A Deadline based Task Scheduling Algorithm for Heterogeneous Grid Environments

Rahul Sharma
Department of C.S.E
GBPEC, Pauri Garhwal,
Uttarakhand, India

S. K. Verma
Department of C.S.E
GBPEC, Pauri Garhwal,
Uttarakhand, India

ABSTRACT
Grid computing is the framework of computer systems that provides high performance computing environment. The challenging issue in grid computing is to design efficient and reliable task scheduling algorithms for efficient utilization of grid computing. In this paper, we are proposing a new Improvised Prioritized Deadline (IPD) based scheduling algorithm for efficient task execution with deadline constraints of users' tasks. The proposed algorithm considers the processing power of the resources while scheduling the tasks. Performance comparison of the algorithm has been done with the other task scheduling algorithms such as Earliest Deadline First (EDF) and Prioritized Based Deadline Scheduling Algorithm (PDSA). The proposed algorithm improves 45%-70% with respect to the average tardiness over the PDSA algorithm. The proposed algorithm also shows good results with respect to the number of non-delayed tasks. In the cases the purposed improvised algorithm has shown good results.

Keywords
Grid Computing, Non-Delayed Tasks, Processing Power, Tardiness, Task Scheduling..

1. INTRODUCTION
Interconnection of geographically distributed computer resources from multiple administrative domains gave a concept of virtual computing called Grid [1]. The main objective of grid computing environment is to combine the computing power involved, with widely distributed resources, as well as to deliver non-trivial services to users. A grid environment collects, integrates and uses heterogeneous or homogeneous resources scattered around the globe by a high-speed network. Over the past decade, the Grid has emerged as a promising platform to address large-scale applications in various fields [2]. However, the grid computing environment is heterogeneous, as the resources present in the grid varies on operating systems, communication bandwidths and the processing abilities. To use the grid resource efficiently, the resource scheduling is the key to the problem which turns into a fundamental issue in grid computing.
Effective and efficient task scheduling algorithms is implemented to better use of the capabilities of the grid computing environment [3]. Scheduling tasks on Grids is an important issue for optimizing the Grid resource allocation. Scheduling tasks on grids is a challenging issue due to the dynamism and the heterogeneity of the resources. In recent years, the researchers have proposed several efficient scheduling algorithms that are used in grid computing to allocate grid resources with a special emphasis on task scheduling [4]. The proposed algorithm is based on the deadline constraints of the tasks and the expected execution time of the task in this paper.

The presented algorithm is an improvement over existing Prioritized Deadline Based Scheduling Algorithm (PDSA) [12] called the Improvised Prioritized Deadline (IPD) based scheduling algorithm. It has considered the task deadline constraint associated with the task for its execution. Many grid users are highly interested in the timely execution of the tasks under the given deadline constraints. Most of the existing scheduling algorithms have not considered deadline perspective for task execution. To evaluate the performance of the scheduling algorithms we have used synthetic workload traces.

The rest of this paper is organized as follows. Section 2 gives an overview on previous researches in the field of task scheduling. Section 3 discusses the system design and implementation details of our improvisation on the task scheduling algorithm respectively. Section 4 describes experimental results which are evaluated by using synthetic workload traces and section 5 concludes the paper.

2. RELATED WORK
Intensive research has been conducted over the past years in grid task scheduling field to solve the problem of mapping a set of tasks to a set of machines [5]. It has been proved that the scheduling problem is an NP complete problem. The mapping criteria are mainly classified into two modes which are online mode and batch mode. In the online mode a task is mapped to the resource as soon as it arrives at the scheduler on the other hand, in the batch mode mapping, a set of tasks is made called the meta-task. Mapping of meta-task is performed at prescheduled times called mapping event [6]. Many algorithms have been proposed by various researchers to schedule the tasks in grid environment. The selection of the algorithm for scheduling the tasks in grid environment is the most critical due to performance major of the grid. The selection also depends on the type of the tasks, number of resources and other constraints like the deadline of the task, processing speed of the processing element, bandwidth of the communication network. Based on these constraints suitable algorithm is used for scheduling the tasks. The various Performance metrics are used to evaluate the results of one algorithm with other existing algorithms like, makespan, tardiness, resource utilization, response time and many more depending upon the scenarios for which the algorithm have been designed.

An algorithm based on combinational backfilling strategy is proposed in [5]. This algorithm selects multiple tasks combined from the waiting task queue to backfill for

maximizing the use of idle resources. This algorithm attains a lower average waiting time of tasks and higher utilization of resources than existing representative backfilling algorithms. The [6] proposed task scheduling algorithm based on task grouping concept. In this strategy the task grouping is based on the memory constraint together with other constraints such as processing power, bandwidth, expected execution and transfer time. These vary constraints are taken at task level rather than at group level. The experimental results demonstrated that this proposed scheduling algorithm efficiently reduces the processing time of tasks. In [7] author presented a heuristic algorithm called Harder First Prior First (HFFP). The algorithm presented in this paper schedule tasks based on the characteristics such as processing time, release time and delivery time. HFFP can minimize the completion time of tasks, especially when the number of tasks is much larger than the number of resources.

In [8] author presented a strategy which is based on two scheduling methods Max-Min and Min-Min and named as the Weighted Mean Time Min-Min Max-Min Selective Scheduling (WMTS). The overall performance of the resources is also considered by this strategy while scheduling the sequence of tasks. This strategy performed better than Max-Min and Min-Min scheduling strategies. In the work of [9], a cost-based workflow scheduling algorithm was presented. This strategy aims at minimizing the cost of execution while reaching the deadline. While in [10], the fairness problem is dealt by dropping the service time frame error. Sufficient computational power is assigned to each task so that it can be completed within its deadline. However, it will be more optimize if priority is given based on the minimum time of execution of tasks. In [11] author discusses policy-based scheduling techniques on heterogeneous resources. This proposed algorithm allocates grid resources to an application under the constraints presented with resource usage policies. First, the algorithm supports the resource usage constrained scheduling. The resources of the grid are controlled and owned by decentralized institutions. Second, optimization based scheduling is provided by the algorithm. It provides an optimal solution to the grid resource allocation problem. Third, the algorithm assumes that a set of resources is distributed geographically and heterogeneous in nature. Fourth, the scheduling method dynamically adjusts to the grid status. This algorithm tracks the current workload of the resources.

The algorithms discussed above do not consider the deadline constraint of the tasks. So these algorithms are not applicable to the grid environment where tasks have deadline associated with them. The tasks need to be completed within the deadline in order to produce useful results. The most common scheduling algorithm with considers the deadline of the tasks is Earliest Deadline First (EDF). EDF or least time to go is a dynamic scheduling algorithm. Earliest deadline based scheduling algorithm is one the simplest scheduling algorithm which is a type of priority scheduling. In this algorithm highest priority is given to the tasks with minimum deadline. It means that earlier the deadline of the task higher will be its priority. After completion of the previous task, the new task having minimum deadline value will be fetched from the ready queue. When the system is overloaded, the set of processes that will miss deadlines is largely unpredictable. EDF is not fair in the cases where the two tasks have the same absolute deadlines as it chooses one of the two at random. No other criterion is considered at that point. The flowchart of the EDF algorithm is given in the fig 1.

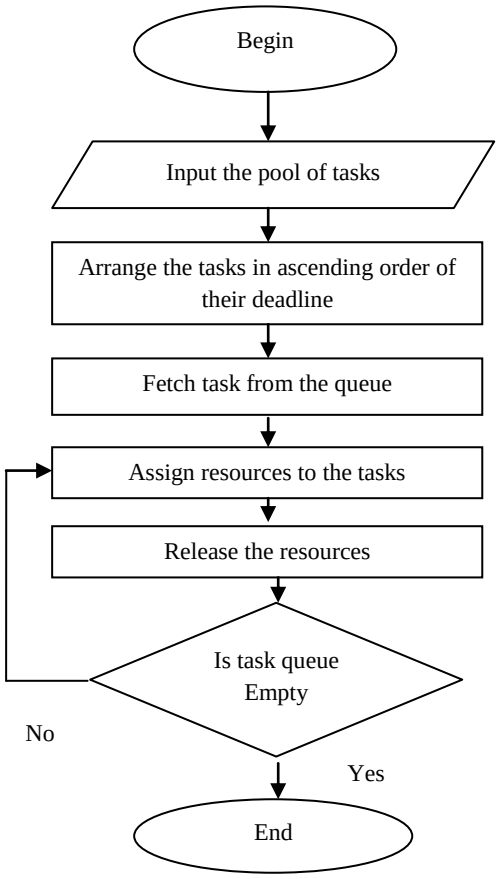

Fig 1: Flowchart of EDF Algorithm

The other scheduling algorithm which considers the deadline of a task is Prioritized Deadline Based Algorithm (PDSA). PDSA is also a dynamic task scheduling algorithm. This algorithm gives priority to the tasks according to their time delay. This delay is calculated as the difference between the deadline and the computational time of the task [11]. The task with the minimum time delay executed when the current task finishes off. The algorithm is based on the allocation of the task to uniprocessors. It means the task requesting only one processing element for its execution. This deadline based algorithm was able to give better results than the EDF. The flowchart of the PDSA is shown by the Figure 2.

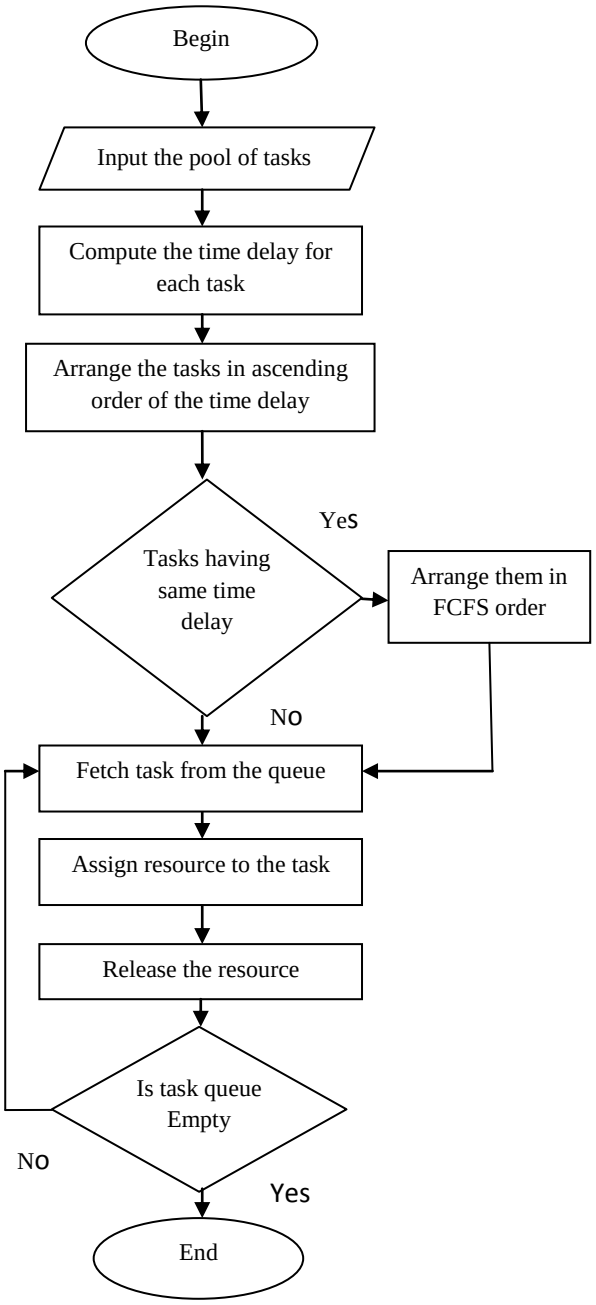

Fig 2: Flowchart of PDSA

3. PROPOSED TASK SCHEDULING ALGORITHM

Improved Prioritized Deadline (IPD) based scheduling algorithm is proposed in this paper. The proposed algorithm is an improved version of the Prioritized Deadline based Scheduling Algorithm (PDSA) [11]. In the system design the resources are ranked according to their total processing power, i.e., the product of the number of the processing elements and the processing power of each element. The processing speed of each processing element in one resource is same. The allocation of the resources to the tasks is based on the time delay which is the difference between the deadline of the task and the expected computation time of the task. Further, the list of resources is maintained for the allocation of the queued task based on their requirements. After making the list of the resources which are suitable for the task, the selection of the resource is done on the basis of the processing speed of the resource. The highest priority is assigned to the resource with the highest processing speed for the faster execution of task.

Let us assume T_i is the i[th] task, n is the number of tasks, a_i is the arrival time of task I, d_i is deadline of task I, ET_i is the expected execution time of task i, TD_i is the time delay of task i, TR_i is the tardiness of task I, Avg_TR is the average tardiness of the schedule and f_i is the finish time of the task i;

Time Delay is referred as the difference between the deadline of the task and the expected execution time of the task as defined in Eq-1.

$$TD_i = d_i - ET_i \quad (1)$$

Tardiness refers to the time delay between the finishing time of task and the deadline of the task as defined in Eq-2.

$$TR_i = d_i - f_i \quad (2)$$

Total Tardiness is the sum of the tardiness of the each task which did not get executed under the provided deadline. The average Tardiness is defined in Eq-3.

$$Avg\_TR = \frac{\sum_{i=1}^{n} TR_i}{n} \quad (3)$$

The number of non delayed tasks is the total number of tasks whose finishing time was less than the deadline of the task, i.e., which finished inside the deadline given to them. The expected completion time is calculated as the mean of the completion time for the task at every resource.

The algorithm takes the input from users, where as each task is described by its taskID T_i, arrival time a_i, expected execution time ET_i, computational length CL_i, deadline d_i and number of processors NP_i required. Then we compute the value of the time delay TDi for each task by using Eq. (1). The tasks in the ready queue are arranged in the ascending order based on the computed time delay (task with minimum time delay will be given priority) of the tasks. If the two tasks have a same computing delay, then the task will ordered on the basis of the first come first serve method in the ready queue. The tasks are executed according to their arrangement in the queue. For a task in the ready queue we will make a list of the suitable processors for the task. Then from that list we select the resource having the best processing speed. The task is then executed on that resource for the time depending on its computational length and the processing speed of the resource. The finishing time f_i of the task is calculated and the tardiness TR_i is calculated using Eq. (2). If there is no tardiness for that task then the number of non-delayed tasks is incremented. When all the tasks in the ready queue are finished then we calculate the average tardiness Avg_TR using Eq. (3). The flowchart of the proposed algorithm is shown in the fig 3.

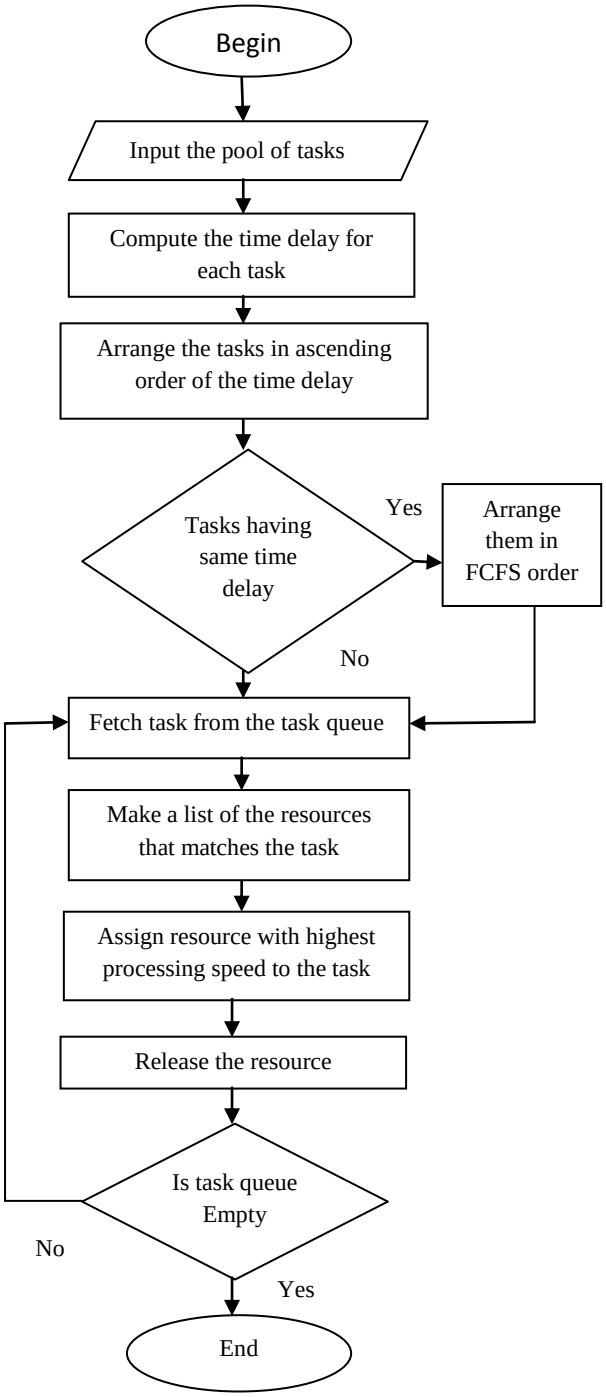

Fig 3: Flowchart of proposed algorithm

3. RESULTS AND DISCUSSION

A grid simulation tool based on the GridSim [13] toolkit is used for implementation of the proposed algorithm. The simulation tool used for this work is Alea 2.1 grid simulator [14]. The proposed scheduling algorithm is compared with Prioritized Deadline Based Scheduling Algorithm (PDSA) and Earliest Deadline First (EDF). The PDSA algorithm is used in [12] is for the task that requires a single processing element for execution. Here we are further extending it and using it for the tasks requiring more than one processing element in heterogeneous environments. In addition to this, the proposed algorithm considers the processing speed of the resources. Metacentrum workload traces are used for the generation of the tasks. The experiment has been performed for varying workload by increasing number of tasks from 1000 to 4000 in a scalable manner.

The performance metrics for the scheduling algorithms is based on the average tardiness and the percentage of non-delayed tasks. In the deadline based system, our main emphasis is to make as much as tasks to be completed inside their deadline. So these two performance metrics give us the clear idea of the performance of the algorithms for these types of systems where the deadline of the task is the main constraint. The comparison of the proposed algorithm with other algorithms is described below using the performance metrics.

Table 1: Average Tardiness for Scheduling Algorithms (in Seconds)

No. of Tasks	EDF	PDSA	IPD
1000	783.22	783.22	106.59
2000	612.34	496.32	83.23
3000	751.35	458.37	110.52
4000	1275.73	1162.97	305.15

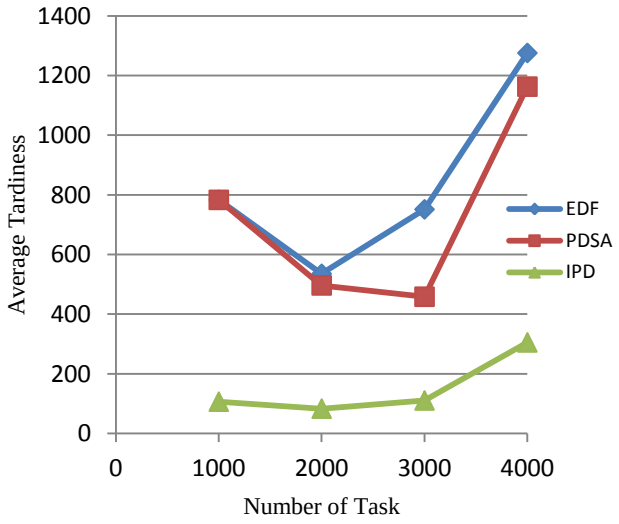

Fig 4: Average Tardiness

Table1. and graph in fig 4. shows the average tardiness of the scheduling algorithms. This graph shows the average tardiness of each scheduling algorithm under a variable number of tasks. It is clear from the graph that our purposed improvised scheduling algorithm gives better results than the other two scheduling algorithms. The tasks with large value of time delay are made to wait for the execution of the tasks with less time delay. Proposed algorithm works on scheduling the tasks with minimum time delay to the best resource with high processing speed. So the maximum tasks are completed inside the deadline given or near away from the deadline, thus reducing the average tardiness as compared to the others two scheduling algorithms. This is clearly seen in the graph given

above. The horizontal axis shows the number of the tasks, whereas the vertical axis shows the average tardiness. The average tardiness can be computed by using the Eq. (3).

Table 2. Percentage of non-delayed tasks for Scheduling Algorithms

NO. OF TASKS	EDF	PDSA	IPD
1000	95.3	95.3	95.85
2000	96.5	97.05	97.35
3000	96.9	97.56	97.86
4000	94.4	95.3	96.63

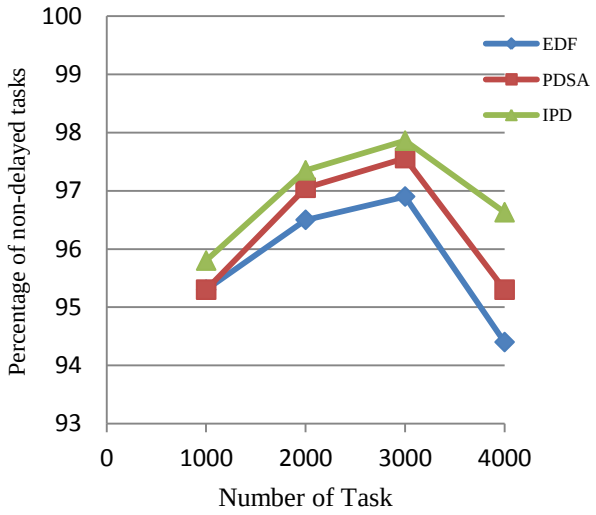

Fig 5: Percentage of non-delayed tasks

The percentage of the non-delayed tasks is shown by the Table 1. and graph in fig 5. The horizontal axis shows the number of the tasks, whereas the vertical axis shows the percentage of the non-delayed tasks. The analysis is done on a variable number of tasks. Again, it is clear from the graph that the purposed improvised algorithm has performed better than the other two scheduling algorithms that are PDSA and EDF. As the resource with the best processing speed will be given priority for executing the tasks, so more tasks will get executed inside their deadline thus decreasing the number of delayed tasks. For the deadline constraint tasks, the main emphasis is on the timely execution of the tasks inside their deadline. The proposed improvised algorithm makes as much as tasks possible to be getting executed inside their relevant deadline.

The overall comparative performance analysis has shown that our proposed algorithm is more efficient than simple PDSA and EDF. Average tardiness and the percentage of non-delayed tasks are the performance metrics. Results show that the average tardiness can be minimized and the number of non-delayed tasks can be increased with this approach in a heterogeneous grid environment.

5. CONCLUSION

In this paper, a scheduling strategy for high performance computing in a Grid Environment is proposed. The experiment has been conducted by varying the number of the tasks from 1000 to 4000. Only few algorithms are proposed for the grid environment where deadline of the task is of main emphasis. In this experiment we have considered the dynamic arrival of the tasks as well as the deadline requirement of each task to be processed. The proposed algorithm considers the expected completion time of the task for the scheduling purpose. This algorithm also considers the overall performance of the resources and their processing speed for deciding the assigning tasks. To decide which task will be submitted for execution, it selects the task with the minimum time delay. Then the task with the minimum time delay is assigned to the available resource with the best processing speed in the grid system. The results show that the proposed algorithm outperforms the PDSA and EDF scheduling algorithm. Two performance metrics average tardiness and non-delayed tasks has been used for the comparison purpose.

6. ACKNOWLEDGMENT

We thank the Czech National Grid Infrastructure Metacentrum for making the Metacentrum workload log publicly available.

7. REFERENCES

[1] I. Foster, C. Kesselman, S. Tuecke, “The Anatomy of the Grid: Enabling Scalable Virtual Organizations”, International J. Supercomputer Applications, 2001, 15(3).

[2] Yun-Han Lee, Seiven Leu, Ruay-Shiung Chang, “Improving task scheduling algorithms in a grid environment”, Future Generation Computer Systems , vol. 27, pp. 991-998, 2011.

[3] Braun T D, Siegel H J and Beck N. “A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems”. Journal of Parallel and Distributed Computing, Vol. 61, No. 1, pp. 810 –837, 2001.

[4] Menglan Hu and Bharadwaj Veeravalli, “Requirement-Aware Scheduling of Bag-of-Tasks Applications on Grids with Dynamic Resilience”, IEEE Transactions on Computer, vol. 62, no. 10, pp. 451-459, 2013.

[5] Shengwei YI,, Zhichao WANG, Shilong MA, Zhanbin CHE, Yonggang HUANG, Xin CHEN. “An Effective Algorithm of Tasks Scheduling in Clusters”, Journal of Computational Information Systems, vol 6, no. 10, pp. 3163-3171, 2010.

[6] Manoj Kumar Mishra, Raksha Sharma, Vishnu Kant Soni, Bivasa Ranjan Parida, Ranjan Kumar Das, “A Memory-Aware Dynamic Task Scheduling Model in Grid Computing”, International Conference On Computer Design And Applications, vol. 1, 2010, pp. 545-549.

[7] Jing Wang, Gongqing Wu, Bin Zhang, Xuegang Hu, “A heuristic algorithm for scheduling on grid computing environment”, Seventh ChinaGrid Annual Conference 2012, pp. 36-42.

[8] Sameer Singh Chauhan, R. C. Joshi, “A Weighted Mean Time Min-Min Max-MinSelective Scheduling Strategy for Independent Tasks on Grid”, 2nd IEEE on

International Advance Computing Conference , 2012, pp. 4-9.

[9] Siriluck Lorpunmanee, Mohd Noor Md Sap, Abdul Hanan Abdullah and Surat Srinoy" A static tasks scheduling for independent tasks in Grid Environment by using Fuzzy C-Mean and Genetic algorithms " Proceedings of the Postgraduate Annual Research Seminar 2006.

[10] Daphne Lopez, S. V. Kasmir Raja"A Dynamic Error Based Fair Scheduling Algorithm For A Computational Grid" Journal of Theoretical and Applied Information Technology © 2005 - 2009 JATIT.

[11] Korkhov, Vladimir V., Jakub T. Moscicki, and Valeria V. Krzhizhanovskaya. "Dynamic workload balancing of parallel applications with user-level scheduling on the Grid." Future Generation Computer Systems, vol 25, no. 1, pp- 28-34, 2009.

[12] Haruna Ahmed Abba, Nordin B. Zakaria, Syed Nasir Mehmood Shah, Anindya.J.Pal, "Deadline Based Performance Evaluation of Task Scheduling Algorithms" IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discover 2012, pp. 106-110.

[13] R. Buyya, M. Murshed, "GridSim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing," Journal of Concurrency and Computation: Practice and Experience, pp. 1175–1220, 2002.

[14] Dalibor Klusacek, Hana Rudova. Alea 2- Task Scheduling Simulator. In proceedings of the 3rd International ICST Conference on simulations and techniques (SIMUTools), ICST 2010.