TAAG: An Efficient Task Allocation Algorithm for Grid Environment

S. Vaaheedha Kfatheen[#1], Dr. M. Nazreen Banu[*2]

[#]Research Scholar, Bharathiar University,
Coimbatore, Tamil Nadu, India.
E-Mail: abuthaheer67@gmail.com

[*]Professor, Dept of MCA, M.A.M. College of Engg,
Tiruchirappalli, Tamil Nadu, India.
E-Mail: nazreentech@mamce.org

Abstract— Grid computing is a form of distributed computing where the resources of various computers are shared to solve a particular problem. Due to heterogeneity of resources, scheduling a task is significantly complex in nature. Scheduling strategy plays a vital role in the grid environment to schedule the user tasks and dispatch them to the appropriate grid resources. A good task scheduling method is the one which reduces the total time taken for execution of a given task in the grid. In this paper, we propose a new scheduling algorithm called TAAG (An Efficient Task Allocation Algorithm for Grid) for efficient allocation of tasks on resources in the Grid environment. The proposed algorithm tries to use the advantages of the recent popular task scheduling algorithm known as "Improved Min-Min Task Scheduling Algorithm" [2] and covers its disadvantages by using a new resource allocation model. In this paper, the loads of the task are divided and are allocated into resources with respect to the computing capacity of the resource. Experimental results show that the proposed algorithm obtain better solution in terms of completion time, cost, makespan and load balancing compared to the existing algorithms.

Keyword- Grid scheduling, Resource management, Workload distribution, Task selection, Resource allocation.

I. INTRODUCTION

Computational Grids are becoming more prevalent, due to the sharp decrease in the hardware and integration costs. Scientific applications in general, require higher computation capabilities. But affordability of a single system becomes very high. Due to the low cost nature of the grids, and their obvious advantages over super computers, grids are mostly preferred for usage in scientific applications. Further, grids can provide better performance than larger parallel super computers. But for providing higher processor efficiencies, efficient scheduling policies are required. The performance of a grid largely depends on optimal scheduling policies used in it. A proper scheduling algorithm should consider most or all of the QoS properties, viz. waiting time, makespan, load balance, flow time, turn-around time, response time, total completion time, bounded slowdown time, stretch time, fairness etc. The more alluring advantages of grids and the highly available nature of the grids have persuaded potential researchers to work in the area.

Scheduling refers to the process of providing the processor time to threads, process or data flows. This is usually done to load balance and to share system resources effectively or to achieve a target quality of service. The need for a scheduling algorithm arises from the requirement for most modern systems to perform multiplexing. Scheduling in a grid is not considered as a single problem, instead a family of problems, with many constraints, and each with different properties which are to be considered while designing a scheduling algorithm for a grid. It is observed that Grid Scheduling is a Non-deterministic Polynomial time (NP-Complete) problem [1].

The "Improved Min-Min Task Scheduling Algorithm" in Grid Computing [2] estimates the execution time of each task on different resources. With estimated result the suitable algorithm is used to schedule the task on the selected resource. Thus the execution of the task is delayed during the estimation process done on each resource.

This paper proposes a new algorithm, named TAAG (An Efficient Task Allocation Algorithm in Grid) to resolve the resource allocation problem by executing a single task on a number of resources by dividing the load of the task with respect to the computing capacity of the resource, by avoiding communication delay to submit a portion of the load to the resources.

The proposed algorithm TAAG has two stages - Stage 1 (Task Selection): In this stage, the list of all available tasks is sorted in ascending order. Then the tasks are selected from front or rear of the list by comparing the values of Average Workload (AW) and Standard Deviation (SD) of the tasks workload. Stage 2 (Resource allocation): In this stage, TAAG divides the load of the selected task with respect to the computing

capacity of the resource. TAAG reduces the minimum completion time of all the tasks on available resources compared to the existing Improved Min-Min Task Scheduling Algorithm. Thus the proposed algorithm TAAG shows a better makespan than the Improved Min-Min Task Scheduling Algorithm in Grid Computing by dividing the loads of the task and also optimizes the load by reordering them according to the capacity of the resource.

The remainder of this paper is structured as follows: Section 2 presents the related works and several well known scheduling algorithms which are used as benchmarks of many other works. In Section 3, Improved Min-Min task scheduling algorithm is presented. In Section 4, a new task scheduling algorithm TAAG is proposed. Section 5 describes the design of resource allocation and showing the experimental results for the proposed model. Finally, Section 6 presents concluding remarks.

II. RELATED WORK

Several grid scheduling algorithms have been proposed to optimize the overall grid system performance. The study of managing resources in the Grid environment started in 1960s. The economic problem [3] results from having different ways for using the available resources; Utilization of the Grid must be cheaper [4] and should also satisfy the requirements of the Grid users. On the other hand, resource providers must know whether it is worth to provide their resources for usage to the Grid users.

Pricing of resources per time unit or slot (eg. cost per minutes) [5] affects the users with slow processing resources. Moreover the users have the knowledge of time units needed to complete their tasks. Therefore, the cost is to be determined based on the task and the resource in which it is processed.

Buyya R et al., [6][7][8] proposed four scheduling algorithms. The Cost scheduling algorithm tries to reduce the amount of money paid for executing the tasks with respect to the deadline. The Time scheduling algorithm attempts to minimize the time required to complete the tasks with respect to their budget allotment. The conservative time scheduling algorithm aims to execute the tasks within the stipulated budget and the deadlines. The cost-time scheduling algorithm works as cost scheduling algorithm except that when there are two or more resources with the same price, it employs time scheduling algorithm.

Buyya R et al., [9] architecture are responsible for managing grid resources, and mapping tasks to suitable resources according to the utility functions used. Parallel virtual machine [10][11] enables the computational resources to be used as if they are a single high performance machine. It supports both execution on a single and multiple resources by splitting the task into subtasks.

Murugesan G et al.,[12][13] proposed a mathematical model with equal portion of load to all the processors. The entire workload received from a source is equally divided with the help of random numbers and a portion of the load is assigned to a processor.

Abuthahir J et al., have proposed an algorithm called RAA (Resource Allocation Algorithm) [14] for efficient grid scheduling workflows involving considerable communication overheads. They have developed an effective iterative model for optimal workload allocation. The model is proposed for load allocation to nodes and links for scheduling divisible workload applications.

K. Etminani et al. have proposed a new algorithm which uses Max-min and Min-min algorithms [15]. The algorithm selects one of these two algorithms, depending on standard deviation value of the expected completion time of the tasks on each of the resources.

S. Parsa et al. have proposed an algorithm called RASA [16]. RASA begins with Min-Min algorithm if the number of available resources is odd and Max-Min algorithm if the number of available resources is even. The remaining tasks are assigned to their appropriate resources by one of the two strategies, alternatively.

III. AN IMPROVED MIN-MIN TASK SCHEDULING ALGORITHM [2]

Firstly it computes the amount of task completion time CT_{ij} for all tasks in MT on all resources from the equation 1:

$$CT_{ij} = ET_{ij} + \mathrm{r_j} \quad \mathrm{Eq}(1)$$

CT_{ij} is completion time and ET_{ij} is expected execution time of task i_{th} on resource j_{th}, and r_j is the ready time for resource j_{th} (r_j is the ready time or availability time of resource j after completing previously assigned tasks). After that, the set of minimum expected completion time for each task in MT is found (resource discovery), then the task with the overall minimum expected completion time from MT is selected and assigned to the corresponding resource (resource selection).

Then all the tasks should be sorted ascending. It means tasks with minimum completion time are in the front of queue and tasks with maximum completion time are in the rear of the queue. Next this algorithm like the Min-Min algorithm, computes minimum completion time of all tasks on available resources. After that, the resource according to the appropriate condition should be chosen. For choosing a task for scheduling, firstly computes average of completion time (ACT), and standard deviation (SD) of existing tasks. After, the proposed

algorithm compares values of ACT and SD. Based on the comparison, there may be two cases: 1. If ACT is less than SD (i.e. the workload of all tasks in MT is in a small range) assign next task to the resource from the front of the queue. 2. Otherwise from the rear of the queue. Results show that the algorithm performs better makespan and optimizes load balancing compared to the Min-Min algorithm.

IV. TAAG (AN EFFICIENT TASK ALLOCATION ALGORITHM FOR GRID)

In TAAG, the task is executed on a number of resources instead of a single resource by dividing its loads with respect to the computing capacity of the resource. There is no communication delay to submit a portion of load to a resource.

The new algorithm is based on the following assumptions:

- Tasks have no deadlines or priorities associated with them.
- The mapping process is to be performed statically in a batch mode fashion.
- The size of the meta-tasks and the number of resources in the heterogeneous computing environment is known priori.

A. Two Stages:

1) *Stage 1 (Task Selection):* Firstly all the tasks are to be sorted in ascending order based on the minimum completion time. It means that the tasks with minimum workloads are in the front of queue and tasks with maximum workloads are in the rear of queue. To schedule a task, compute the average workload and standard deviation of existing tasks. According to [17], the average of workload (AW) and standard deviation (SD) of tasks are calculated using the equation 2 and 3:

$$\mathbf{AW} = \frac{\sum_{i=1}^{n} \boldsymbol{Workload}}{\boldsymbol{n}} \qquad \text{Eq (2)}$$

$$\mathbf{SD} = \sqrt{\frac{\sum_{i=1}^{n} (\boldsymbol{Worload - AW})^2}{\boldsymbol{n}}} \qquad \text{Eq(3)}$$

Where n - Number of tasks and AW-Average Workload.

The proposed algorithm now compares the values of AW and SD. Based on it, two cases may occur:

- If AW is less than SD (i.e. the workload of all tasks in MT is in a small range), assign the next task from the front of the queue.
- Otherwise, assign it from the rear of queue.

Tasks	T1	T2	T3	T4
Workload	3	7	12	47

(a) AW = 17.25 , SD =17.47
AW < SD (Select task from rear of queue)

(a)

Tasks	T1	T2	T3
Workload	3	7	12

(b) AW= 7.33 , SD = 3.68
AW> SD (Select task from front of queue)

(b)

Tasks	T2	T3
Workload	7	12

(c) AW = 9.5 , SD = 2.5
AW > SD (Select task from front of queue)

(c)

Tasks	T3
Workload	12

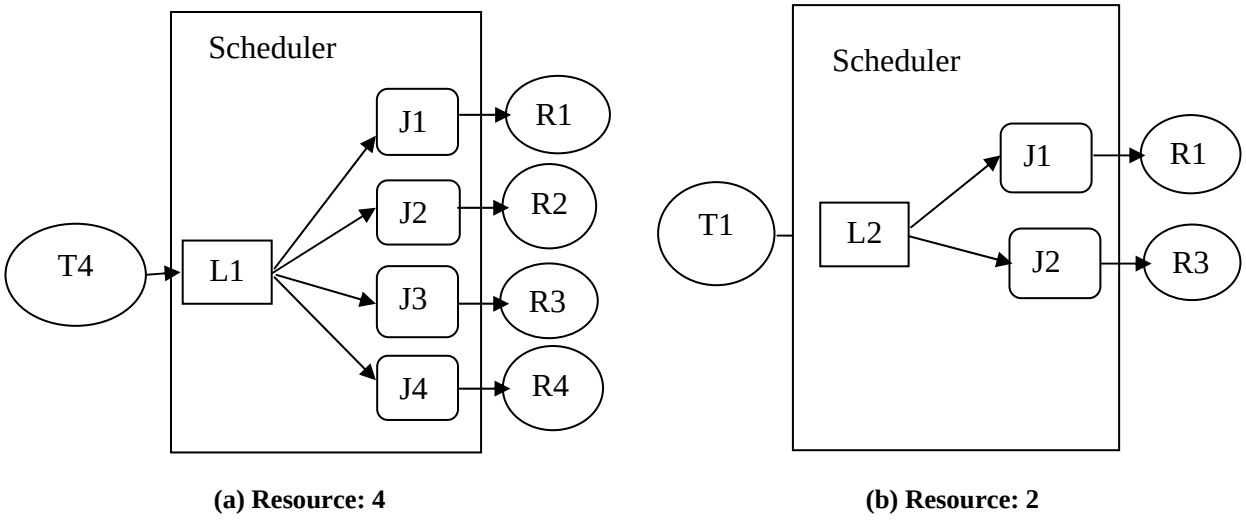
(d) (Select the last task of queue)

(d)

Fig. 1. Task Selection in TAAG

Fig. 1 shows that how proposed algorithm selects tasks for scheduling, according to the values of Average Workload and Standard Deviation.

2) *Stage 2 (Resource Allocation):* The loads received from the selected task become divisible based on the resource capacity. Let us assume that if there are m number of tasks {T1, T2, . . . , Tm }and n number of resources {R1, R2, . . . , Rn} the workloads from each tasks become L = {L1, L2, . . . ,Lm} where L1 be the total workload received from the task T1 and so on. Each workload Li is divided into J1, J2 . . . .Jn. The divided load Li is reordered by the scheduler with respect to the capacity of the resource to achieve good performance of computation.



(a) Resource: 4 **(b) Resource: 2**

Fig. 2: Load allocated among resources.

In Fig 2(a) the workload of T4 is divided into four parts and in fig 2(b) the workload of T1 is divided into two parts and allotted into resources. Similarly the remaining tasks are allotted into resources.

1) Initialize Parameters MT,S,t,L
(2) Sort all tasks in MT ascending order
(3) While there are tasks in MT
(4) For all tasks t_i in MT
(5) Calculate average workload & standard deviation of all tasks in MT.
(6) If AW> SD then // AW = average workload, SD = standard deviation
(7) tf = tfront
(8) Call Allocate [tf]
(9) Else
(10) tr = trear
(11) Call Allocate [tr]
(12) End if
(13) Delete assigned task from MT.
(14) End While
Allocate [t]:

a) Insert t details to S
b) Unequally Divide the Li of t into J1, J2…..
c) Reordering of Li with respect to the resource capacity // Load balancing.
d) Update node details (start time & end time) to S.

Fig. 3. The pseudo code of TAAG

V. EXPERIMENTAL RESULTS

Interactive software is developed in C++ to execute both algorithms. The Grid system consists of four resources, namely, R1, R2, R3 and R4 with four tasks T1, T2, T3 and T4 which are trying to utilize the grid system to execute their workloads.

TABLE 1
Available Resource Capacity

Resource	Processing time / Unit Workload (min)	Resource Cost(Rs)/unit Processing Workload
R1	**2**	**1**
R2	**3**	**2**
R3	**6**	**5**
R4	**4**	**3**

Table 1 shows the set of available resources, the time required by each of them to process one unit of workload and the associated cost.

TABLE 2
Details of Workloads of Tasks

Tasks	Work Loads (MB)
T1	**3**
T2	**7**
T3	**12**
T4	**47**

Table 2 shows the details of total workload of each task in MB.

TABLE 3
Allocation of Tasks

Tasks	Resources Allotted	Allotted Workloads	Time taken to complete the tasks by the Resources	Cost to complete the task by Resources	Overall Time taken to Complete the tasks	Overall cost to Complete the tasks
T4	R1	30	60	30	115	68
	R2	15	45	30		
	R3	1	6	5		
	R4	1	4	3		
T1	R1	1	2	1	14	11
	R3	2	12	10		
T2	R1	2	4	2	24	17
	R2	2	6	4		
	R3	1	6	5		
	R4	2	8	6		
T3	R1	6	12	6	33	21
	R2	3	9	6		
	R4	3	12	9		

Table 3 shows the details of workload allotment to the resources involved in the process. The workload of T4 is divided into four parts, T1 into two parts, T2 into four parts and T3 into three parts and allocated into the allotted resources. Also, it shows the time and cost taken to complete the tasks by the individual resource and Overall Time and cost taken to complete the tasks by the resources.

A. **Completion time of Tasks:** The workload of T4 (47 MB) is divided as 30 MB to Resource 1 (R1), 15 MB to Resource 2 (R2), 1 MB to Resource 3 (R3) and 1 MB to Resource 4 (R4). R1 Processing time/Unit Workload is 2. Therefore the time taken to complete the work by R1 is 30 x 2 = 60. Similarly the time taken to complete the task by R2, R3 and R4 is 45, 6 and 4 respectively. The overall time taken to complete task 4 is the sum of the Time taken by resources R1, R2, R3 and R4 i.e., 60+45+6+4 = 115.Similarly for Task 2 and Task 3, the Completion time are calculated.

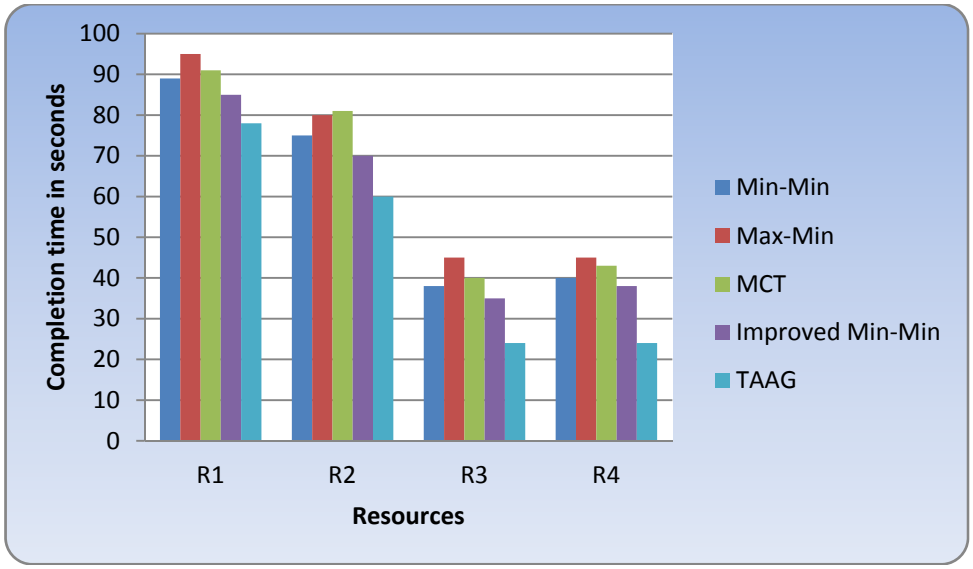


Fig. 4. Comparison based on Completion time

Fig.4 gives the comparison charts of traditional heuristics and TAAG. From this, we observe that the completion obtained by TAAG is less compared to other heuristic algorithms Min-Min, Max-Min, MCT and Improved Min-Min. Also the proposed algorithm uses all of the resources concurrently which help load balancing.

B. **Cost for Tasks completion:** The Cost per unit Workload of R1 is 1. According to Table 3 T4 30 MB workloads are allotted to R1. Therefore the total workload cost is 1* 30 =30. Similarly the total workload cost occurred to complete the T4 by R2, R3 and R4 is 30, 5 and 3 respectively. The overall cost occurred to complete the T4 is the sum of the total workload cost occurred by Resources R1, R2, R3 and R4 (30+30+5+3 = 68). Similarly Task 2 and Task 3 costs are calculated.

C. **Makespan:** The most popular optimization criterion in scheduling algorithm is minimization of makespan. Makespan is used to measure the throughput of grid system. It can be defined as

Cmax= max {Cj, j=1,……,n}

Where C – Completion Time of resources.

TABLE 4
Comparison of heuristics – Makespan in Seconds

Problem set	Min-Min	Max-Min	MCT	Improved Min-Min	TAAG
P1	12	15	13	10	8
P2	28	34	31	25	20
P3	41	45	43	38	32
P4	53	57	55	49	40
P5	54	60	57	50	45

Figure 4 shows the comparison of traditional heuristics and TAAG. It shows that the makespan obtained by TAAG is less compared to other heuristic algorithms Min-Min, Max-Min, MCT and Improved Min-Min.

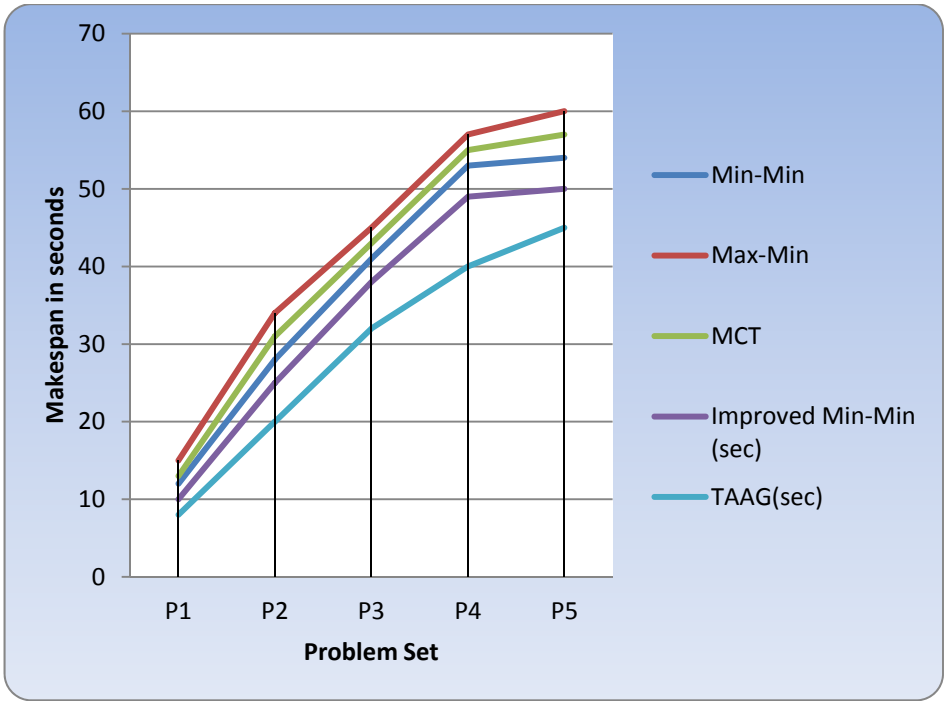


Fig. 5. Comparison based on Makespan

VI. Conclusion and Future Work

To overcome the limitations of the Improved Min-Min Task Scheduling Algorithm, in this paper a new Task Allocation Algorithm is proposed. It uses the advantages of the Improved Min-Min Task Scheduling Algorithm and covers their disadvantages. This algorithm proposed a new condition for selection of the task for scheduling, balances the load by dividing the loads of the tasks with respect to resource capacity. Experimental results show that the proposed algorithm outperforms better makespan than existing traditional heuristics algorithms and also helps load balancing. The study can be further extended by applying the concept in dynamic grid environment and other issues like flow time, execution time, deadlines on tasks and resources can also be considered.

References

[1] He, X., Sun, X.-H., Laszewski, G.V.: QoS Guided Min-min Heuristic for Grid Task Scheduling. Journal of Computer Science and Technology 18, 442–451 ,2003.
[2] Soheil Anousha and Mahmoud Ahmadi: An Improved Min-Min Task Scheduling Algorithm in Grid Computing, J.J. Park et al. (Eds.): GPC 2013, LNCS 7861, pp. 103–113, 2013. © Springer-Verlag Berlin Heidelberg 2013.
[3] Nakai J. Pricing computing resources: Reading between the lines and beyond. Technical report, National Aeronautics and Space Administration, 2002.
[4] He L, Sun X, and Laszewski G V. A qos guided scheduling algorithm for grid scheduling, 2003.
[5] He L Designing economic-based distributed resource and task allocation mechanism for self-interested agents in computational grids. Proceeding of GRACEHOPPER,2004.
[6] Buyya R, Economic-based distributed resource management and scheduling for grid computing. PhD thesis, Monash University, Melbourne, Australia, 2002.
[7] Buyya R, Abramson D, and Giddy J. Nimrod/g: An architecture for a resource management and scheduling system in a global computational grid, HPC ASIA'2000, 2000.
[8] Garg S K, Buyya R and Siegel H J. Scheduling Parallel Applications on Utility Grids: Time and Cost Trade-off Management, Proceedings of the Thirty-Second Australasian Computer Science Conference (ACSC2009), Wellington, Australia, Conferences in Research and Practice in Information Technology(CRPIT), Vol. 91.
[9] Buyya.R and Vazhkudai.S. Computer power market: Towards a market oriented grid. CCGRID, pg.574-581, 2001.
[10] Beguelin A, Dongarra J, Geist A, and Sunderam V. Visualization and debugging in a heterogeneous environment. Computer, 26(6):88-95, 1993.
[11] Sunderam V. PVM: A framework for parallel distributed computing. Concurrency, Practice and Experience, 2(4):pg. 315-340, 1990.
[12] Murugesan G and Chellappan C, An Optimal allocation of loads from multiple sources for Grid Scheduling, ICETIC'2009, International Conference on Emerging Trends in Computing, 2009.
[13] Murugesan G and Chellappan C, An Economic Allocation of Resources for Multiple Grid Applications, Proceedings of the World Congress on Engineering and Computer Science, Vol I, WCECS 2009.
[14] J.Abuthahir, S. Vaaheedha Kfatheen and Dr. M. Nazreen Banu, RAA: Task Scheduling Algorithm in Grid Computing Environment, CIIT International Journal of Networking and Communication Engineering, ISSN: 0974-9713 Volume 6, No 01, 2014.
[15] Etminani, K.Naghibzadeh, M.: A Min-min Max-min Selective Algorithm for Grid Task Scheduling. In: The Third IEEE/IFIP International Conference on Internet, Uzbekistan, 2007.
[16] Parsa, S., Entezari-Maleki, R.: RASA: A New Grid Task Scheduling Algorithm.International Journal of Digital Content Technology and its Applications 3(4), 2009.
[17] Cao, J., Spooner, D.P., Jarvis, S.A., Nudd, G.R.: Grid Load Balancing Using Intelligent Agents. Future Generation Computer Systems 21(1), 135–149, 2005.

AUTHOR PROFILE

S. Vaaheedha Kfatheen received her B.Sc Computer Science degree in 1993 and MCA degree in 2001, both from Bharathidasan University,Tiruchirappalli, Tamil Nadu, India. Presently pursuring Ph.D. in the area of Grid Computing in Bharathiyar University, Coimbatore, Tamilnadu, India. Presently she is an Assistant professor in the Department of Computer Science, Jamal Mohamed College, Tiriuchirappalli, Tamilnadu, India.

M.Nazreen Banu received her B.Sc chemistry Degree in 1994 and MCA Degree in 1997 , both from Bharathidasan University, Tiruchirappalli, Tamil Nadu, India . She gained her Ph.D. degree from Nagoya Institute of Technology, Nagoya, Japan. At present, she is working as a professor in Department of Computer Science & Engineering M.A.M College of Engineering, Tiruchirappalli, Tamil Nadu, India. Her research interests include Distributed & Parallel processing, operating systems and Advanced computer networks. She was a recipient of Japanese (MEXT) scholarship, from 2008 – 2012. She worked as an assistant professor in Jamal Mohamed College, Tiruchirappalli from 1999 – 2008.