International Journal of Applied Engineering Research ISSN 0973-4562 Volume 14, Number 1 (2019) pp. 92-96

Task Group Scheduling Algorithm for Mapping a Set of Independent Tasks in Each Group Based on QoS onto Heterogeneous Resources in a Distributed Grid Environment

Dr. G.K.Kamalam[1] and B.Anitha[2]

[1]*M.E.,Ph.D, Department of Information Technology, Kongu Engineering College, Perundurai, Erode, Tamil Nadu 638 060, India.*

[2]*M.E., Department of Information Technology, Kongu Engineering College, Perundurai, Erode, Tamil Nadu 638 060, India.*

Abstract

This paper proposes the Task Group Scheduling Algorithm (TGSA) that is used to schedule tasks in the distributed grid environment by applying quality of service to satisfy the user's needs. The proposed TGSA algorithm distributes tasks into four groups. Each group has tasks with similar attributes (user type, task type, task length, and task priority). After adding tasks into right group, it starts scheduling these tasks into available resources. Scheduling is done in two steps: first step is deciding which group will be scheduled first. This depends on the attributes of the tasks that belong to each group so the group that has tasks with high value of task attributes and high priority will be scheduled first. Second step is deciding which task inside the chosen group will be scheduled first. This depends on the completion time of task so the task that has the minimum completion time will be scheduled first.

Keywords: Task Scheduling, Quality of Service, NP-Complete, makespan.

INTRODUCTION

Grid computing enables large-scale computing resource sharing and collaboration for solving advanced science and engineering applications that are computationally complex [1]. Coordination of geographically distributed heterogeneous computing resources creates virtual organizations that support the utilization of idle resources [2]. The technology that aggregates distributed computer resources across the world have led to the foundation of new paradigm known as Grid Computing [3,4]. Grid Computing is a type of parallel and distributed system that involves the integrated and collaborative use of large-scale computing resources depending on their availability and capability to satisfy the demands of researchers to execute advanced science and engineering applications that requires large amount of communication and computation power. Central to the grid computing is the scheduling of application tasks to the resources. Scheduling independent tasks on it is more complicated. General task scheduling is an NP-Complete problem [5]. Schedule task in a grid environment efficiently is a new challenge because grid is a distributed and heterogeneous system. The requirement in grid computing environment is scheduling the current tasks to be executed with the given constraints. The meaning of constrains here is applying QoS that users need and balancing between these QoS and fairness among the execution of the tasks. The quality of service (QoS) demands of the user is like minimizing the overall execution time and executing the tasks that have highest priority. This paper proposes a new scheduling algorithm to decrease job's completion time in a grid environment. The proposed algorithm calculates priority to each task depending on the task attributes and grouped the tasks based on the priority and then schedules the group that have high priority. However the proposed TGS Algorithm uses the procedure of Min-Min algorithm to schedule tasks inside each group.

RELATED WORK

Min-min algorithm is the best heuristic scheduling algorithm for scheduling the tasks to the resources. Min-min algorithm schedules the tasks in two phases, in first phase, the minimum expected completion time for each task in all resources is calculated. In second phase, the task with overall minimum expected completion time is selected and scheduled to the corresponding resource. The algorithm does not consider the idleness of the resource [6,7,8].

The previous work [9,10,11] Min-mean heuristic scheduling algorithm works in two phases. In the first phase, Min-mean heuristic scheduling algorithm starts with a set of all unmapped tasks. The algorithm calculates the completion time for each task on each resource and finds the minimum completion time for each task. From that group, the algorithm selects the task with the overall minimum completion time and allocates to the appropriate resource. Removes the task from the task set. This process repeats until all the tasks get mapped. The algorithm calculates the total completion time of all the resources and the mean completion time. In the second phase, the algorithm selects the resource whose completion time is greater than mean completion time. The algorithm arranges the selected resources in the decreasing order of the completion time. The algorithm reschedules the tasks assigned on the selected resources to the resource, whose completion time is less than the mean completion time. The previous work [12] Credit Score Tasks Scheduling Algorithm is to identify the appropriate resource and to find the order in which the set of tasks to be mapped to the selected resource. The order in which the tasks to be mapped is identified based on the Credit Score of the task. The Credit Score of the task is calculated based on the execution time of the task. The previous work [14] Task Unique Credit System Scheduling Algorithm, provides the order in which the tasks are to be scheduled is determined based on the highest credit value of the task. The highest credit value of the task is calculated based on the length of the task and unique identification value of the task. The previous work [13] QoS

Guided Prominent Value Tasks Scheduling Algorithm based on the task requirement of QoS classifies the tasks into high QoS tasks and low QoS tasks. The grid resources based on the task constraints are classified into high QoS provision resources and low QoS provision resources. QoS Guided Prominent Value Tasks Scheduling Algorithm performs the better mapping between the tasks and the grid resources by computing the Prominent Value for the task. The tasks are ordered into the Prominent Value Set from minimum to the highest prominent value of the task. The current research problem in task scheduling is to bring out an efficient algorithm to reduce the overall completion time of the task [2]. The scope of this work is to propose an efficient task scheduling algorithm and categories the tasks into four groups based on QoS constraints and schedule the group that has the highest priority.

MATERIALS AND METHODS

A. Problem Definition

An application consists of 'n' independent task and a set of 'm' heterogeneous resources. The problem of mapping the 'n' tasks to the set of 'm' heterogeneous resources in a grid computing environment is an NP-Complete problem [2,9]. This paper proposes a new algorithm Task Group Scheduling Algorithm for solving the scheduling problem in a grid computing environment.

The order in which the tasks to be mapped to a set of resources determines the efficient scheduling which results in the reduced makespan. The proposed new algorithm TGS Algorithm provides an ordered set of tasks, which specifies the order in which the tasks to be scheduled to the set of 'm' resources. The proposed Task Group Scheduling Algorithm provides reduced makespan than the existing Min-min heuristic scheduling algorithm.

B. Proposed TGS Algorithm

The TGS Algorithm uses the idea of grouped tasks into four different groups based on the priority of the task and task attributes and applies QoS and then uses Min-Min algorithm to schedule tasks to the resources inside each group. The main idea of TGS Algorithm is to divide all tasks into groups based on the task attributes. Each group will have tasks with similar attributes. These groups will be ordered to schedule based on priority that is given to the task attributes. The first scheduled group will have tasks with high task attributes value or high priority compared to that of the other groups. Then in the chosen group the task with minimum completion time will be scheduled first.

The input of TGS Algorithm is number of independent tasks n and number of resources m. Each task has three attributes:

1. UserType (UT): type of users (class A, class B, and class C).
2. priortyTask(PT): priority of tasks (urgent, high, and low priority).
3. TaskLength (TL): defines the length of tasks (normal, long).

The proposed TGS Algorithm comprises of five categories:

1. UrgentUser& UrgentTask: includes tasks with user belong to class A and priority of task is urgent.
2. UrgentUser: includes tasks with user belonging to class A and the priority of the task depends on the load of the task.
3. UrgentTask: includes tasks with priority of task is urgent.
4. LongTask: includes long tasks with the priority of the task is high.
5. NormalTask: includes all remaining tasks with the priority of the task is low.

The order of priority of the five categories is UrgentUser&UrgentTask, UrgentUser, UrgentTask, LongTask and NormalTask. The four different groups based on these five categories are:

GroupI: includes tasks with user belong to class A and priority of task is urgent and then the Long Task with user belong to class A and priority of task is high and then the Normal Task with user belong to class A and priority of task is low.

GroupII: includes tasks with user belong to class B and class C and priority of task is urgent.

GroupIII: includes Long Tasks with user belong to class B and class C and priority of task is high.

GroupIV: includes Normal Tasks with user belong to class B and class C and priority of task is low.

If GroupI has tasks then these tasks should be scheduled first before tasks inside GroupII and so on.

MET matrix (Minimum Execution Time) is the matrix that stores the estimation of expected execution time of all tasks on all resources. MET matrix has number of rows that is equal to the number of tasks (n), and number of columns that is equal to the number of resources (m) and MET (i, j) is time that resource j needs to execute task i. MET matrix is initialized with random numbers, but should be taken into consideration whether the type of task is long or normal. Because if the task is long, the range of random time in MET matrix MET (i, j) needs to be higher than the range of time is if the task is normal.

After defining the input of algorithm, initializing MET matrix and distributing tasks into four groups, will start scheduling. If there is a new task need to execute, first it is needed to define which group it belongs based on its attributes and then this task is added to the decided group.

C. Simulation analysis

The proposed TGS Algorithm used Min-min algorithm to schedule tasks inside chosen group, while priority assigned to tasks is used to decide which group in TGS Algorithm will be scheduled first. The priority order of the groups to be scheduled is GroupI, GroupII, GroupIII and finally GroupIV. The GroupI includes tasks with user belong to class A, The task may comprises of Urgent Tasks, Long Task and Normal Task. So, In

GroupI, if it has Urgent Tasks then these tasks should be scheduled first before Long Tasks and Normal Tasks, and then the Long Tasks should be scheduled next and finally the Normal Tasks are scheduled. After scheduling the tasks in GroupI and then the tasks in the GroupII, GroupIII, and GroupIV are scheduled one after another. To evaluate the behavior of proposed TGS Algorithm and compare its performance with that of Min-min, two simulation programs were developed to simulate the two algorithms (TGS, and Min-min Algorithm). The simulation programs were developed using Java. To obtain accurate results the two simulation programs will use the same simulation parameters as shown in Table 1.

Table 1: Simulation Parameters

Simulation Parameters	Value
Number of tasks	200, 400, 800, 1200, 2400
Number of resources	50, 100
Percentage of tasks (Users Є Class A)	10%
Percentage of tasks (Users Є Class A)	20%
Percentage of tasks (Users Є Class A)	70%
Percentage of tasks (Size Є Long)	10%
Percentage of tasks (Size Є Normal)	90%
Percentage of tasks (Task Є Urgent)	35%
Percentage of tasks (Task Є High)	25%
Percentage of tasks (Task Є Low)	40%
Percentage of tasks (Users Є Class A, Task Є Urgent)	2.5%

Performance metrics makespan will be evaluated for the two algorithms:

makespan is the time duration taken from beginning of first task, start processing and end with last task finished the processing.

makespan=Te-Ts.

where Te is time of ending last task, and Ts is time of start first task.

RESULTS AND DISCUSSION

A. An Illustrative Example

To illustrate how the proposed algorithm TGS Algorithm schedules the tasks to the resources efficiently. The following illustration considers five tasks and two resources. The MET matrix for five tasks and two resources is given in Table 2.

The existing algorithm, Min-min algorithm schedules the task as shown in Table 3 .The task-resource selected pair is t1-r1 t3-r1 t4-r1 t2-r1 t5-r1. The makespan produced by Min-min algorithm is 24.2 as shown in Table 3.

To achieve efficient scheduling, the proposed algorithm TGS Algorithm groups the task into two groups GroupI and GroupII. GroupI consists of Long Tasks such as task t2 and t5. GroupII consists of Normal Tasks such as task t1, t3 and t4. The proposed algorithm TGS Algorithm first schedules the tasks in GroupI and then the tasks in GroupII. The proposed algorithm TGS Algorithm, schedules the task as shown in Table 4 .The task-resource selected pair is t2-r1 t5-r1 t1-r2 t3-r2 t4-r2. The makespan produced by the proposed algorithm TGS Algorithm is 20 as shown in Table 4. From Table 3 and Table 4 it is evident that the proposed algorithm TGS Algorithm achieves better makespan and improved resource utilization that the existing Min-min algorithm.

Table 2: Sample MET Matrix

task id	task length (MI)	r1 (5000 MIPS)	r2 (2000 MIPS)
t1	5000	1	2.5
t2	40000	8	20
t3	6000	1.2	3
t4	10000	2	5
t5	60000	12	30

Table 3: makespan calculation for the existing Min-min Algorithm

Step	t1		t2		t3		t4		t5		selected task & resource	Min CT
	r1	r2	r1	r2	r1	r2	r1	r2	r1	r2		
1	0+1=1	0+2.5 =2.5	0+8=8	0+20 =20	0+1.2 =1.2	0+3 =3	0+2 =2	0+5 =5	0+12 =12	0+30 =30	t1 & r1	1
2			1+8=9	0+20 =20	1+1.2 =2.2	0+3 =3	1+2 =3	0+5 =5	1+12 =13	0+30 =30	t3 & r1	2.2
3			2.2+8 =10.2	0+20 =20			2.2+2 =4.2	0+5 =5	2.2+12 =14.2	0+30 =30	t4 & r1	4.2
4			4.2+8 =12.2	0+20 =20					4.2+12 =16.2	0+30 =30	t2 & r1	12.2
5									12.2+12 =24.2	0+30 =30	t5 & r1	24.2
makespan=MAX(CT)												24.2

Table 4: makespan calculation for the proposed TGS Algorithm

Group	t2		t5		---		selected task & resource	Min CT
	r1	r2	r1	r2	----	----		
1	0+8=8	0+20 =20	0+12= 12	0+30 =30	---		t2 & r1	8
			8+12= 20	0+30 =30	---		t5 & r1	20
	t1		t3		t4		---	---
	r1	r2	r1	r2	r1	r2	---	---
2	20+1 =21	0+2.5 =2.5	20+ 1.2 =21.2	0+3 =3	20+2 =22	0+5 =5	t1 & r2	2.5
			20+ 1.2 =21.2	2.5+3 =5.5	20+2 =22	2.5+5 =7.5	t3 & r2	5.5
					20+2 =22	5.5+5 =10.5	t4 & r2	10.5
							makespan=MAX(CT)	20

B. Evaluation Parameters

Makespan

As shown in Fig. 1 the performance of Min-min algorithm is slightly lower than that of the proposed TGS Algorithm. This is because Min-min algorithm searches in the whole MET matrix for the resources executing tasks faster. So the search is done on the entire MET matrix. But the proposed TGS Algorithm searches in MET matrix for resource executing task with minimum completion time.

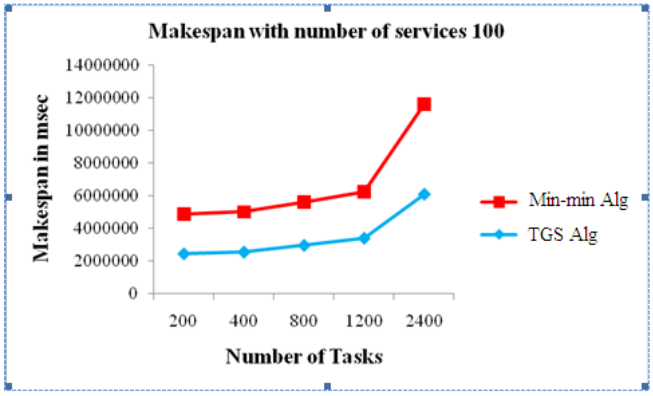

Figure 1: Comparison based on makespan

CONCLUSION AND FUTURE WORK

The objective of the paper is to define an algorithm used in grid computing which gets minimum completion time to all tasks (Urgent Tasks, Long Tasks, and Normal Tasks) with tasks with urgent priority, high priority to low priority. The proposed TGS Algorithm schedules tasks into resources with driven QoS. The proposed TGS Algorithm combines many types of task attributes (user type, task priority, task length) which are used to measure the priority of tasks.

The experimental results show that the proposed TGS Algorithm achieves minimum makespan compared to that of the Min-min algorithm. This is performed for number of resources equal to 50 and 100. Also, the number of tasks is varied from 200,400,800,1200,2400 . It's clear from the Fig that increases in number of resources from 50 to 100, leads to decrease in makespan. Also it appears that the proposed TGS Algorithm achieves minimum makespan than the Min-min algorithm. The proposed TGS Algorithm achieves load balancing and better resource utilization.

The proposed TGS Algorithm uses only three task attributes and applies QoS, and also the proposed TGS Algorithm only works for independent tasks and needs to queue all tasks first to make them into groups. So the future work is to increment the number of attributes that apply QoS in the algorithm and to make the algorithm to work with dependent tasks and also in real time.

REFERENCES

[1] D.D. Roure, M.A. Baker, N.R. Jennings, The evolution of the grid, in: F. Berman, G. Fox, T. Hey (Eds.), Grid Computing: Making the Global Infrastructure a Reality, John Wiley & Sons Ltd., 2003, pp. 65–100.

[2] D. Neumann, J. Stößer, C. Weinhardt, J. Nimis, A framework for commercial grids—economic and technical challenges, Journal of Grid Computing 6 (2008) 325–347.

[3] D R. Buyya, S. Venugopal, A Gentle Introduction to Grid Computing and Technologies, CSI Communications, July 2005.

[4] I. Foster, C. Kesselman, The Grid2: Blueprint for a New Computing Infrastructure, Elsevier, 2003.

[5] M. Gareym, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, WH Freeman & Co., San Francisco, 1979.

[6] R.Armstrong, D.Hensgen, and T.Kidd, "The Relative Performance of Various Mapping Algorithms is Independent of Sizable Variances in Run-time Predictions", In 7th IEEE Heterogeneous Computing Workshop(HCW" 98), pp. 79-87, 1998.

[7] R.F.Freund and H.J.Siegel,"Heterogeneous Processing", IEEE Computer , 26(6), pp. 13-17, 1993.

[8] R.F.Freund, and M.Gherrity, "Scheduling Resources in Multi-user Heterogeneous Computing Environment with Smart Net", In Proceedings of the 7th IEEE HCW, 1998.

[9] G.K.Kamalam, and Dr. V..Murali Bhaskaran, "New Enhanced Heuristic Min-Mean Scheduling Algorithm for Scheduling Meta-Tasks on Heterogeneous Grid Environment", European Journal of Scientific Research, Vol.70 No.3, pp. 423-430, 2012.

[10] G.K.Kamalam, and Dr. V..Murali Bhaskaran, "A New Heuristic Approach:Min-Mean Algorithm For Scheduling Meta-Tasks On Heterogeneous Computing Systems", IJCSNS International Journal of Computer Science and Network Security, Vol.10 No.1, pp. 24-31, 2010.

[11] G.K.Kamalam, and Dr. V..Murali Bhaskaran, "An Improved Min-Mean Heuristic Scheduling Algorithm for Mapping Independent Tasks on Heterogeneous Computing Environment", International Journal of Computational Cognition, Vol. 8, N0. 4, pp. 85-91, 2010.

[12] Dr.G.K.Kamalam, B.Anitha and S.Mohankumar, "Credit Score Tasks Scheduling Algorithm for Mapping a Set of Independent Tasks onto Heterogeneous Distributed Computing Grid Environment", International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE),Vol.20, No.2,pp.182-186, 2016.

[13] Dr.G.K.Kamalam, B.Anitha and S.Mohankumar, "QoS Guided Prominent Value Tasks Scheduling Algorithm in Computational Grid Environment", International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE),Vol.20, No.2,pp.187-191, 2016.

[14] V.Manju Bargavi, Dr.G.K.Kamalam, and B.Anitha , "Task Unique Credit System Based Scheduling Algorithm for Mapping Meta-Tasks on Heterogeneous Grid Environment", International Journal of Innovative Research in Computer Science and Engineering (IJIRCSE),Vol.1, No.13,pp.13-22, 2015.