International Journal of Computer Applications Technology and Research
Volume 2– Issue 2, 121 - 125, 2013, ISSN: 2319–8656

Bragged Regression Tree Algorithm for Dynamic Distribution and Scheduling of Jobs

Navneet Randhawa
Department of Information Technology
Adesh Institute of Engineering and Technology
Faridkot, India

Davinder Singh Gill
Department of Information Technology
Adesh Institute of Engineering and Technology
Faridkot, India

Parneet Kaur
Department of Information Technology
Guru Nanak Dev University
Jalandhar, India

Abstract: In the past few years, Grid computing came up as next generation computing platform which is a combination of heterogeneous computing resources combined by a network across dynamic and geographically separated organizations. So, it provides the perfect computing environment to solve large-scale computational demands. As the Grid computing demands are still increasing from day to day due to rise in large number of complex jobs worldwide. So, the jobs may take much longer time to complete due to poor distribution of batches or groups of jobs to inappropriate CPU's. Therefore there is need to develop an efficient dynamic job scheduling algorithm that would assign jobs to appropriate CPU's dynamically. The main problem which dealt in the paper is, how to distribute the jobs when the payload, importance, urgency, flow time etc. dynamically keeps on changing as the grid expands or is flooded with number of job requests from different machines within the grid.

In this paper, we present a scheduling strategy which takes the advantage of decision tree algorithm to take dynamic decision based on the current scenarios and which automatically incorporates factor analysis for considering the distribution of jobs.

Keywords: Grid Computing; Job Scheduling; Regression Tree; factorization; Scheduler

1. INTRODUCTION

Grid computing is a high performance computing environment to solve large-scale computational demands. It emerged as a next generation computing platform which is a collection of heterogeneous computing resources connected by a network across dynamic and geographically dispersed organizations, to form a distributed high performance computing infrastructure. Job scheduling is a fundamental issue in achieving high performance in grid computing systems, as it is very difficult to tackle the new features of Grid systems such as its dynamic nature and the high degree of heterogeneity of jobs and resources. Grid computing demands are still increasing from day to day due to rise in large number of complex jobs worldwide. So the jobs may take much longer time to complete due to poor distribution of batches or groups of jobs to inappropriate CPU's. The main goal of scheduling in grid computing is to minimize the job completion time and wastage of CPU cycles but scheduling jobs in a heterogeneous grid environment is different compared to parallel architectures. Before scheduling the tasks in the grid environment one must keep in mind the new features of Grid systems such as its dynamic nature and the high degree of heterogeneity of jobs and resources.

The main problem which dealt in the dissertation is how to distributing the jobs when the payload, importance, urgency, flow time, dynamically keeps on changing as the grid expands or is flooded with number of job requests. So in this research work, we proposed a scheduling algorithm "Bragged regression tree algorithm for dynamic distribution and scheduling of jobs" that takes into consideration the heterogeneity of jobs.

This paper is organized as follows: Section 2 describes the present work. Authors Section 3 presents a proposed methodology. Section 4 provides implementation details. Section 5 presents results and discussion. Section 6 presents the conclusion and future scope of this paper. Thus at last the acknowledgement and references are presented.

2. RELATED WORK

In the present work we will discuss the papers related to my work. In [1], N. Muthuvelu, et. al. presents a dynamic job grouping-based scheduling algorithm that groups the jobs according to processing power of the available resource. In this he evaluated a dynamic scheduling strategy that maximizes the utilization of Grid resource processing capabilities, and reduces the overhead time and cost taken to execute the jobs on the Grid. The algorithm reduces the total processing time and cost of the jobs, it also maximizes the resource utilization. However, the algorithm does not take the dynamic resource characteristics into account and it does not pay attention to the network bandwidth of the resources.

In [2] Ng Wai Keat, et. al. proposed a bandwidth-aware job-grouping based scheduling algorithm that schedules jobs in grid systems by taking into consideration of computational capabilities and the communication capabilities of the resources. It uses network bandwidth of the resources to determine the priority of each resource. These jobs are grouped in such a way that, it maximizes the utilization of resources. This algorithm minimizes the network latency, but the scheduling strategy is not ensuring that the resource having a sufficient bandwidth to send the group jobs within required time.

Moreover some of the scheduling algorithms do not consider the dynamic behavior of Grid resources and there characteristics such as [3]. Thus further going through various other papers such as in [6] author found there were some limitations such as the algorithm build in this paper is efficient in all ways except when user changes the priority at dynamic time. In paper [4] author found there were some drawbacks

such as in this algorithm resource are selected in FCFS order, there is no priority for selecting resources.

Therefore there is need to develop an efficient dynamic job scheduling algorithm that would take best decision about assigning of jobs to appropriate CPU's dynamically.

In this paper, we present a scheduling strategy which takes the advantage of decision tree algorithm to take dynamic decision based on the current scenarios and which automatically incorporates factor analysis for considering the distribution of jobs.

2.1 Research Objectives

The following are the objectives of the dissertation:-

i. To develop a framework for scheduling batches (jobs) in grid computing.
ii. To develop parameter models for taking scheduling decisions.
iii. Apply factorization for scheduling algorithm "Bragged Regression Tree Algorithm for Dynamic Distribution and Scheduling of Jobs".
iv. Run task scheduler and get the result verified.

3. Proposed methodology

The methodology used in this proposed work clearly explains what to be done to achieve the objectives defined above. This is done by designing the basic model of job scheduling and work flow model.

3.1 Job scheduling model

The Job scheduling model shown in Figure 1 explains the overall methodology we proposed to obtain our objectives or goals. In this scheduling model, first of all various metrics are calculated which are required for decision making of job scheduling which includes user preferences such as Importance and Urgency, Grid resource such as task payload, time parameters and CPU states and many more. Then our algorithm conducts a factor analysis to find out factors which are urgent and important for decision making analysis and then pass on these extracted factors to the regression tree algorithm. Then this algorithm basically develops multiple linear discriminate functions as per regression tree algorithm. Thus finally the algorithm will help us in taking the dynamic decision based on the current scenarios for scheduling the jobs to appropriate CPU's.

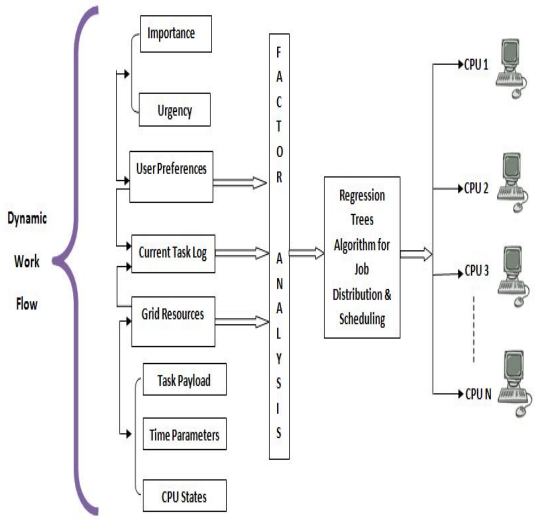

Figure 1 Shows the Job scheduling model

3.2 Work Flow Architecture

The work flow diagram shows the flow of research work to be done to obtain our objectives. In this flow diagram we first, set up Grid environment in which we will calculate batch task matrices such as user preference matrices. Then the log sheet is prepared which includes data set based on matrices and user preferences. Then do the factor analysis which will help in finding the important and useful factors on which variables are dependent. Then take the decisions based on factor analysis done in previous step. The decisions can be, to which CPU what kind of work is to be allotted. Then run the task scheduler and get the results. So this is basically shown in the work flow diagram below in Figure 2

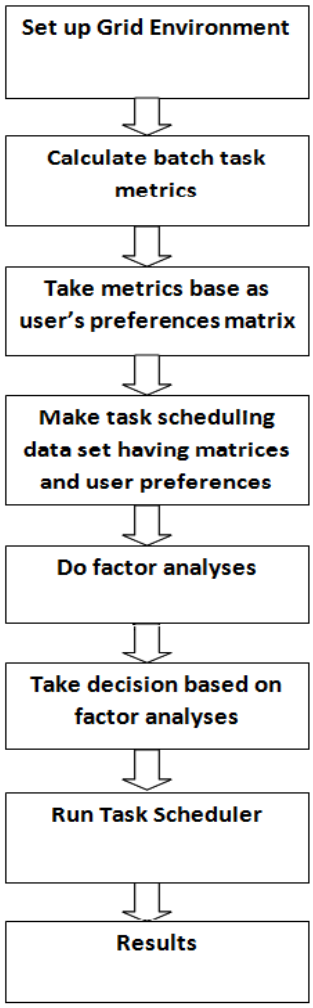

Figure 2 Shows Work Flow Architecture

4. IMPLEMENTATION OF PROPOSED METHODOLOGY

We need to consider some polynomial form of the equation, when we cannot represent in a linear form of equation that is Y= M(x) +C. In a function, whenever a trend line is made, there is no possibility of Y intercept having two values. But if an intercept will have two intercepts or more than that, then it is no longer a function as a definition. By plotting X or Y we only come to know that how X is behaving against Y. But we are unable to find some associative, casual, trivial, scientific facts between the two variables or simply we are unable to understand the nature of two variables. The two variables are

differentiated on the basis of who is dependent on whom and which is independent. So in that case we need to find the regression between two variables. This will help us to know the nature of the variables. The term nature here means, regression basically tries to develop the concept of discriminate function.

The biggest challenge is to design the discriminate function. In simple terms what should we do that, we are able to differentiate from large group of data points, from large domain of variables. If the relationship between X and Y is linear we call it linear discriminate function (LDA). So there are statistical tests like F-test, T-test, Chi-square test or R-square test which help us to determine the overall quality of our model (data points which are being discriminated. But when the relationship between X and Y is non linear and it is even difficult to represent in a polynomial from which may implies that, it has multiple Y intercepts and highly erratic, slow gradient. The best way to find the gradient /slope of polynomial data points which may even be spiral in nature is to run multiple linear discriminate functions. Based on which the multiple Y (predictor) intercepts.

Now here comes another problem. The problem is that, "on but basis multiple linear discriminate function must be run and for each discriminate function, how it must be designed or how it must be validated or tested to test the coefficient of determinant or goodness of the model".

Therefore there arises the need for developing classification trees which are non parametric and non linear in nature following certain if-then-else rules and there is no need for implicit assumptions that the underlined relationship between the predictor value and the dependent are continuous outcome. This model is particularly suited for doing decision making tasks where there is little prior knowledge nor any coherent set of theories or predictions regarding which variables are related and how.

The aim of doing classification and regression tree analysis is to run the algorithm in such a manner that it produces best possible predictive accuracy. Operationally, the most accurate prediction is defined as the prediction with minimum cost, for example if we have a job scheduling task, the classification tree tries to find based on its prediction, which is further based on minimum cost to the resources to which it wants to schedule the job. The notion of cost is basically a generalized way to represent best prediction having lowest misclassification rate which is the performance parameter for its run.

So, we have a choice of either minimizing the cost or minimizing the misclassification rate or both. This process is called designing the tree classifier. In this one of the important task is to select a right size tree as unreasonable big tree may consumes more computational resources and decisions hard to interpret due to enormous size of tree. Typically one should grow the tree size to find the optimal size of the tree. So the four steps involved in this computation are as follows:

1. Specifying the criteria for predictive accuracy,
2. Selecting splits
3. Determining when to stop splitting
4. Selecting the "right-sized" tree

4.1 Algorithm Description

In this research work we have proposed the job scheduling algorithm named "Bragged Regression Tree Algorithm for Dynamic Distribution and Scheduling of Jobs" as shown in Figure 3 below. This algorithm will include the factor analysis and regression analysis for scheduling the jobs to appropriate CPU's. The input to this algorithm is log sheet prepared by simulator, based on the factors taken into account such as make span, flow time, computational time and cost, user preference, number of resources, number of users etc. Thus the algorithm does factor analysis of the given input and gives the important and urgent list of factors as variables to regression analysis which further develops a decision tree. So therefore, this decision tree will tell us which job is to send to what CPU.
Calculation of classification decision tree is as follows:

Data points calculated before uses if –Then – Else rules which will help in sending the job to the appropriate CPU. Thus these rules play vital role in job scheduling algorithm. These can be explained as follows:

If DP >= Threshold Value 1 <=Threshold value 2 then send the job to CPU 1
End
If DP >= Threshold value 3 <= Threshold value 4 then send the job to CPU 2
End
If DP >= Threshold value 4 <= Threshold value 5 then send the job to CPU 3
End
If DP >= Threshold value 5 <= Threshold value 6 then send the job to CPU 4
End

Before we approach the test set, we'll see how well the model has classified the training set. This is not a good estimate of the accuracy the model would have on a test set; it is biased because the classification is being done on the same samples used for training. However, it is an indicator of whether a successful classification of test data may be possible.

Algorithm of Proposed Methodology

```
For each Feature,
        Do factor analysis
            For each top ranked Feature
                Develop a Feature Row.
                Develop a Resource Matrix.

        For each Resource Matrix
            Label each Feature Row with CPU number
        End

    For each Feature Row and label
        Find upper and lower thresholds.

        If there are more sets of lower and upper thresholds

            Develop decision tree

            Else

            Develop decision based on upper and lower
                thresholds for each label and row.
        End
    End
End
```

Figure 3 Shows the Algorithm of Proposed Methodology

5. RESULTS AND DISCUSSION

The results of our proposed algorithm are evaluated according to which the dynamic distribution of jobs to the CPU's is decided. That is decision tree will clearly define which job is to be send to what CPU based on if then else rules applied in classification tree.

The Figure 4 below depicts how each CPU is getting which job or in simple terms, what is the probability against the predicted class or CPU's. It is found that when probability lies between 0.1 and 0.8 the jobs are scheduled to CPU 1. If the probability lies between 0.1 and 0.9 then the jobs are scheduled to CPU 3. But if the probability lies between 0.0 and 0.1 then jobs are scheduled to CPU 4. So at last when probability is found between 0.7 and 1 then jobs are scheduled to CPU 2. This result clearly shows us how the group of jobs is scheduled to different CPU's.

As well as the predicted classes, the predict function outputs a second result, the scores. The score of an observation on a class is the probability that it comes from that class, as indicated by the model. These probabilities are used by the model when making classifications; the class with the highest probability is the one predicted. The scores can be used to give a measure of the confidence the model has in a particular prediction; if a sample has a much higher probability of being one class than all the others, than the prediction will be fairly solid. But if the predicted class is only a little more probable than all the others, the prediction is more shakier.

The scores can be conveniently displayed in a parallel coordinates plot. The four values on the x-axis are the four classes into which samples can be classified. Each sample is represented by a line, and the height of that line at each x-axis value gives the probability that the sample comes from that class. It is clear that all the samples are being predicted with high confidence.

The Figure 5 below shows the structure of decision tree applied to the data points given as inputs, which are the log entries to the grid. Based on If then else rules the figure also shows, the decision making points for identifying the various CPU's based on the feature given to it, which includes the input parameters. So this figure plays very important role in taking the decisions that which job is to be sent to what CPU.

The Figure 6 below shows the decision tree for classification. Moreover, it displays a confusion matrix, which contains a count of the samples, broken down by their actual and predicted classes. The rows represent the samples' actual class and the columns the predicted class. Numbers on the diagonal indicate a correct prediction; off-diagonal entries indicate a misclassification. The model is successfully classifying all samples in the training set, which is good but not unexpected.

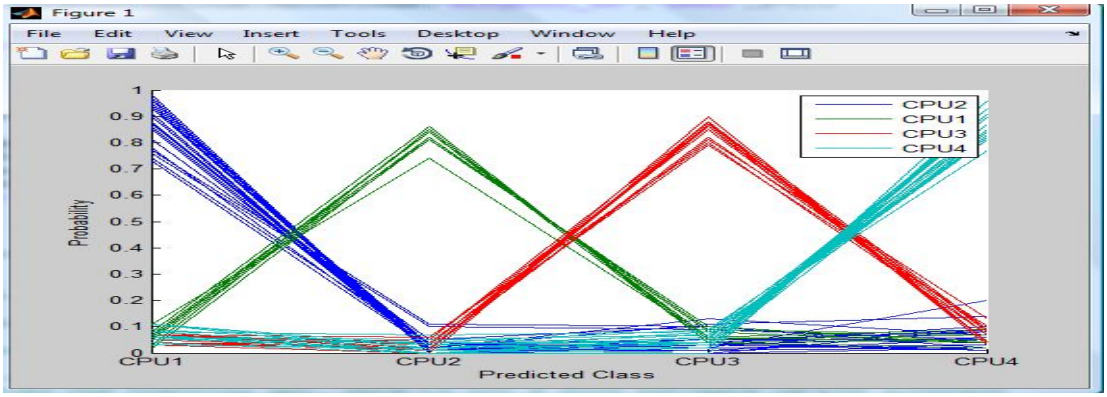

Figure 4 Shows the Probability against the predicted class.

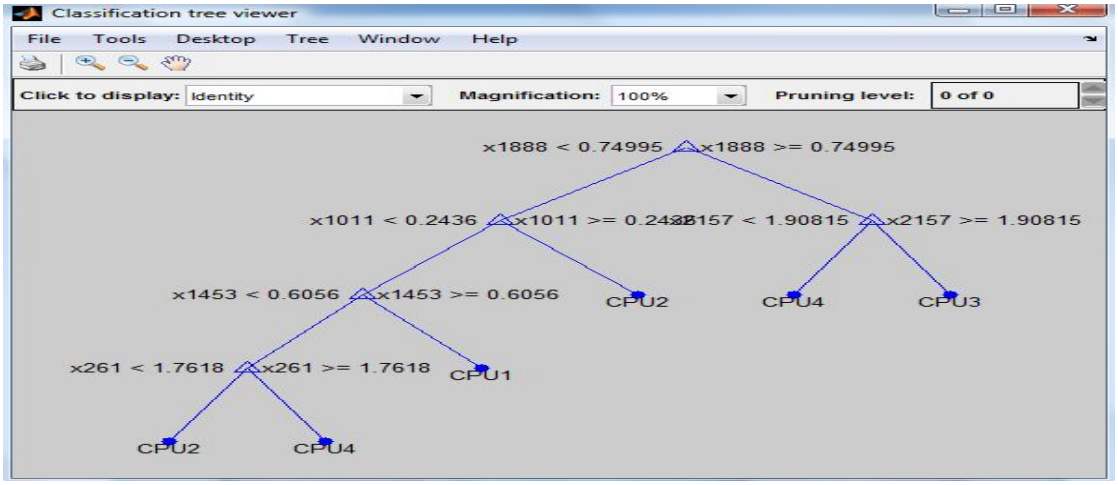

Figure 5 shows the structure of decision tree applied to the data points given as inputs.

```
Decision tree for classification
 1  if x1888<0.74995 then node 2 elseif x1888>=0.74995 then node 3 else CPU2
 2  if x1011<0.2436 then node 4 elseif x1011>=0.2436 then node 5 else CPU2
 3  if x2157<1.90815 then node 6 elseif x2157>=1.90815 then node 7 else CPU4
 4  if x1453<0.6056 then node 8 elseif x1453>=0.6056 then node 9 else CPU1
 5  class = CPU2
 6  class = CPU4
 7  class = CPU3
 8  if x261<1.7618 then node 10 elseif x261>=1.7618 then node 11 else CPU2
 9  class = CPU1
10  class = CPU2
11  class = CPU4
```

	CPU2	CPU1	CPU3	CPU4
CPU2	23	0	0	0
CPU1	0	8	0	0
CPU3	0	0	12	0
CPU4	0	0	0	20

Figure 6 Shows the Decision tree for classification

6. CONCLUSION AND FUTURE SCOPE

In this thesis we have discussed about the problem of job scheduling in heterogeneous grid where basic issue was how to distribute the jobs when the payload, importance, urgency, flow time etc. dynamically keeps on changing as the grid expands or is flooded with number of job requests. So we have proposed a scheduling algorithm named "Bragged Regression Tree Algorithm for Dynamic Distribution and Scheduling of Jobs" which takes the advantage of decision tree algorithm to take dynamic decisions based on current scenario and which automatically incorporates factor analysis for considering the distribution of job. In this research we have taken 4 CPU's and close to 20 parameters from wide choice of 34 parameters. So finally we came to conclusion that, decision tree should be taken into account which will dynamically decide which jobs are to be sending to what CPU's based on nature of job. Moreover by using this algorithm we are able to handle the dynamic job distribution task in heterogeneous environment very effectively with proper utilization of resources and reduction of total computational cost. .

6.1 Future Scope

The proposed approach has been analyzed and can be refined further as a more improved job scheduling algorithm can be considered in which some more factors could be included such as, energy parameters etc. Moreover some QOS requirements can also be taken into account. Furthermore it can't reflect the real computational grid environment that promotes further research in proposed work

7. ACKNOWLEDGMENT

I would like to thank my guide Er. Navneet S. Randhawa, Asstt. Professor and Head, Department of Information Technology, Adesh Institute of Engg. And Technology, Fridkot, India for his valuable assistant in the research work. I am highly grateful to Dr. Vikas Chawla**,** Director-Principal, Adesh Institute of Engineering & Technology, Faridkot, for providing this opportunity to carry out the present thesis work. I would also like to express gratitude to Er. Amit Makkar**,** Asstt. Professor & Head, Department of Computer Science & Engg., Dr. Urvinder Singh**,** Asstt. Professor, Department of Electronics and Communication Engg. and to other faculty members of CSE/IT Deptt., for their intellectual support throughout the course of this work and for providing valuable support in this research work.

8. REFERENCES

[1] Nithiapidary, M., Junyang, L.,Na L. S., Srikumar, V., Anthony, S., and Rajkumar, B. 2005. A dynamic job grouping-based scheduling for deploying applications with fine-grained tasks on global grids. In Proceedings of the 2005 Australasian workshop on Grid computing and e-research.

[2] Ng W. K., Ang T. F., Ling T. C., and Liew C. S. 2006. Scheduling framework for bandwidth-aware job grouping-based scheduling in grid computing. Malaysian Journal of Computer Science.

[3] T.F. Ang and W.K. Ng. 2009. A bandwidth-aware job scheduling-based scheduling on grid computing. Asian Network for Scientific Information.

[4] Quan L., Yeqing L. 2009. Grouping based fine-grained job scheduling in grid computing. In Proceedings of the 2009 First International Workshop on Education Technology and Computer Science.

[5] Raksha S., Vishnu K. S., Manoj K. M., Prachet B. 2010. A Survey of Job Scheduling and Resource Management in Grid Computing. World Academy of Science, Engineering and Technology.

[6] Saeed F. 2009. Efficient Job Scheduling in Grid Computing with Modified Artificial Fish Swarm Algorithm. International Journal of Computer Theory and Engineering.

[7] Jia Y., Rajkumar B., Kotagiri R. 2008. Workflow Scheduling Algorithms for Grid Computing. In Metaheuristics for Schdeuling in Distributed Computing Environments.

[8] Fangpeng D., Selim G. A. 2006. Scheduling Algorithm for Grid Computing: State of the Art and Open Problems.