result to the master processor.
4. Accumulate the fitness of each individual collected from the different processors.
5. Master processor selects the individuals for next generation by proportional selection scheme using the fitness value computed in Step-4.
6. Perform multi-point crossover and mutation on these new individuals in the master processor.
7. If the performance criterion is satisfied, then stop; otherwise, goto Step-2.

Note that data and control parallelism address different kinds problems [16-18]. Data parallelism addresses the problem of very large databases. Control parallelism addresses the problem of very large search space. Hence, it would be desirable to exploit both kinds of parallelism in a multi-objective genetic algorithm for data mining. Model 2 addresses these

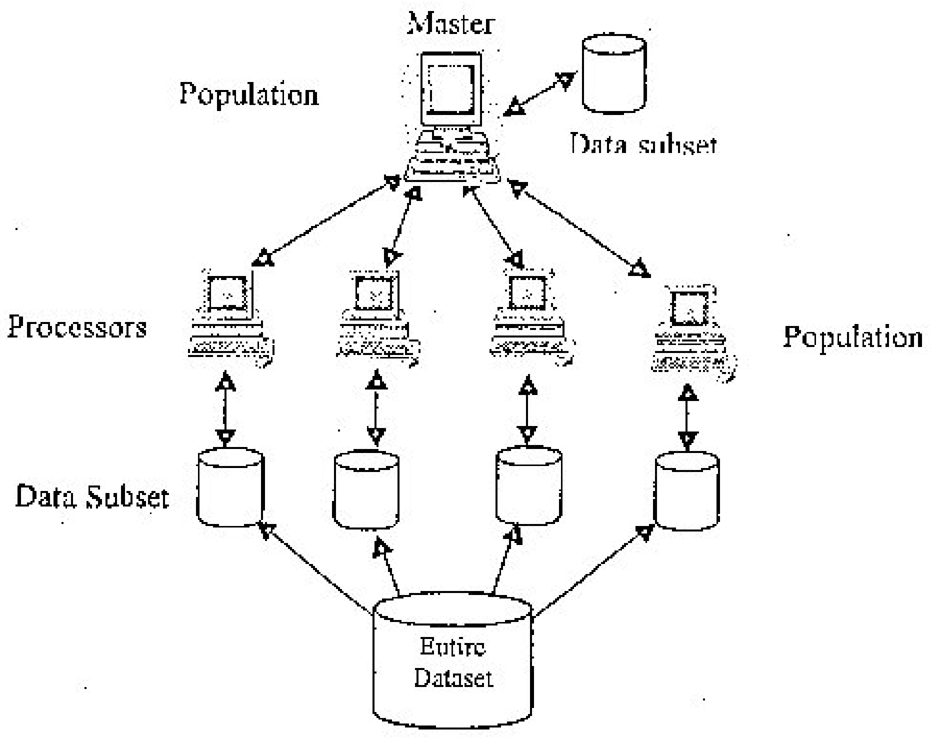

Fig 1 Master slave model based on data parallelism

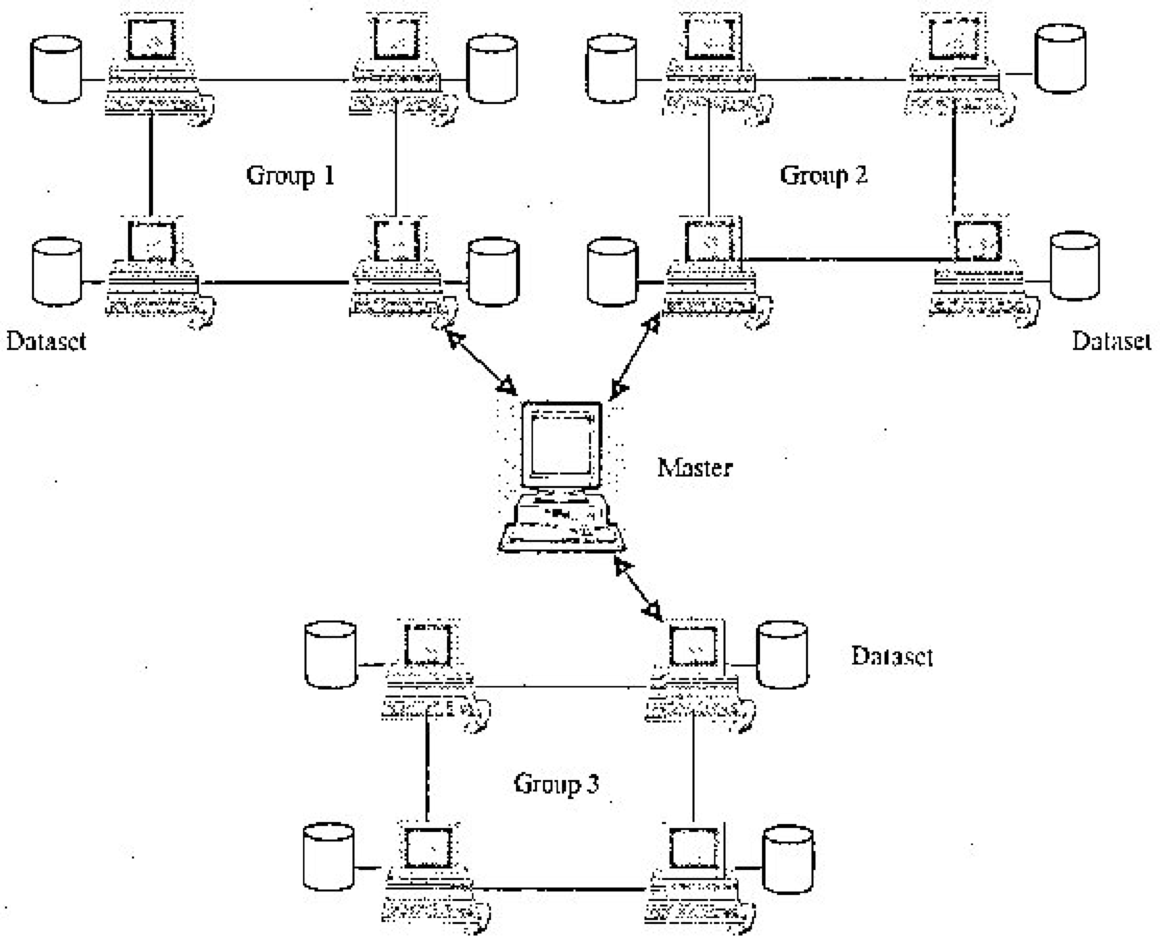

Fig 2 Master slave model based on data and control parallelism

two kinds of parallelism. In addition, the processors are divided into different groups based on the number of objectives in the mining problem. In our classification rule mining problem, the number of objectives is three. So the processor groups are also three. Both the data and control parallelism are exploited through this model.

In this model, the fitness evaluation phase exploits both data parallelism and control parallelism by having the individuals passing through all the processors in a kind of round-robin fashion. In this scheme, the physical interconnection of processor nodes is mapped into a logical ring of processor nodes, so that each processors node has a right neighbour and left neighbour.

At first each processor node computes a partial measure of fitness for all the individuals (rules) in its local subpopulation, by accessing only its local dataset. Then, each processor transfers 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 processor groups for their respective objective functions. 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.

5. EXPERIMENTAL RESULTS

This section discuses the results of our experiments using two datasets obtained from the UCI machine repository [19] called Nursery and Adult datasets. The nursery dataset contains 12960 instances (records) 9 attributes and 5 classes. The class values are not_recom, recommend, very_recom, priority, and spec_prior. The adult dataset contains 48844 instances and 15 attributes.

In all our experiments, the genetic algorithm had 200 individuals in each subpopulation, and it was run for 100 generations. The experiments were performed on a cluster of workstations using the following protocols: MPI (Message Passing Interface) for formulating cluster, 350 MHz. Pentium III computers each with 128 MB RAM and 6 GB disk, with operating system Linux Redhat 6.5. The interconnection network was Ethernet at 10 Mbps. In our system, one of the four processors runs the master program, which controls the slave programmes.

We have measured the speed up (Spd) of the parallel version of the algorithm over its sequential counterpart, defined as: Spd $= \frac{T_s}{T_p}$, where T_s is the sequential processing time and T_p is the parallel processing time on p processors. Tables I and II shows the speed up value for the two different datasets obtained from Model 1.

As shown in Tables 1 and 2, the parallel MOGA (called pMOGA) achieved a reasonable speed up over the sequential version. As expected the speed up was greater in the case of Adult dataset. The reason is that this dataset is larger than the Nursery dataset, so there is more opportunity for the exploitation of data parallelism than the former. Of, course, real life datasets would usually be significantly larger then the two benchmarking datasets used in our experiments. Therefore, we expect that our model will achieve higher speed-ups for large real life dataset.

Tables 3 and 4 show the speed up of the two different datasets obtained from Model 2. Figures 3 and 4 shows the speed up achieves by the two bench marking datasets. From the Fig 4 it has been observed that if the dataset is too large then Model 2 is the best suitable.

6. CONCLUSIONS

We have discussed two different parallel models for classification rule mining using multi-objective genetic algorithm. Model 1 exploits data parallelism where as Model 2 exploits both data and control parallelism. Our experimental results show that a good speed up can be achieved by Model 2 provided that the data being mined is of relatively large size. This is due to the fact that the degree of data parallelism is proportional to the size of the data being mined. Our future work would concentrate on more extensive set of experiments with a continuous and nominal datasets, to further validate the results reported in this

TABLE 1: Speed up obtained from nursery dataset

No. of Processor	SPT	PPT	Speed Up
2	30 min.	17 min.	1.7647
3	30 min	12 min	2.5
4	30 min.	10 min	3.0
5	30 min.	10 min	3.0
6	30 min	11 min	2.7272

TABLE 2: Speed up obtained from adult dataset

No. of Processor	SPT	PPT	Speed Up
2	90 min.	48 min.	1.875
3	90 min.	32 min.	2.8125
4	90 min.	30 min.	3.0
5	90 min.	30 min.	3.0
6	90 min.	32 min.	2.8125

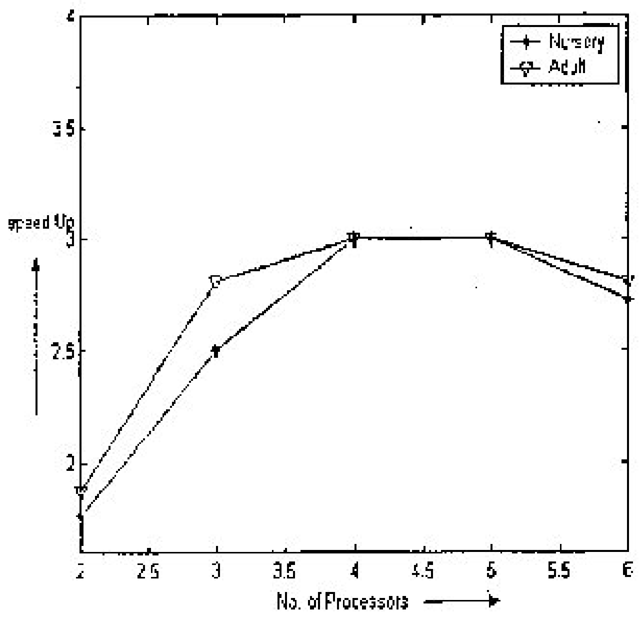

Fig 3 Number of processors vs speed up of Model-1

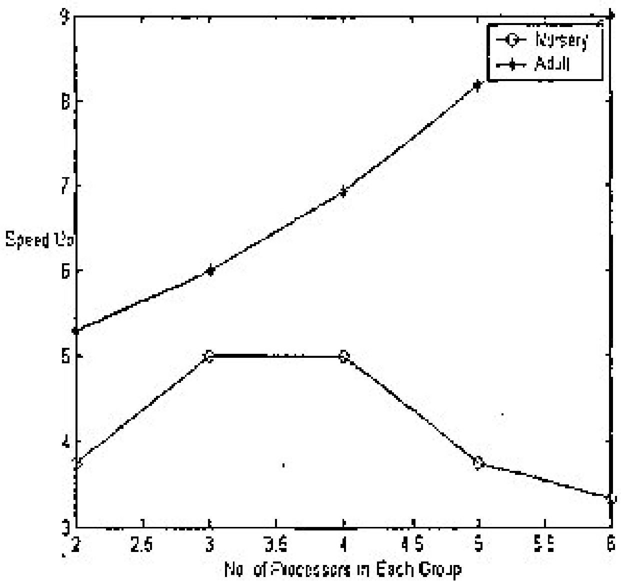

Fig 4 Number of processors in each group vs speed up of Model-2

paper. In addition, a general framework is needed for any kind of rule mining problem with an integrated set of objective functions.

ACKNOWLEDGEMENT

Authors would like to thank professor S Pattanayak of the Institute of Mathematics and Application, Bhubaneswar and our other colleagues for helpful discussions.

REFERENCES

1. R Agrawal & JC Shafer, Parallel mining of association rules: design, implementation and experience, *IBM Research Report RJ 1004*, 1996.
2. S Dehuri & R Mall, Predictive and comprehensible rule discovery using a multi-objective genetic algorithm, *Knowledge Based System*, vol 19, pp 413-421, 2006.
3. A A Freitas & S H Lavington, *Mining very large databases with parallel processing*, Kluwer, 1998.
4. C M Fonseca, & P J Fleming, An overview of evolutionary algorithms in multi-objective optimization, *Evolutionary Computation*, vol 3, no 1, pp 1-16, 1995.
5. E Alba & M Tomassini, Parallelism and evolutionary algorithms, IEEE Transaction on Evolution Computation, vol 6, no 5, pp 443-461, 2002.
6. M Miki, Y Tanimura & T Hiroyasu, Efficiency between the two models of parallel evolutionary algorithms on PC cluster system, *Proceeding of the Fourth International Conference on High Performance Computing in the Asia-Pacific Region*, 2000.
7. D Quagliarella, & A Vicini, Sub-population policies for parallel multi-objective genetic algorithm with applications to wing design, In *IEEE International Conference on Systems, Man and Cybernetics*, vol 4, pp 3142-3147, IEEE, 1998.
8. D A Van Veldhuizen, J B Zydallis & G B Lamont, Considerations in engineering parallel multi- objective

TABLE 3: Speed up obtained from nursery dataset

No. of Processor Processor in each group	SPT	PPT	Speed Up
2	30 min.	8 min.	3.75
3	30 min	6 min	5
4	30 min.6	min	5
5	30 min.8	min	3.75
6	30 min	9 min	3.33

Table 4: Speed up obtained from adult dataset

No. of Processor Processor in each group	SPT	PPT	Speed Up
2	90 min.	17 min.	5.294
3	90 min.	15 min.	6.0
4	90 min.	13 min.	6.923
5	90 min.	11 min.	8.181
6	90 min.	10 min.	9.0

evolutionary algorithms, *IEEE Transactions on Evolutionary Computation*, vol 7, no 2, pp 144-173, 2003.

9. A Ghosh & S Dehuri, Evolutionary algorithms for multi-criterion optimization: A survey, *International Journal on Computing and Information Science*, vol 2, no 1, pp 38-57, 2004.

10. A A Freitas, Generic, set-oriented primitives to support data parallel knowledge discovery in relational databases, *Ph D thesis, University of Essex*, UK, July 1997.

11. M J Pazzani, Knowledge discovery from data? *IEEE Intelligent Systems*, pp 10-12, 2000.

12. D J Hand, *Construction and Assessment of Classification Rules*, Wiley, 1997.

13. J M Cover & J A Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991.

14. A A Freitas, On objective measures of rule surprisingness, *Proc of 2nd European Symposium on Principle of data Mining and Knowledge Discovery (PKDD-98)*, Lecturer Notes in Artificial Intelligence, 1510, 1-9, 1998.

15. E Cantu-Paz, A summary of research on parallel genetic algorithms................

16. D L A Araujo, H S Lopes & A A Freitas, A parallel genetic algorithm for rule discovery in large databases, *Proc 1999 IEEE Systems Man and Cybernetics Conf*, V.III, 940-945, Tokyo, Japan.

17. S S Anand, C M Shapcott, D A Bell & J G Hughes, Data mining in parallel, *Proc World Occam and Transputer User Group Conf Manchester*, April 1995.

18. J Shafer, R Agrawal & M Mehta SPRINT: a scalable parallel classifier for data mining, *Proc 22nd Int Conf Very large Databases (VLDB-96)*, Bombay, India, 1996.

19. P M Murphy & D W Aha, UCI Repository of machine learning databases [http://www.ics.uci.edu/-mlern/MLRepository.html], *Irvine. CA: University of California, Department of Information and Communication Science*, 1994.

AUTHORS

Satchidananda Dehuri received the MSc degree in mathematics from the Sambalpur University in 1998, the MTech and the PhD degrees in Computer Science from the Utkal University in 2001 and 2006, respectively. Currently, he is a Reader and the Coordinator of the PG Department of Information and Communication Technology, FM University, Balasore. He also holds the position of Director of Software Development Cell at the University Science Instrumentation Centre, Fakir Mohan University, Vyasa Vihar, Balasore. His research interests include evolutionary algorithm for multi-objective problems, data mining, data warehousing, soft computing and parallel algorithm.

* * * *

Ashish Ghosh received the Bachelor's degree in Electronics and Telecommunication Engineering from Jadavpur University, Kolkata in 1987, the MTech and the PhD degrees in Computer Science from the Indian Stastical Institute, Kolkata in 1989 and 1993, respectively. Presently, he is a Professor of the Machine Intelligence Unit at the Indian Statistical Institute, Kolkata. He has published 100 papers in national and international journals and referred conferences and has edited 6 books. He is a guest editor of various Journals. His research interests include evolutionary computation, neural networks, image processing, fuzzy sets and systems, pattern recognition and data mining. He received the Young Scientist Award in Engineering Sciences from INSA in 1995 and from Indian Science Congress Association in Computer Science in 1992. He is an associate of the Indian Academy of Sciences, Bangalore since 1997.

* * * *

R Mall received the BTech, the MTech and the PhD degrees from the Indian Institute of Science, Bangalore. Presently, he is a Professor in the Department of Computer Science and Engineering at the IIT Kharagpur. He has published more than 150 papers in national and international journals and referred conferences. His research interests include software engineering, data mining and parallel processing techniques. He has supervised 6 PhD theses.

* * * *