International Journal of Computer Applications (0975 – 8887)
Volume 168 – No.9, June 2017

Scheduling Tasks in Heterogeneous System using Load Balancing Algorithm

Pratiksha Patil
Dr. D.Y.Patil School of Engineering and Technology, Lohagoan, Pune

Roshani Ade, PhD
Dr. D.Y.Patil School of Engineering and Technology, Lohagoan, Pune

ABSTRACT

Innovative idea of distributing the tasks to their best processor to reduce the execution time of task by using various scheduling techniques is given in this paper. This paper presents hybrid scheduling techniques which provide better solution of scheduling task that means combination of different scheduling provides better performance without degrading the result quality. Scheduling algorithms such as MinMin+, MaxMin+ and Sufferage+ are suitable for overcomes the drawback of previously used scheduling methods such as MinMin, MaxMin and Sufferage as well as scheduling in this paper provides better complexity as compare to previous scheduling methods. This scheduling are also suitable for heterogeneous environment more effectively to execute different set of task on different processors with different configurations. To get better the show of the existing system we have to improve existing algorithm with the load balancing.So same load should overloaded to all processors.The future algorithm have implement with detailed pseudocodes.

Keywords

Task Scheduling, MinMin, MaxMin, Sufferage, Standard Deviation, Load Balancing.

1. INTRODUCTION

Distribution of large application into task for faster processing is one of the important process in the area of distributed systems. Although many types of resources can be shared and used in a distributed system, usually they are accessed through an application running in the network. Normally, an application is used to define the piece of work of higher level in the system. An application can generate several tasks, which in turn can be composed of subtasks; this system is responsible for sending each subtask to a resource to be solved. It performs an important step of mappings task to different machines based on the expected execution time. Normally an application is used to define the piece of work of higher level in heterogeneous environment.

Since this application can generate several number of jobs that can be divided into subtasks and provided to different processors that should get completed within minimum time so that the processor use can be made to assign different task. Makespan is one of the most important term in case of mapping task to their processors using different scheduling. Makespan is nothing but turnaround time that is maximum of completion time. An optimal schedule will be the one that minimizes the makespan [1, 2].

Large numbers of task scheduling algorithms are available to minimize the makespan. All these algorithms try to find resources to be allocated to the tasks which will minimize the overall completion time of the jobs. The simple well-known existing algorithms used for scheduling are MinMin and Maxmin and sufferage. These algorithms work by considering the execution and completion time of each task on the each available grid resource. Scheduling is considered to be an important issue in the current distributed system existing algorithms used for scheduling are Min-Min and Maxmin and sufferage. These algorithms work by considering the execution and completion time of each task on the each available grid resource. Scheduling is considered to be an important issue in the current distributed system scenario. The demand for effective scheduling increases to achieve high performance computing. Typically, it is difficult to find an optimal resource allocation which minimizes the schedule length of jobs and effectively utilize the resources. The three main phases of scheduling are resource discovery, gathering resource information and job execution. The choice of the best pair of jobs and resources in the second phase has been proved to be NPcomplete problem.

The existing scheduling algorithms provide the various techniques for assigning different task to different resources with minimum completion time. These existing scheduling algorithms can be divided into two classes that are Online mode and Batch mode scheduling. In online mode, a task is assigned to processor on its arrival to scheduler [1]. Wherein Batch mode scheduling tasks are not assigned to processor immediately instead they are collected in to set of tasks also called as Metatask that are examined for assigning at prescheduled times to different processors also called as mapping events. Since in this system, batch mode is used in very efficient way for mapping different independent tasks to processors. Also these existing algorithms can be applied for heterogeneous environment effectively.

The proposed system in this work contains various scheduling methods along with hybrid technology such as Minmin+, Maxmin+, and Sufferage+. Hybrid technology involves combination of different scheduling methods to overcome disadvantages of minmin and maxmin. Overall, the scheduling algorithms aim to minimize the idle time and makespan of tasks. This paper also involves the concept of Load Balancing [2, 6], wherein, once scheduling of task is done using some scheduling the load balancing algorithm will take place to reschedule the task to utilize all the resources in the heterogeneousenvironment [5]. Each of this scheduling provides better performance and also decreases time complexity without degrading the solution quality.

To avoid the drawbacks of the existing scheduling algorithm, the proposed system algorithms are being used to enhance the system performance. Every one of the issues talked about in those techniques are taken and dissected to give a more powerful schedule. The calculation proposed in this paper beats every one of those calculations both in wording of makespan and load balancing. In this way a superior load balancing is achieved accomplished and the aggregate reaction time of the framework is made better. The proposed calculation applies the Min-Min system in the first stage and afterward reschedules by considering the greatest execution time that is not exactly the makespan got from the first stage.

2. LITERATURE SURVEY

A distributed scheduling algorithm aims to increase the utilization of resources with light load or idle resources thereby freeing the resources with heavy load. The algorithm tries to distribute the load among all the available resources. At the same time, it aims to minimize the makespan with the effective utilization of resources.

In classical distributed systems comprised of homogeneous and dedicated resources, scheduling algorithms have been intensively studied. But these algorithms will not work well in Grid architecture because of its heterogeneity, scalability and autonomy. This makes load balanced scheduling algorithm for grid computing more difficult and an interesting topic for many researchers. The Non-traditional algorithms differ from the conventional traditional algorithms in that it produces optimal results in a short period of time. An option is to choose a proper planning calculation to use in a given heterogeneous environment on account of the attributes of the undertakings, machines and system heterogeneity resources. These scheduling algorithms will work well even for heterogeneous resources also. Heterogeneous systems provides with the facility of utilization of all available resources as load balancing concept that aims at keeping resources busy [5].

Opportunistic Load Balancing (OLB) is specially used to keep all the processors busy that is to make utilization of all the available resources by making task assignment in subjective order, to the next available processor, without considering task expected execution time on that particular processor but this result in poor makespan [6].

Minimum Execution Time (MET) assigns tasks to processor in arbitrary order with best expected execution time for that task, without taking into consideration processors availability.

Since assigning the task to its best processor provides better performance but causes severe load imbalancing and does not provide support for heterogeneous environment [4, 5].

Minimum Completion Time (MCT) assigns tasks to different processors in arbitrary order, with minimum expected completion time for that task. Since this causes some of the task to be assigned to the processor that do not have minimum execution time. For this purpose this minimum completion time is defined in a way that combines the benefit of both opportunistic load balancing (OLB) and minimum execution time (MET) to provide better performance of task mapping [3].

Min-Min algorithm starts with a set of all unmapped tasks. The machine with best execution time is selected. Then the job with the overall minimum completion time is selected and mapped to that resource. The ready time of the resource is changed. This process is occurs repeated until all the unmapped tasks are assigned. Compared to MCT this algorithm considers all jobs at a time. So it produces a better makespan. When selected task is assigned to the resource it is removed from the metatask that is set of task. Since this method provides easiest way to assign task to processor but has one drawback due to selection of task having minimum expected completion time, the task with largest expected completion time remains unassigned for longer time and also load is not balanced across the systems, due to which some resources remains idle and this also results in increase in makespan.

Max-Min is similar to Min-Min algorithm. The machine that has the MCT for all tasks is selected. Then the job with the overall maximum completion time is selected and mapped to that resource. The ready time of the resource is changed. This process occurs repeatedly until all unmapped tasks are assigned. The idea of this algorithm is to reduce the wait time of the large jobs. Since this scheduling algorithm provides the way of mapping task to its best machine with longer execution time prioritize current task to complete concurrently with the task that having shorter execution time. Since this mapping of task to resources is better than minmin scheduling wherein the task with smaller execution time is selected and assigned to the available resource for execution and then task with longer execution time are executed while several machines sit idle. Since maxmin scheduling provides better load balancing across machines as well as better makespan [6, 7].

Sufferage scheduling differs with previous scheduling algorithms in the sense of task selection process. Like minmin and maxmin it also begins with set of unassigned task that has minimum completion time i.e. sufferage scheduling is also based on the concept of minimum completion time, since it differ from the previous scheduling in the sense it selects and assigns the task to the processor on the basis of sufferage value and not minimum or maximum completion time. Since it computes second MCT value instead of computing MCT value for each task and calculates sufferage value which is defined as difference between MCT and second MCT of a task is considered. This scheduling selects the task with largest sufferage value and assigns it to available resource. Thus sufferage scheduling differs from minmin and maxmin scheduling in the task selection policy [4, 5].

3. ENHANCED SCHEDULIN METHODS

3.1 Minmin Scheduling

Minmin+ scheduling uses different methods such as MinMin+Select function invokes a MIN(Qk) operation on each priority queue Qk to find candidate task for corresponding processor Pk. The hopeful undertaking Ti chose for processor Pk is viably the errand that will expand the present culmination time of Pk by the littlest sum if Ti is alloted to Pk. For every processor Pk, the execution time of the hopeful errand Ti on Pk is added to ek to register the upgraded ek esteem for Pk if Ti is allocated to Pk. A running min calculation performed over these K updated ek values gives the minimum MCT for current loop as well as the taskto-processor assignment (i’, k’) that has this minimum MCT value. At the end of each iteration of the main loop, the assigned task Ti’ is deleted from all priority queues. For the implementing this priority queue two alternatives are being considered that are binary heap and sorted linear array, and also some operations are being used like sorting operation, deletion operation, and necessary check is made on the queue to know which task has not being yet assigned to available resource that is with minimum completion time. Hence the overall running time complexity is reduced.

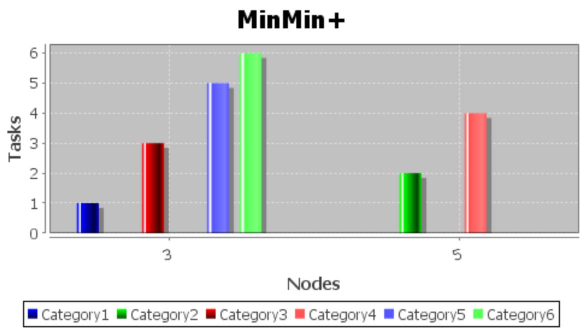

Fig2. Assigning task to processor using MinMin+ Algorithm

As shown inthe figure tasks T1,to T6 is assigned to the 6processor P1 to P6 with minimum load and makespan by using different datasets.The datasets used for assigning task to processor with MinMin+ is Link Extraction,Code Extract ion,DisplayStreemerAlgorithm,Word Extraction.

3.2 Maxmin+ Scheduling

The solution quality obtained in the earlier loops is not proper due to the late assignment of very huge tasks. In MaxMin scheduling, the larger tasks are assigned in previous loops, but not always to their desired processors. Since, in the first initial loops of MaxMin, the first loop assigns the task that is major to its desired processor. It is understood that the second major task has same desired processor as the major task. In the second loop, the task selection policy of MaxMin prevents the allocation of the second major task to its desired processor. For the next loop, the third major task loses the elasticity to get assigned to the desired processors of the largest two tasks and so on. To improve the drawbacks of the MinMin and MaxMin scheduling, these algorithms are united under a hybrid scheduling, which referred to as MaxMin. Like MinMin and MaxMin, the MaxMin+ booking includes a fundamental circle that doles out a chose undertaking to a processor at every emphasis. Within each loop, the scheduling first computes a task to processor allocation according to the MinMin scheduling. The computed assignment is realized only if makespan is unaffected in the previous iteration. If, however, the calculated allocation results in increase of makespan, the task-to-processor allocation is recalculated with respect to the MaxMin scheduling. This scheduling overcome drawback of maxmin scheduling of task assignment problem to same processor by doing the combination of maxmin with minmin+ under a hybrid scheduling that is maxmin+.

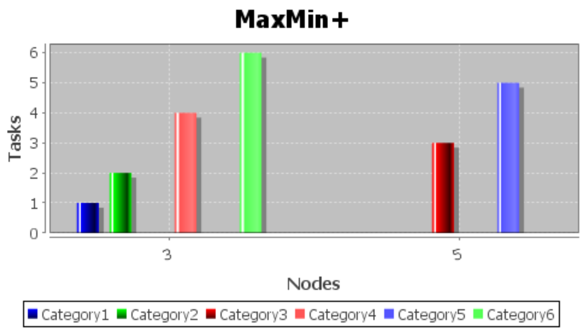

Fig2. Assigning task to processor using MaxMin+ Algorithm

As shown inthe figure tasks T1,to T6 is assigned to the 6processor P1 to P6 with minimum load and makespan by using different datasets.The datasets used for assigning task to processor with MaxMin+ is Link Extraction,Code Extract ion,DisplayStreemerAlgorithm,Word Extraction.

3.3 Suffferage+ Scheduling

The main idea behind the Sufferage+ scheduling is to perform critical assignment decisions by Sufferage so that the solution quality is not degraded. With this approach, execution time of Sufferage is decreased with a small potential degradation in the solution quality. Since Sufferage+ working is similar to sufferage scheduling since to make applicable sufferage scheduling to large datasets it is combine with minmin+ scheduling under a new scheduling that is sufferage+. As in MaxMin+, in this scheduling also the MinMin+Init function performs the necessary initializations. It computes the assignment according to MinMin+. The comparison operation checks if the computed assignment use affects makespan. Then this algorithm calculates the task-to-processor allocation according to Sufferage. This scheduling differs from previous scheduling in the sense that when assignment is computed, lead to increase in makespan of previous iteration otherwise assignment is recomputed using sufferage scheduling.

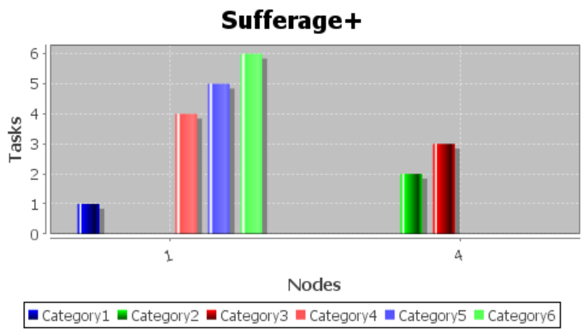

Fig3. Assigning task to processor using Suff+ Algorithm

As shown inthe figure tasks T1,to T6 is assigned to the processor P1 to P6 with minimum load and makespan by using different datasets.The datasets used for assigning task to processor with Suff+ is Link Extraction,Code Extract ion,DisplayStreemerAlgorithm,Word Extraction.

3.4 Switchers cheduling

As the name indicates it is the combination of different scheduling also known as hybrid scheduling. Switcher scheduling is based on concept of standard deviation value comparison with threshold value. Based on this it switches between scheduling that is if standard deviation [6] value is less than threshold value than asks are considered to be with minimum execution time and minmin scheduling is applied to assign the tasks to available resources, otherwise maxmin scheduling is used to assign the tasks to available resources.

This process is repeated until all the tasks are assigned to their respective available resources [3, 13]. Standard deviation concept is specially used for hybrid scheduling that is combination of different scheduling. Wherein, the standard deviation value is compared with threshold value to check which scheduling to be applied for mapping of task to different resources [6]. Since the standard deviation value is calculated on the basis of average of completion time of all tasks, as mention below:

$$avgCT = \frac{\sum_{i-1}^{s} CT_{ij}}{s}$$

Where avgCT denotes average of completion time that is sum of all completion time of given tasks and s is nothing but index of task. Using this average value standard deviation is calculated as:

$$sd = \sqrt{\frac{\sum_{i-1}^{s}(CT_{ij} - avgCT)^2}{s}}$$

Based on above mentioned formulae standard deviation is calculated. Since, this is compared with the threshold value in case of hybrid scheduling wherein the multiple scheduling are called by algorithm alternatively for mapping task to processors. This hybrid scheduling will use standard deviation concept wherein if the calculated standard deviation is less than threshold value then that particular task is assigned using the minmin scheduling to available resource otherwise the task is assigned to available resource using maxmin scheduling. Since after the assignment of task to resource it will be deleted from set of task that is metatask and the hybrid scheduling will repeat all the process until all the task are assigned to processors.

There are many different types of hybrid algorithms that call alternatively different scheduling for mapping tasks to their best processors. This type of scheduling also maintains the proper load balance across the processors due to which all the available resources get fully utilized and no resource remains an idle.

4. LOAD BALANCING IN DISTRIBUTED SYSTEM

To better utilize the machines and to minimize the machine idle time, the load should be balanced. Task size must be considered in order to balance the load. In a heterogeneous system, execution time of a task varies on different machines. The average execution time of a particular task over the entire machines can give task size. Task assignment depends on priority based on sizes. Priority can be set depending on task size. According to experiment, mapping of smaller tasks leads to load imbalance. Also, larger task allocation gives complete load balance. Scheduling mentioned above like minmin and maxmin maps different tasks to different available resources efficiently but it does not maintain proper load balancing among the resources due to which some resources are utilized and some remain idle. This load balancing concept can be applied to this scheduling to get done execution faster [8].

Minmin scheduling selects tasks with minimum completion time and allocates it to available resource, due to which task with longer execution time remains unassigned although the resource is available that causes resources to remain idle. Similarly in maxmin scheduling tasks with maximum completion time is selected and assigned to processor, due to which smaller tasks are assigned after long time to available processors [9]. Solution for above is to apply load balancing concept with this types of scheduling.

This can be done when tasks are assigned to their resources that is once minmin scheduling is applied to assign tasks to available resources using minimum completion time, the load balancing method is applied again on this assigned task for rescheduling it i.e. it may happen that minmin scheduling will use some resources to assign task and some remain idle then load balancing method will select the task with maximum completion time that will be less than makespan produced by Minmin scheduling and reschedule it to the resource that is available and not utilized yet so that execution of tasks will be more faster[11].Other tasks maximum completion time is not less than makespan. So whichever task has maximum completion time less tan makespan is selected and rescheduled to available resource.

The time required to execute a task is always dependant on the load of that system.The more the load of the system, the more would be the time taken to execute the task and vice versa. Hence,time required for every task to perform on every processor is dependent on load on that processor.

Load on each processor α Time required for task to perform on that processor

From above equation it is clear that for execution of any task is completely dependent on the load on the processor. More tasks is allocated to a processor which has fewer loads on it.Hence from existing systems derived that task allocation is dependent on both implementation time and load on processor. But considering our equation system will consider only one parameter that is load.

System calculates load on processors when no tasks is running.Also system want to determine load on processors when tasks will be running.From the later system will calculate average load.

4.1 Architectural Design Of system

A block diagram is a specialized, high-level type of flowchart. Its highly structured form presents a quick overview of major process steps and key process participants, as well as the relationships and interfaces involved. A block diagram is a useful tool both in designing new processes and in improving existing processes. In both cases the block diagram provides a quick, high-level view of the work and may rapidly lead to process points of interest. Because of its high-level perspective, it may not offer the level of detail required for more comprehensive planning or analysis. To construct a block diagram we must have a clear understanding of how the process operates.

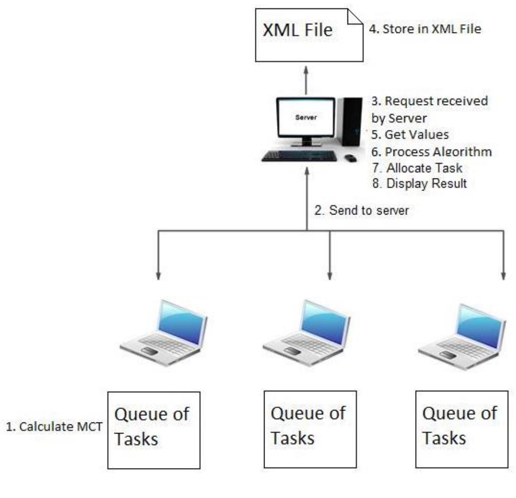

Fig: Architectural view of System

Above figure shows the block diagram of system. It mainly contains a individual server serving a client user. A client has a set of task which he needs to divide and assign to other clients. Client sends request to server to assign the tasks. Server stores it in XML files and processes the tasks division by the algorithm. In this system there is three algorithmMinmin+, Maxmin+, and Sufferage+. After dividing the tasks server assigns the tasks to each clients.

Table1:Notations Used For Algorithm

$Task_k$	Task for kth processor
$Pload_i$	Load on ith processor.by default it is the load when no task is running
$BackUp1_i$	Backup of load on ith processor.by default it is load when no tasks is running
N	Total tasks
L	Total Processors
AP	Assigned Processors.

4.2 Algorithmic Steps

The proposed system consists of following algorithmic steps:

```
1.  Fori←0 to N
2.  Min=∞;
3.  Forj←0 to L
4.  If($Pload_j$+$task_i$) < minthen
5.  $Pload_j$ = $Pload_j$+$task_i$
6.  Min = $Pload_j$
7.  $AP_i$ = j;
8.  b = j;
9.  fork ← 0to j
10. Ploadk = $BackUp1_i$
11. End for
12. End if
13. End for
14. BackUp $1_i$ = $Pload_k$
15. End For
```

Initially there are no tasks running on any processor.hence Pload will contain loads of processor with no tasks running on it. Minimum value is set to infinity. While assigning task it will be assumes if task is allocated t ith processor (step3) then current load will be summation of current load and average load of that particular task and processor will be assigned. But on further processing step 3 may get satisfied for other processor for same task. In that condition task is assigned to that new processor. But the previous changes in the load are restored(step9).

5. EXAMPLES OF LOAD BALANCING IN DISTRIBUTED SYSTEM

Consider a heterogeneous environment with two resources R_1 and R_2 and metatask that contain four different tasks T_1, T_2, T_3 and T_4 as shown below in table1 that contains tasks, resources along with expected execution time for mapping tasks to their respective resources.

Table 1: Resources and Tasks with Expected Execution Time

Tasks	Resources	
	R_1	R_2
T_1	7	2
T_2	13	4
T_3	14	1
T_4	11	3

As shown in above table task assignment is done to different processors using minmin scheduling is done in following way.

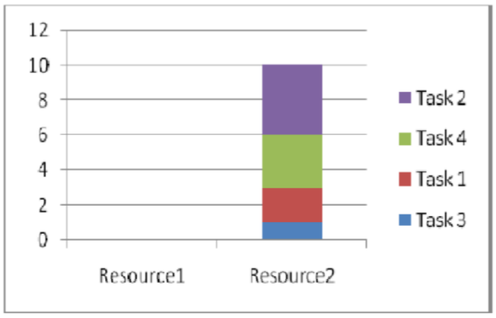

Figure 1: Mapping of Tasks to Resources with Minmin Algorithm

As shown in figure1 minmin scheduling will select tasks according to given execution time so task T3 will be assigned first to Resource R2, then task T1 will be assigned again on resource R2, then task T4 will be assigned again on resource R2 and finally the remaining task that is task T2 will also be assigned to resource R2 only according to given expected execution time in Table1. Since on resource R2 task completion is faster than on resource R1, so all the task will be assigned on resource R2 only. Once minmin scheduling is applied for mapping tasks to available resources, load balancing technique is applied for again rescheduling tasks to make utilization of idle resources that minimizes overall task completion time thatis makespan.

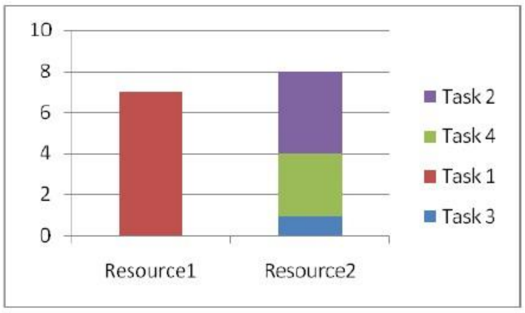

Figure 2: Rescheduling of tasks to Resources with Load Balancing method

As shown in figure2 task T1 is rescheduled to balance the load as it provides maximum completion time on resource R1 as shown in Table1 as well as it is less than makespan produced by minmin scheduling. While remaining tasks although have maximum completion time but are not less than makespan. So task T1 is rescheduled on resource R1 that results in better makespan as compared to minmin scheduling.

6. CONCLUSION

The aim of this paper was to present various scheduling methods like minmin, maxmin, sufferage, hybrid, load balancing techniques in the field of distributed systems. The scheduling like minmin and maxmin are suitable for small scale distributed systems but when number of tasks increases than these scheduling cannot schedule task appropriately that affects on makespan which relatively become large. To overcome limitations of these scheduling and make them applicable for large scale distributed systems, a new task scheduling algorithm like minmin+, maxmin+ and sufferage+ along with hybrid scheduling are used that also maintains proper load balancing across the systems. This scheduling uses advantages of minmin and maxmin and covers there disadvantages. This study can be further extended by considering task heterogeneity and machine heterogeneity.

7. REFERENCES

[1] E. Kartal Tabak, B. Barla Cambazoglu, and Cevdet Aykanat, "Improving the Performance of Independent Task Assignment Heuristics MinMin, MaxMin and Sufferage",IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 25, NO. 5, MAY 2014.

[2] T. D. Braun,H. J. Siegel,N. Beck, L. L. Boloni, "A comparison of eleven static scheduling for mapping a class of independent tasks onto heterogeneous distributed computing systems", J. Parallel Distrib. Comput., vol. 61, no. 6, pp. 810837, 2001.

[3] Kamali Gupta, Manpreet Singh, "Scheduling Based Task Scheduling In Grid", International Journal of Engineering and Technology (IJET), vol. 4, pp. 254260,Aug-Sep 2012.

[4] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen, and R.F. Freund, "Dynamic mapping of a class of independent tasks onto heterogeneous computing systems", J. Parallel Distrib.Comput., vol. 59, pp.107131, 1999.

[5] H. J. Siegel and S. Ali, "Techniques for mapping tasks to machines in heterogeneous computing systems", J. Syst. Archit., vol. 46, no. 8, pp. 627639, 2000.

[6] T. Kokilavani, Dr. D.I. George Amalarethinam, "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing", International Journal of Computer Applications, vol. 20, April 2011.

[7] George Amalarethinam. D.I, VaaheedhaKfatheen .S, "Max-min Average Algorithm for Scheduling Tasks in Grid Computing Systems", International Journal of Computer Science and Information Technologies, Vol. 3, pp.3659-3663, 2012.

[8] Balasangameshwara J. ,Raju N, "Performance-Driven Load Balancing with a Primary-Backup Approach for Computational Grids with Low Communication Cost and Replication Cost", IEEE Transactions on Computers,Volume:62, Issue: 5.

[9] Shah R, Veeravalli B. ,Misra, M, "On the Design of Adaptive and Decentralized Load Balancing Algorithms with Load Estimation for Computational Grid Environments", IEEE Transactions on Parallel and Distributed Systems,Volume:18,Issue: 12.

[10] He. X, X-He Sun, and Laszewski. G.V, "QoS Guided Minmin Scheduling for Grid Task Scheduling", Journal of Computer Science and Technology, Vol. 18, pp. 442-451,2003.

[11] T. D. Braun,H. J. Siegel,N. Beck, L. L. Boloni, "A comparison of eleven static scheduling for mapping a class of independent tasks 810837, 2001.

[12] Sameer Singh Chauhan,R. Joshi. C, "QoS Guided Scheduling Algorithms for Grid Task Scheduling", International Journal of Computer Applications (09758887), pp 24-31, Volume 2, No.9, June 2010.

[13] Singh. M and Suri. P.K, "QPS A QoS Based Predictive Max-Min, Min-Min, Switcher Algorithm for Job Scheduling in a Grid", Information Technology Journal, Vol. 7, pp. 1176-1181, 2008.

[14] Yagoubi. B, and Slimani. Y, "Task Load Balancing Strategy for Grid Computing", Journal of Computer Science, Vol. 3, No. 3, pp. 186-194, 2007.