International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 11 (2015) pp. 28635-28644

http://www.ripublication.com

ATSA: Advanced Task Scheduling Algorithm For Grid Computing Environment

S. Vaaheedha Kfatheen
Research Scholar, Bharathiar University, Coimbatore, Tamil Nadu, India.
E-Mail: abuthaheer67@gmail.com
Dr. M. Nazreen Banu, Professor, Dept of MCA,M.A.M. College of Engg, Trichirappalli, TN, India.
E-Mail:nazreentech@gmail.com

Abstract

Grid computing has its role in completing various different and difficult tasks at a time. Grid computing includes a huge number of computers or resources that has been connected together and share information among them to attain the task. The Grids uses all its resources efficiently to finish the tasks assigned to it within time. Some resource could complete the task before time and few could take more time to finish the task. Therefore, this reveals that each resource has its capacity in unique. The resource that completes its assigned task before time might not be utilized up to its maximum capacity. This could cause the minimum utilization of the resource in the grid. In order to increase the utilization of a resource and to make it as an optimized resource, the other tasks could be assigned continuously to that resource. A new task scheduling algorithm called "Advanced Task Scheduling Algorithm for Grid Environment" (ATSA) has been designed to reduce the resource idle time and makespan.. The proposed task scheduling algorithm is compared with other existing heuristics and the results are shown to prove that the new algorithm outperforms the other.

Keywords: Grid Computing, Task Scheduling, Heuristics, Makespan,

Introduction

Grid can be of many types like Manuscript grid, Column grid, Modular grid, Hierarchical grid. Each grid works up to its maximum in various environments [1]. The grid could be considered as the distributed system with its workloads. Grid computing includes the large number of resources that origins from different locations to finish the assigned tasks [2]. The assigned tasks have been introduced into task queue so that which task has to be started and allocated to which resource and at what

time would be easily known. The grid could use the tasks in the queue that makes scheduling more convenient. This task scheduling should be efficient to increase the performance of the resources. But task scheduling has its difficulties in resource allocation, allocation of time slots, scheduling constraints on tasks and machines and optimisation. It has many problems like single-machine problems, multi-machine problems, single-stage problems and multi-stage problems [4]. The other problems include tasks without precedence constraints that cause collaboration, non pre emption, condition that each machine could handle only one task at a time, each task could perform on only one machine [5]. Because of these problems task scheduling has become the major issue in grid computing. This problem could become a very big challenge for almost all the scheduling algorithms. Many scheduling algorithms have been previously designed to solve the problems. An algorithm has been designed that has the main objective as the make span minimization. Some algorithms have been designed to reduce the processing time of tasks by paying a certain amount [6]. Though such algorithms have been designed for an efficient task scheduling, there some problems that still exists. This proposed algorithm "Advanced Task Scheduling Algorithm for Grid Environment" (ATSA) has been designed for an efficient task scheduling to increase the utilization of resources and to reduce the response time.

The remainder of this paper is structured as follows: related works in Section 2, discussion about proposed algorithm in Section 3, discussion about the results and evaluation in Section 4 and the conclusion in Section 5.

Related Works

Kanate Ploydanai et. al. [6] proposed "Algorithm for Solving Task Shop Scheduling Problem Based on machine availability constraint". This algorithm has been focused to solve the task shop scheduling problem. It has been designed by considering the machine availability constraints. With the constraints, realistic make span has been calculated for factories to have breaking period during processing time. This method has been shown a good result for all the problems. In future, this has been extended to the flexible task shop scheduling problem that could be very complex.

Saeed Farzi [3] proposed an algorithm, "Efficient Task Scheduling in Grid Computing with Modified Artificial Fish Swarm Algorithm" for task scheduling in grid computing. It has established two problems like food concentration and structure of AF to optimize the task scheduling in modified AFSA. The food concentration has the principles of make span minimization and flow ime minimization. This method has considered the representation of encoding the solutions of the task scheduling into AFs. This algorithm has been compared with the Genetic algorithm and Simulated Annealing to identify the performance of MAFSA. The comparison result has shown that MAFSA is better than GA and SA.

Adán Hirales-Carbajal et. al. [7] proposed "Multiple Workflow Scheduling Strategies with User Run Time Estimates on a Grid" and concentrated on allocation strategies. It has taken dynamic site state information and workflow properties for task allocation. This method has the requirements such as type and amount of information. It has included stages like labelling, adaptive allocation, prioritization

and parallel machine scheduling to analyze the strategies. Regarding workflows and workflow tasks allocation, this method has identified various levels of information to introduce scheduling decisions. It has examined many two and four stage allocation strategies with EASY backfilling local scheduling algorithm.

It has been shown the high efficiency as independent task allocation policies with dynamic site state information by using adaptive allocation strategies. It has also given an explanation that DAG scheduling algorithms for single DAG and single machine settings would not be suitable for grid scheduling contexts.

It has shown the information regarding local scheduling that has been used for workflow allocation. This has the estimation of algorithms with task allocation strategies on user run time does not give stable results. By examining the overall grid performance, it has been found that priority based scheduling has given a better performance than the task allocation based on user run time estimates and local scheduling information. It has been concluded that simple scheduler like MaxAR with minimum information could provide higher performance for multiple workflow scheduling.

Yongcai Tao et. al. [8] proposed "Dependable Grid Workflow Scheduling Based on Resource Availability". It has introduced a dependable Grid workflow scheduling system with the Markov Chain-based resource availability prediction model. This DGWS scheduling algorithm is based on list scheduling and consists of three phases such as ranking, grouping and scheduling.

The evaluation of performance results have shown that DGWS has improved the number of tasks completion and also reduced the make span of workflow. Thus it has increased the workflow execution in dynamic grid environments. In future, it has been planned to provide perfect resource availability prediction model. Further, using the fault tolerance technology dependable workflow execution has to be researched.

Andrei Tchernykh et. al. [9] proposed an algorithm called "On-line Hierarchical Task Scheduling on Grids with Admissible Allocation". They have concentrated on non pre-emptive online scheduling on a grid that consists of several similar processors. The grid scheduling model has used two stages. At first, tasks have been allocated to a suitable machine and then each machine had the local scheduling.

At last they have examined the adaptive admissible allocation. The grid scheduling model had used two layered hierarchical structure which includes the main properties of grids. Therefore this method could have performed the future heuristic grid scheduling algorithm that could be implemented in real computational grids. In future, they have planned to evaluate the practical performance of this proposed method and based on heuristics comparison has to taken place with the existing grid scheduling.

Proposed Methodology- ATSA

This proposed algorithm considers the set of tasks and the resources. The tasks that are involved in this algorithm consist of different time taken to be finished by the resources even though the tasks have same capacity. Each resource in the grid has its own capacity to complete the tasks. With this available tasks and resources, this method initially selects the resource that has minimum execution time.

If a resource with minimum execution time has been found, the capacity of the resource is compared with the capacity of the task. If the resource has maximum capacity, immediately the task has been assigned to the resource otherwise the task has been segregated according to the capacity of the resource and then it has been assigned to the resource.

If the resource with minimum execution time is more, the resource with maximum capacity is selected and continued as said above.

Fig 1 shows the Architecture of proposed Advanced Task Scheduling Algorithm (ATSA)

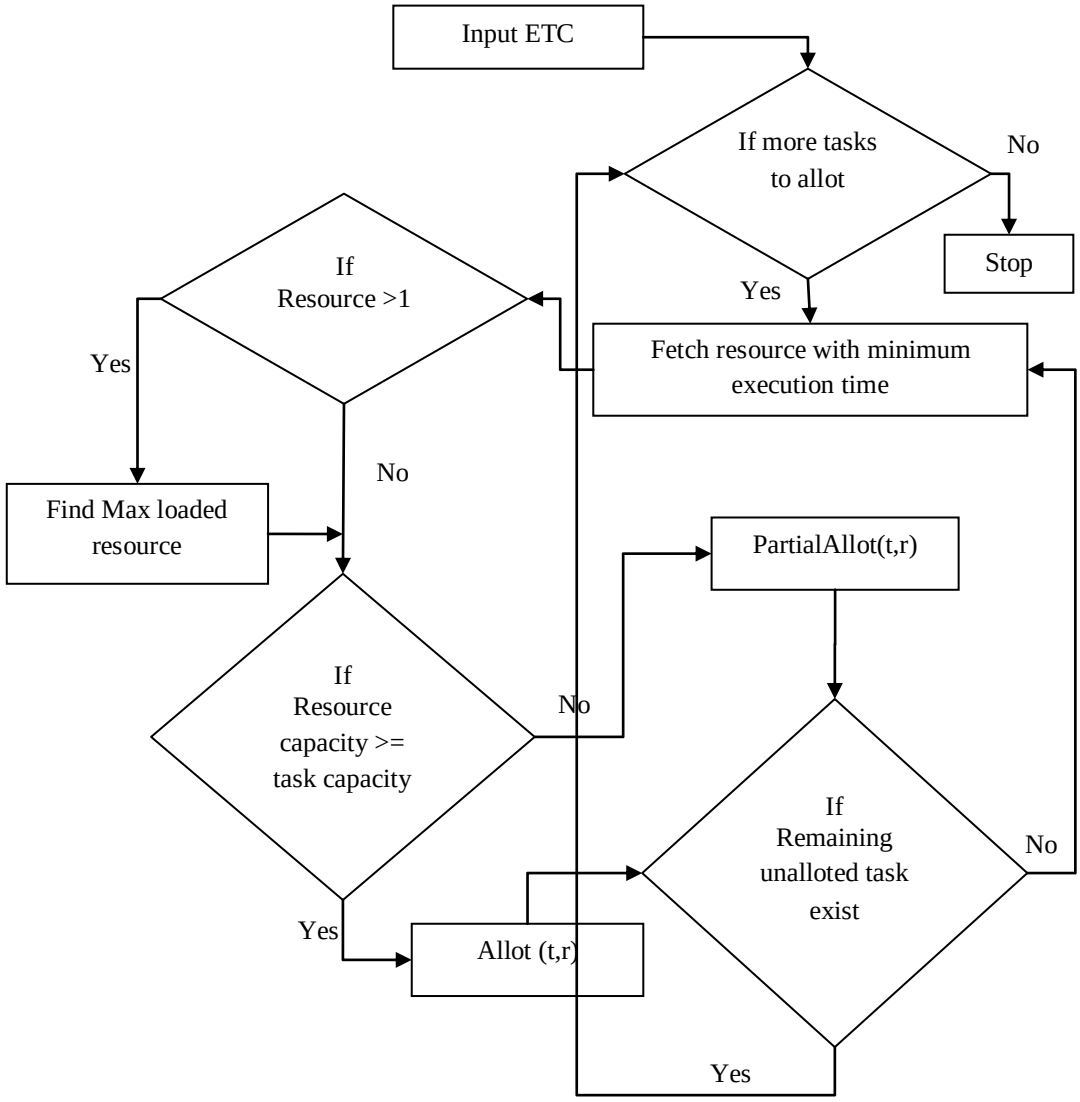


Figure 1: Architecture Diagram

The proposed algorithm has set of n tasks (T) and set of m resources (R). Earliest Task Completion time (ETC) has been generated with the taken tasks and the resources. Available Resource (AR) has obtained the resource with the minimum execution time from resources R for all the task t_i from T.

Min r $\in R \; \forall \; t_i$

The mathematical statement that is shown above has been used to select the resource with minimum execution time. If the count of r in AR is more than one, the resource with the maximum ability to complete the task has been selected. This selection has been done with the help of the below calculation.

MAX $\sum_{k=0}^{nor} load(ARk)$

After the selection of the resource, resource utilization (ru) and total weight of the task (tw) has been found. The resource utilization could be found using the total strength of the resource and utilized strength of the resource as shown below.

$\sum TS(AR) - \sum US(AR)$

The total weight (tw) of the task has been manipulated using total needs and completed needs of the task as shown below

$\sum TN(ti)$ - $\sum CN(ti)$

If the resource utilisation is greater than the total weight of the task, the task has been assigned to the resource. If the resource utilization is less than the total weight of the task, the task t_i has been split and then assigned to the resource.

If any task has been available without assigned to any of the resource, steps from 5 to 13 have been continued. Thus this proposed method increases the utilisation of resources in the grid.

Algorithm 1

ATSA ALGORITHM

1. Let T be the set of n tasks
2. Let R be the set of m resources
3. Let E be the ETC
4. Do with $t_i \in T \; \forall$ i = 0 to n
5. AR= Min r $\in R \; \forall \; t_i$
6. Do If ($Count \; r \in AR$ >1)
7. AR= MAX $\sum_{k=0}^{nor} load(ARk)$
8. ru=$\sum TS(AR) - \sum US(AR)$
9. tw= $\sum TN(ti)$ - $\sum CN(ti)$
10. If ru >= tw
11. ALLOT(t_i , AR)
12. Else
13. ALLOTPARTIAL(t_i,AR)
14. If $\sum unallot(ti) > 0$ goto step 5
15. Stop

Where
ti is each task in T, r is each resource in R
AR is the available resource
Nor is the number of available resources
ru is the resource utilisation
tw is the task weight
TS is the total strength
US is the utilized strength
TN is the total need
CN is the completed need

Results and Discussion

This proposed algorithm has been evaluated with an ETC. The sample ETC has been shown below.

Table 1: Sample ETC

(Time in MIPS)	**R1 (100)**	**R2 (20)**	**R3 (50)**
T0 (20)	4	3	1
T1 (100)	2	8	4
T2 (20)	8	9	2
T3 (30)	4	4	5

Table 1 has many tasks such as T0, T1, T2 and T3. In this, each task has different capacities where T0 has 20 MIPS, T1 has100 MIPS, T2 has 20 MIPS and T3 has 30 MIPS respectively. The resources taken are R1, R2 and R3. Resource R1 has the capacity to complete any task up to 100 MIPS. Resource R2 has the capacity to complete the task up to 20 MIPS. Resource R3 has the capacity to complete the task up to 50 MIPS.

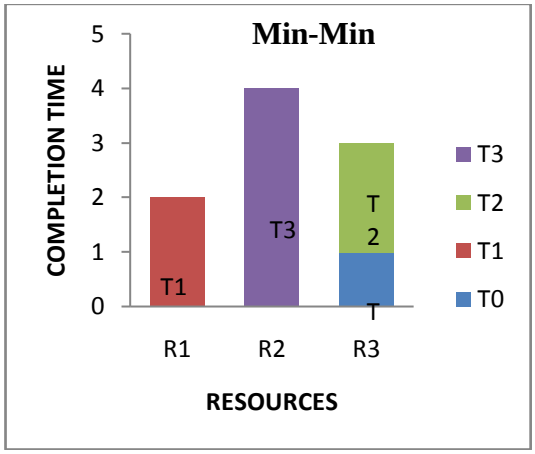
Task T0 has been assigned to resource R3 as it has minimum time taken of 1 minute to complete task T0. After the completion of task T0, resource R3 has the capacity to complete the task of 30 MIPS. Task T1 has been assigned to resource R1 as it has minimum time taken of 2 minutes to complete task T1. Now after the completion of task T1, resource R1 has used its full capacity to complete the task T1. Then Task T2 has been assigned to resource R3 as it has minimum time taken of 2 minutes to complete task T2. After the completion of task T2, resource R3 has the capacity to complete the task of 10 MIPS.

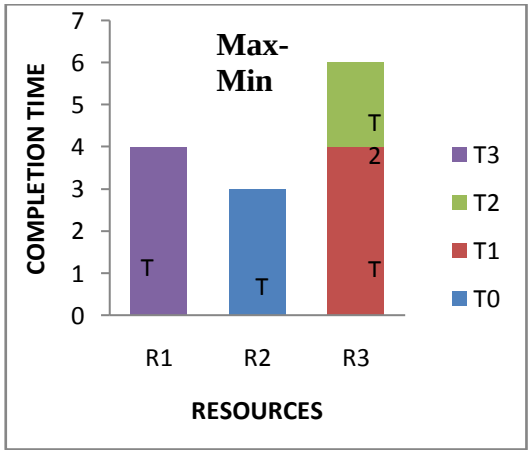
Finally Task T3 could be assigned to two resources such as resource R1 and R2 since both of them has minimum time taken of 4 minutes to complete the task. In this case the capacity of the resources is compared. As a result of comparison it has been noted that resource R1 has more capacity than resource R2. Therefore Task T3 has

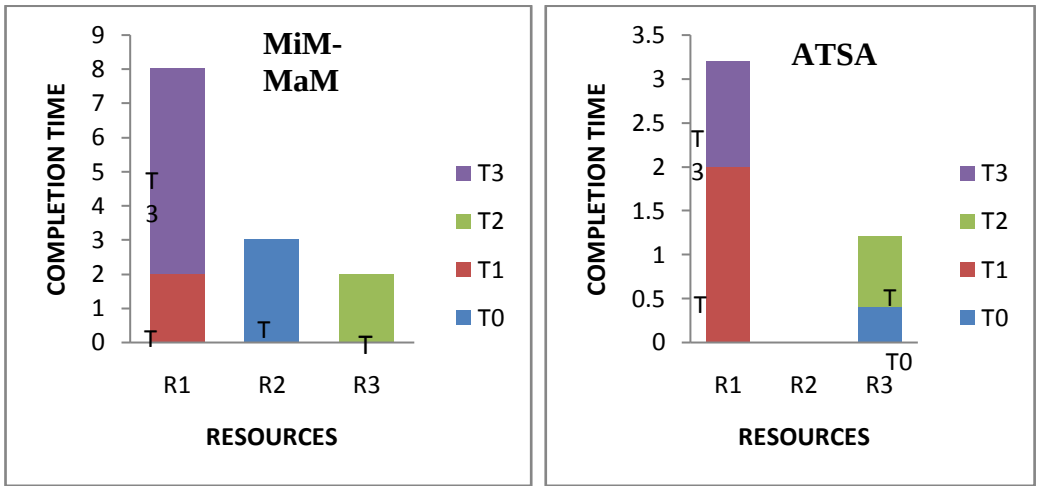
been assigned to resource R1. From the above analysis this proposed algorithm proves that it reduces the makespan.

Table 2: Comparison of ATSA with other methods

Task	Resource	Data used	Start Time	End Time
Min-Min				
T0	R3	N/A	0	1
T1	R1	N/A	0	2
T2	R3	N/A	1	3
T3	R2	N/A	0	**4**
Max-Min				
T3	R1	N/A	0	4
T1	R3	N/A	0	4
T2	R3	N/A	4	**6**
T0	R2	N/A	0	3
MiM-MaM				
T1	R1	N/A	0	2
T2	R3	N/A	0	2
T0	R2	N/A	0	3
T3	R1	N/A	2	**8**
ATSA				
T0	R3	20	0	0.4
T1	R1	100	0	2
T2	R3	20	0.4	1.2
T3	R1	30	2	**3.2**







It has also utilized the resources up to their optimization. But in this method still some of the resources have not been utilized for any kind of tasks. This could not lead to any issues instead it could reduce the cost for completion of the tasks. The proposed method has been compared with other different existing methods to find out the accuracy.

In Table 2 ATSA has been compared with methods such as Min-Min, Max-Min and MiM-MaM. Min-Min has the maximum time span of 4 after the completion of all the tasks. Max-Min has the maximum time span of 10 for the completion of the tasks. MiM-MaM has the maximum time span of 8 to finish the whole tasks. ATSA has the maximum time span of 3.2.

With this comparison it has been found out that proposed algorithm has less time span than the other algorithms. It has also been noted that the proposed algorithm has considered the amount of arithmetic calculations that has been assigned to each and every task but not the other methods.

Table 3: Makespan of different methods

Algorithms	**Make span**
Min-Min	**4**
Max-Min	**6**
MiM-MaM	**8**
ATSA	**3.2**

Table 3 shows the make span of different methods. Compared to all the other methods, proposed method could complete any tasks with minimum time taken with optimum resource utilization.

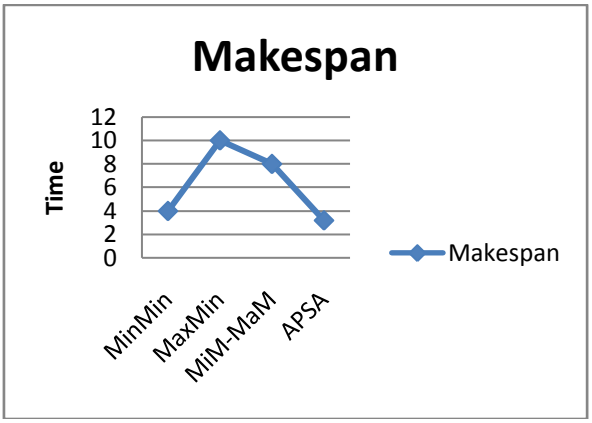


Figure 2: Graph of Make span

The above graph shows the time as X-axis and different existing methods as Y-axis.

The diamond shaped mark in the graph represents the finishing time of all the tasks (make span) of the given methods. In this graph except ATSA all the other methods have reached the high make span. ATSA has the lowest make span among them that provides the advantage of fast completion of the tasks by the resources.

Conclusion

In grid computing, various resources perform different types of tasks from different environment. The tasks are assigned to the resources after the completion of the previous task. On the other hand each resource in the grid should be utilized properly to complete the tasks efficiently. This proposed algorithm has achieved the optimum utilization of resources, the parallel assignment of tasks and it has given the lowest makespan among the other existing algorithm that has been discussed. But this algorithm has a drawback as it has not maintained the load balancing factor. In future this load balancing issue could be solved that might increase the efficiency of the algorithm.

References

[1] http://www.vanseodesign.com/web-design/grid-types/
[2] http://en.wikipedia.org/wiki/Grid_computing
[3] Saeed Farzi, ”Efficient Task Scheduling in Grid Computing with Modified Artificial Fish Swarm Algorithm”, International Journal of Computer Theory and Engineering, Vol. 1, No. 1, April 2009 1793-8201.
[4] http://www.cs.ubc.ca/labs/beta/Courses/CPSC532D-05/Slides/scheduling-chris.pdf

[5] Jacek Bta~ewicz, Wolfgang Domschke, Erwin Pesch, "The task shop scheduling problem: Conventional and new solution techniques", European Journal of Operational Research 93 (1996) 1-33.

[6] Kanate Ploydanai, Anan Mungwattana, "Algorithm for Solving Task Shop Scheduling Problem Based on machine availability constraint", International Journal on Computer Science and Engineering Vol. 02, No. 05, 2010, 1919-1925.

[7] Adán Hirales-Carbajal, Andrei Tchernykh, Ramin Yahyapour, José Luis González-García, Thomas Röblitz, Juan Manuel Ramírez, Alcaraz , "Multiple"Workflow Scheduling Strategies with User Run Time Estimates on a Grid", © Springer Science+Business Media B.V. 2012, DOI 10.1007/s10723-012-9215-6.

[8] Yongcai Tao, Hai Jin, Song Wu, Xuanhua Shi, Lei Shi, "Dependable Grid Workflow Scheduling Based on Resource Availability", © Springer Science+Business Media Dordrecht 2012, DOI 10.1007/s10723-012-9237-0.

[9] Andrei Tchernykh1, Uwe Schwiegelshohn, Ramin Yahyapour, Nikolai Kuzjurin, "On-line Hierarchical Task Scheduling on Grids with Admissible Allocation".