International Journal of Scientific Research in Computer Science and Engineering **Research Paper**
Vol.6, Issue.2, pp.20-26 , April (2018) **E-ISSN:** 2320-7639

TRP Maximization Technique based Efficient Scheduling in Grid Environment

V. Indhumathi

Department of Computer science, Periyar University, Salem, India

Email:sabarishindhu@gmail.com

Available online at: www.isroset.org

Received: 10/Mar/2018, Revised: 22/Mar/2018, Accepted: 02/Apr/2018, Online: 30/Apr/ 2018

Abstract - The problem of heterogeneous job scheduling in grid environment is well studied. Number of approaches considers different factors of resource and process in terms of completion, waiting and throughput. However, the methods suffer to achieve higher performance in scheduling in grid environment. To improve the performance, an efficient TRP maximization algorithm is presented in this paper. First, the method identifies list of processes and their required resource with their hold time. Based on the information obtained, a set of two hop sequences are generated for each resource available. For each two hop sequence, the method compute throughput maximization support, resource utilization maximization factor and process completion maximization factor. Using all these, a scheduling maximization support has been estimated. Based on the support measure a single one has been selected for each resource available. The same will be iterated for each level of scheduling and the method produces higher efficiency in scheduling. The performance of scheduling has been improved with less waiting time.

Keywords: Grid Environment, Resource Utilization, Maximization, TRP, Job Scheduling

I. INTRODUCTION

The grid environment has been considered to contain number of resource of which has higher cost. Not all the organizations have the capability to purchase their own resource and the presence of grid provides a way to execute their jobs in a costly environment. The resources available in the grid could be used to execute their jobs. However, the presence of grid resource would vary depending on the resource requirement of the user jobs. It is not necessary that the number of resources required for a batch job to be equal to the number of resource available in the grid. So, it requires that the jobs have to be scheduled over the resources available.

There are number of scheduling algorithms available in literature. Some of the methods use the completion time or processing time of the jobs. Some other algorithms uses, resource utilization as the factor and some other uses other features. But whatever the algorithm, the efficiency of the algorithm is highly depending on how good the algorithm in maximizing different features of scheduling like resource utilization, throughput and process outcome. However, there exist number of algorithms for grid scheduling; they suffer to achieve higher performance in above mentioned features.

Resource utilization is the factor which represents the efficiency of using the resource. It has been measured based on the idle time of the resources and the number of jobs being used by the resources. However, the resource utilization has to be maximized to improve the performance of scheduling. The scheduling performance is highly depending on various factors like resource utilization, throughput performance and process execution. So all these parameters has to be maximized to improve the performance of scheduling.

In this paper, TRP maximization algorithm is presented. Throughput-Resource Utilization-Process Execution maximization algorithm has been presented in the sense to maximize all the factors considered. The throughput maximization can be achieved by reducing the overall completion time and the resource utilization can be maximized by reducing the idle time of resources. Similarly, the process execution efficiency can be maximized by reducing the waiting time of process or job. The detailed approach of scheduling has been discussed in the next section as section I contains the introduction of scheduling in grid, section II contain the related work of scheduling performed in various papers in Grid, Section III contains the proposed architecture , Data preparation , TRP Estimation and TRP maximization based scheduling, Section IV describes the results and discussion, section V concludes research work with future enhancements.

II. RELATED WORK

There are number of approaches has been discussed for the improvement of grid environment and this section discusses certain methods related to the heterogeneous scheduling problem.

A semi-static approach to mapping dynamic iterative tasks onto heterogeneous computing system [1], implement and evaluate a semi-static methodology involving the on-line use of off-line-derived mappings. The off-line phase is based on a genetic algorithm (GA) to generate high-quality mappings for a range of values for the dynamic parameters. A dynamic parameter space partitioning and sampling scheme is proposed that partitions the parameter space into a number of hyper-rectangles, within which the "best" mapping for each hyper-rectangle is stored in a mapping table. During the on-line phase, the actual dynamic parameters are observed and the off-line-derived mapping table is referenced to choose the most suitable mapping.

Scheduling of a meta-task with QoS requirements in heterogeneous computing systems [2], formulate the QoS-based scheduling problem by using utility and penalty functions, where a utility function associated with a task is used to measure how much the owner of this task will benefit from a given scheduling decision, while penalty functions associated with resources are used to provide incentives to users to set their QoS requirements in accordance with their needs. In order to solve the QoS-based scheduling problem, a computationally efficient static scheduling algorithm (QSMTS_IP) which assumes time-invariant penalty functions is developed. We later extend the QSMTS_IP to the case where penalty functions are time varying. Furthermore, it is shown that the QSMTS_IP can be modified to run as a dynamic scheduling algorithm.

Load Balancing and Bandwidth in Grid Computing Environments [3], Improved Enhanced Gridsim with Deadline Control (IEGDC) model is used to reduce bandwidth and for load balancing. It enhances the utilization of the resources and prevents the resource overloading. The selection method for the scheduling proposed here is called Fastest Bandwidth to Highest Capacity (FBHC). Our selection method considers the state of resource bandwidth and capacity of the resources. The incorporation of the said features of the proposed method makes it quite attractive in grid applications.

Improving job scheduling algorithms in a grid environment [4], propose a hierarchical framework and a job scheduling algorithm called Hierarchical Load Balanced Algorithm (HLBA) for Grid environment. In our algorithm, we use the system load as a parameter in determining a balance threshold. And the scheduler adapts the balance threshold dynamically when the system load changes. The main contributions of this paper are twofold. First, the scheduling algorithm balances the system load with an adaptive threshold and second, it minimizes the makespan of jobs

A hybrid scheduling algorithm with load balancing for computational grid [5], proposes a new scheduling algorithm for computational grids that considers load balancing, fault tolerance and user satisfaction based on the grid architecture, resource heterogeneity, resource availability and job characteristics such as user deadline. This algorithm reduces the makespan of the schedule along with user satisfaction and balanced load. A simulation is conducted using Grid Simulator Toolkit (GridSim).

Adaptive scoring job scheduling algorithm for grid computing [6], propose an Adaptive Scoring Job Scheduling algorithm (ASJS) for the grid environment. Compared to other methods, it can decrease the completion time of submitted jobs, which may compose of computing-intensive jobs and data-intensive jobs.

A non-cooperative game theory approach to optimize workflow scheduling in grid computing [7], provide suitable performance and response times, the available resources have to be scheduled and coordinated for workflow implementation in the grid environment. Therefore, task scheduling and resource allocation are very important to achieve high performance in grid computing. Game theory is one approach which can be used for scheduling. In this paper, a noncooperative game is proposed to minimize the time and cost of scheduling. Moreover, the aim of the proposed approach is to encourage resource brokers to use an optimal scheduling algorithm. In the proposed game, the broker profit is increased when a solution is proposed with lower time and cost for the users.

A bee colony task scheduling algorithm in computational grids [8], uses artificial bees to appropriately schedule the submitted tasks to the grid resources. Applying the proposed algorithm to the grid computing environments, the maximum delay and finish times of the tasks are reduced. Furthermore, the total makespan of the environment is minimized when the algorithm is applied. The proposed algorithm not only minimizes the makespan of the environment, but also satisfies the deadline and priority requirements of the tasks. Simulation results obtained from applying the algorithm to different grid environments show the prominence of the algorithm to other similar scheduling algorithms.

Immediate/Batch Mode Scheduling Algorithms for Grid Computing [9], reviews the literature concerning Minimum Execution Time (MET) along with Minimum Completion

Time (MCT) algorithms of online mode heuristics and more emphasis on Min-Min along with Max-Min algorithms of batch mode heuristics, while focusing on the details of their basic concepts, approaches, techniques, and open problems.
Max-min Average Algorithm for Scheduling Tasks in Grid Computing Systems. [10], proposed a new heuristic technique called Maxmin Average algorithm for task scheduling in the heterogeneous grid computing environment. Usually the performance of the grid computing is measured by the reduction in the idle time and makespan.

OPT-Min-Min Scheduling Algorithm of Grid Resources. [11], to overcome the drawbacks. Upon the basis of applying Min-Min pre scheduling into stage one, by adapting the strategy of two rounds scheduling, the assignments on heavy load resources are rescheduled to balance the load. Actual cases are used to illustrate the superiority of OPT-Min-Min scheduling algorithm to the Min-Min scheduling algorithm. Simulation experiments were carried out to the batch grid resource scheduling algorithms including Min-Min, Max-Min, and Min mean and OPT-Min-Min. According to the ETC generation technique used in experimental benchmarks, ETC matrix was generated.

In [12], Memory Constrained Load Shared Minimum Execution Time (MCLSMET) scheduling is proposed to make best use of the resource utilization in a grid environment to reduce makespan. Load balancing is achieved by rescheduling the resources based on memory requirement and execution time of the tasks. This algorithm considers memory as Quality of Service (QoS) factor. Results: The proposed algorithm has been implemented in a simulated environment and the results are compared with the Minimum Execution Time (MET) algorithms. In MCLSMET algorithm, the Maximum Completion Time, Resource Utilization is computed to compare with the existing MET scheduling Algorithm. The MET scheduling algorithm produces the makespan 34 ms whereas the proposed method reduces the makespan to 15 ms for a task. In the existing MET scheduling Algorithm produces severe load imbalance problem. In the proposed method load is shared among the available resource and the resource utilization percentage is increased.

TLLB: Two-Level Load Balanced Algorithm for Static Meta-Task Scheduling in Grid Computing [13], a new Two Level Load Balanced (TLLB) grid scheduler algorithm is proposed. In First Level min-min algorithm is used to create ITQ and in Second Level a new Transformation technique is used to reschedule.
Efficient Loads Balancing for Grid Computing System Using Min-Min Scheduling Algorithm [14], propose new scheduling algorithm based on well known task scheduling algorithms, Min-Min. The proposed algorithm tries to use the advantages of this basic algorithm and avoids its drawbacks. To achieve this, the proposed algorithm firstly like Min-Min estimating of the completion time of the tasks on each of resources and then selects the appropriate resource for scheduling. The experimental results show that the proposed algorithm improved total completion time of scheduling in compared to Min-Min algorithm.

Static Batch Mode Heuristic Algorithm for Mapping Independent Tasks in Computational Grid [15], proposes a new heuristic algorithm for mapping independent tasks in a grid environment to be assigned optimally among the available machines in a grid computing system. Due to the multi-objective nature of the grid scheduling problem, several performance measures and optimization criteria can be assumed to determine the quality of a given schedule. The metrics used here include makespan and resource utilization. This algorithm provides effective resource utilization by reducing machine idle time and minimizes makespan. This algorithm also balances load among the grid resources and produce high resource utilization with low computational complexity.

III. TRP MAXIMIZATION BASED SCHEDULING SCHEME

The TRP maximization algorithm identifies the list of processes and list of stages involved. For each stage of the processes, the resource required and the time of access has been identified from the job list. Then for each stage with the resources, the method computes list of possible sequences and for each of them, the method computes different maximization measures. Based on the maximization measures, a scheduling maximization support is estimated. Using the maximization support, a single sequence has been selected for each resource.

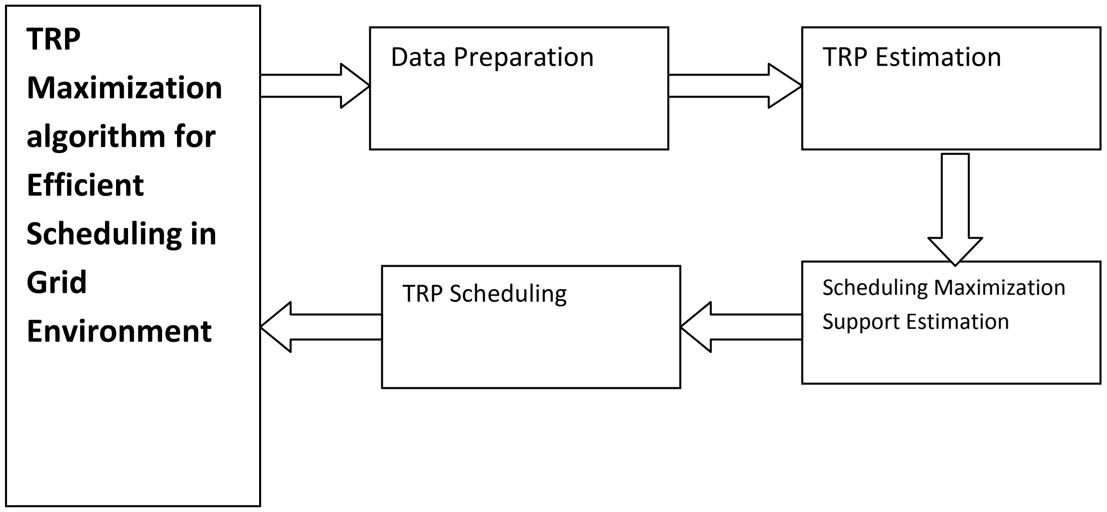
TRP Maximization algorithm for Efficient Scheduling in Grid Environment
Data Preparation
TRP Estimation
Scheduling Maximization Support Estimation
TRP Scheduling

Figure 1: Architecture of Proposed TRP based Scheduling

The Figure 1, shows the architecture of proposed TRP scheduling algorithm and shows various components of the proposed system.

A.Data Preparation

The job set given would have number of jobs and each job would have different resource requirements. The method identifies the tasks and their subsequent resources required have been identified. For each subsequent task, the method identifies the processing time and all has been converted into a feature vector. Generated feature vector has been used to perform scheduling.

Algorithm:

```
Input: Task Set Ts, Resource Set Rs
Output: Preprocessed Task set Pts
Start
    Read Task set Ts.
    Read Resource set Rs.
    For each task T
        Identify the list of subsequent task.
        Sts = $\sum Subsequenttask \in T$
        For each sub task st
            Identify processing time Prt = $st(Ti).ProcessingTime \in Ti$
            Identify the resource required ReR = $st(Ti).Resource \in Rs$
            Generate Feature Vector Fv =$\sum(fv \in Fv) \cup \{st, Prt, ReR\}$
        End
        Pts = Pts ∪ Fv
    End
Stop
```

The above discussed algorithm identifies the information about the processes and the resources required.

Table 1: Details of tasks.

Process	Resources required	Resource access Time	Completion Time
1	R1,, R2, R4	4,7,9	20
2	R1, R3, R5	2,5,9	16
3	R1,R2,R3,R4	3,6,8,9	26
4	R1,R3,R5	6,2,8	16
5	R1,R2,R3	4,7,9	20

The Table 1, shows the details of tasks being used to perform scheduling.

B.Pattern Generation

In this stage, for each resource available with the process, the possible sequences are generated. The number of instance of any resource would deflect from number of processes require the resource. When the number of resource instance required is higher than the number of resources available, the possible sequences can be generated to select a single one. Such possible sequences are generated in this stage which will be used to estimate different measures.

Algorithm:

```
Input: Preprocessed Task set Pts
Output: Pattern Set Ps.
Start
    Read task set Pts.
    Identify list of processes Pl = $\sum_{i=1}^{size(PTs)} Pk \in PTs(i) \nexists Pl$
    Identify list of resources required RRl = $\sum_{i=1}^{size(Pl)} Pl(i).Resource$
    For each resource Ri
        Compute required number Rn = $\sum_{i=1}^{size(RRl)} RRl(i) == Ri$
        If ResourceAvailable<Rn then
            Process set Prs = Identify list of process requires Ri.
        Generate sequences s = $\int_{i=1}^{Number\ of\ Process} PossibleSequences(Prs)$
        Add to pattern set Ps = $\sum(Sequence \in Ps) \cup S$
    End
Stop
```

The above discussed algorithm generates possible sequences in which the processes can be executed and the resources can be allocated. Based on the sequences generated, the method can estimate different measures to schedule the tasks.
According to Table 1, the following pattern has been generated for the resource R2.

Table 2: Sample pattern generated on resource R2

Hook Order Sequence
1-2-3-4
1-2-5-4
2-3-5-4

The Table 2 shows the sample sequence being generated for resource R2. Similarly you can generate 2^3 = 8 number of sequences.

D.TRP Estimation

In this stage, the method reads the sequences being generated by the pattern generation algorithm. Then for each sequence, the method compute the throughput maximization support, resource utilization maximization support and process execution maximization support. All these measure are estimated based on the idle time of resources, waiting time of processes; number of tasks gets cleared and so on.

Algorithm:

```
Input: Pattern p, Task Set Ts, Resource Set Rs, Process set Prs
Output: Tm, Rm, Pm
Start
    Read task set Ts.
    Read pattern p.
    For each resource R
        Compute idle resources IdR = $\sum_{i=1}^{size(Rs)} Rs(i) \nexists p$
    End
    Compute resource utilization maximization support Rm.
    Rm = $\frac{\sum(IdR) \times \sum ProcessingTime(p)}{No\ of\ Resource\ Types}$
    Identify no of idle jobs IdJ = $\sum_{i=1}^{size(prs)} PRs(i) \nexists p$
    Throughput maximization support Tm = $\frac{size(p)}{IdJ} \times makespan(p)$
    Process execution maximization Pm.
        Pm = $\frac{No\ of\ process\ complete\ with\ sequence\ p}{IdJ} \times \frac{makespan(p)}{idle\ Time\ (p)}$
Stop
```

The above discussed algorithm computes different maximization support measures to be used to perform scheduling.

C.TRP Maximization Based Scheduling

The algorithm reads the task set Ts, Resource Set Rs initially. Then the method prepares the data to identify the list of tasks and their resource constraint with time. For each resource at each round, the method generates the patterns. For each pattern generated, the method, computes the different maximization support values. Based on the support measures, a scheduling support has been estimated. The sequence which has higher scheduling support will be selected for the particular round trip.

```
Algorithm:
Input: Task set Ts, Resource Set Rs
Output: Null
Start
    Read Ts, Rs.
    Pts = data Preparation (Ts,Rs)
    For each round Rn
        For each resource R
            Pattern set Ps = GeneratePattern(pts)
            For each pattern p
                [Tm,Rm,Pm] = TRP-Estimation ()
            End
            Compute Scheduling Support Sms.
```

```
                Sms = $\frac{Tm}{Rm} \times \frac{Pm}{Tm}$
                Choose the pattern with maximum scheduling support.
                P = Max(P(Sms))
            End
        End
Stop
```

The above discussed algorithm computes the scheduling maximization support for each sequence. Using the value of scheduling support a single sequence has been selected. Finally a single sequence has been selected for scheduling.

IV. RESULTS AND DISCUSSION

The proposed TRP maximization scheduling algorithm has been implemented and evaluated for its efficiency. The method has been evaluated for its efficiency under various simulation conditions. The method has been implemented using advanced java. The method has produced the following results.

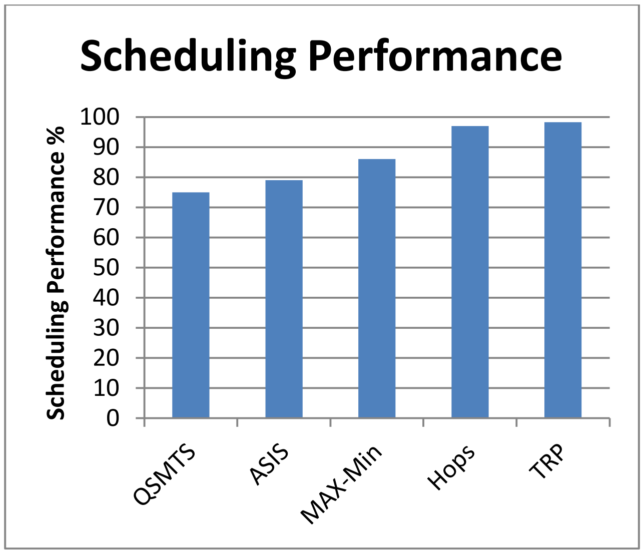

Figure 2: Comparison on scheduling performance

The Figure 2 shows the comparison result on scheduling performance produced by various methods. The proposed TRP maximization approach has produced higher scheduling performance than other methods.

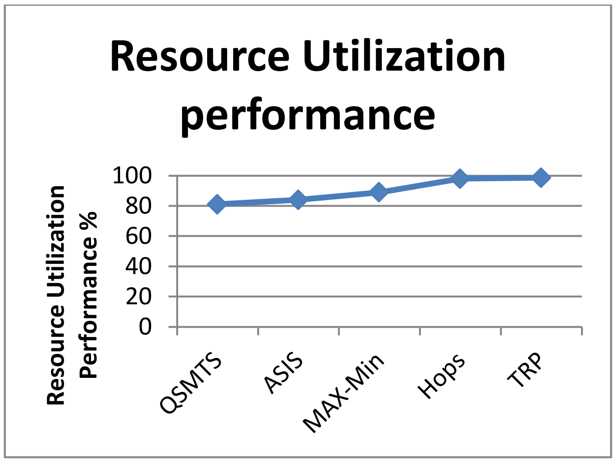

Figure 3: Comparison on resource utilization

The Figure 3 shows the comparative result on resource utilization produced by various methods. The result shows clearly that the proposed Hops approach improves the performance of resource utilization than other methods.

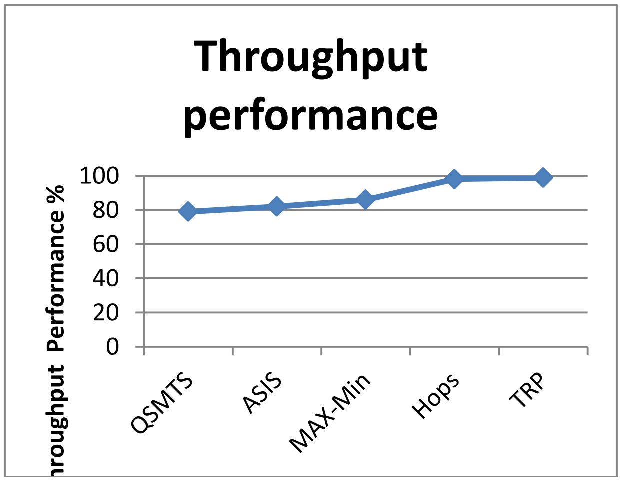

Figure 4: Comparison on throughput performance

The Figure 4 shows the comparison result on throughput performance produced by various methods. The proposed TRP maximization scheduling algorithm has produced higher throughput than other methods.

V. CONCLUSION

In this paper, a real time TRP (Throughput-Resource utilization-process execution maximization) scheduling algorithm is presented. The method preprocesses the task set to identify the list of task and their sub task available. Then the method identifies the list of resources required and their execution time is identified. Using the preprocessed data, the method generates fist level patterns which are possible in the combinatory. Then for each level, the method generates the different possible pattern and add to the pattern set. Third, using the pattern generated, the method computes the throughput maximization support, resource utilization maximization support and process execution maximization support measures. Using these measures a scheduling support has been computed to select a most weighted sequence for scheduling. The method produces higher efficiency in scheduling and increases the resource utilization factor as well.

REFERENCES

[1] Kwok, Y.K., Maciejewski, A.A., Siegel, H.J., Ahmad, I., Ghafoor, A.: A semi-static approach to mapping dynamic iterative tasks onto heterogeneous computing system. Journal of Parallel and Distributed Computing 66, 77–98 (2006)

[2] Dogana, A., Özgüner, F.: Scheduling of a meta-task with QoS requirements in heterogeneous computing systems. Journal of Parallel and Distributed Computing 66(2), 181–196 (2006)

[3] J. Preethi1 , R. Jayasudha2,Load Balancing and Bandwidth in Grid Computing Environments , International Journal of scientific research in computer science, engineering and technology, vol.2, issue 2, 2017.

[4] Y. H. Lee, S. Leu, R. S. Chang, "Improving job scheduling algorithms in a grid environment", Future generation computer systems, vol. 27, no. 8, pp. 991-998, 2011.

[5] P. Keerthika, N. Kasthuri, "A hybrid scheduling algorithm with load balancing for computational grid", *International Journal of Advanced Science and Technology*, vol. 58, pp. 13-28, 2013.

[6] Chang Ruay-Shiung, Lin Chih-Yuan, Lin Chun-Fu, "An adaptive scoring job scheduling algorithm for grid computing", Information Sciences, vol. 207, pp. 79-89, 2012.

[7] M. Yaghoobi, A. Fanian, H. Khajemohammadi, T. A. Gulliver, "A non-cooperative game theory approach to optimize workflow scheduling in grid computing", *Communication Computers and Signal Processing (PACRIM)*, 2013.

[8] Z. Mousavinasab, R. Entezari-Maleki, A. Movaghar, "A bee colony task scheduling algorithm in computational grids" in Digital Information Processing and Communications, Springer Berlin Heidelberg, pp. 200-210, 2011.

[9] J.Y Maipan-Uku1 , I. Rabiu2 , Dr. Amit Mishra3, Immediate/Batch Mode Scheduling Algorithms For Grid Computing: A Review,International Journal Of Research, 2017.

[10] Amalarethinam, G.D.I. & Kfatheen V.S., (2014). Max-min Average Algorithm for Scheduling Tasks in Grid Computing Systems. International Journal of Computer Science and Information Technologies. 3, pp. 3659-62.

[11] Cao, L., Liu, X., Wang, H., & Zhang, Z. (2014). OPT-Min-Min Scheduling Algorithm of Grid Resources. Journal of Software, 9 (7), pp. 1868-1875.

[12] Hemamalini, M., & Srinath, M. V. (2015). Memory Constrained Load Shared Minimum Execution Time Grid Task Scheduling Algorithm in a Heterogeneous Environment. Indian Journal of Science and Technology, 8 (15)

[13] Kfatheen, S.V., Banu, M.N., & Selvi, S.K., (2014). TLLB: Two-Level Load Balanced Algorithm for Static Meta-Task Scheduling in Grid Computing. International Journal of Computer Applications (0975 – 8887). 105 (38 - 43), pp. 38 – 41.

[15] Maipan-uku, J.Y, Muhammed, A., Abdullah, A., Hussin, M. (2016). Efficient Loads Balancing for Grid Computing System Using Min-Min Scheduling Algorithm. International Journal of Applied Engineering Research (IJAER).