HCTL Open International Journal of Technology Innovations and Research (IJTIR)
http://ijtir.hctl.org
Volume 12, December 2014
e-ISSN: 2321-1814, ISBN (Print): 978-1-62951-791-9

QoS Based Task Scheduling Heuristic in Grid Computing using Fault Tolerance

Kailash Chandra Sethwal[1], Dr. Ajit Singh[2] and Gaurav Pareek[3]

kailash7988@gmail.com

Abstract

Grids are becoming more and more important to solve the problems and time calculations of science, industry, and engineering. QoS based task scheduling heuristic is clearly able to achieve the objectives like less execution time and resource usage. Also it takes into consideration the performance and reliability factors which are missing in traditional batch mode heuristics. Since the probability of failure increases with increasing number of components, fault tolerance is an essential characteristic of massively parallel systems.

Keywords

Grid Computing, Task scheduling, QoS (Quality of Services), Fault tolerance

Introduction

Grid Computing [1] is a large set of geographically distributed heterogeneous resources for solving problems in science and engineering, but to select the best resource for a given job is still a major problem in the area of Grid Computing. So a good resource selection technique in such environments is critical to obtain reasonable Quality of Service (QoS) [2]. The resource selection is based on the prior knowledge of the performance characteristics of the application on a particular resource and lag between a resource selection decision and the time the job appears in the system's monitoring facility will decrease the trust and performance of Grid. But In the process of resource selection in heterogeneous environment, it is difficult to find rules that determine the right selection decisions. However researchers use heuristics to help obtaining better selection decisions. Heuristics are approaches that help make right decisions; but they do not always produce the correct selection decisions.

[1]MTech Student, BTKIT Dwarahat, Almora, Uttarakhand, India. Email: kailash7988@gmail.com
[2]Associate Professor, BTKIT Dwarahat, Almora, Uttarakhand, India. Email: erajit@rediffmail.com
[3]Assistant Professor, BTKIT Dwarahat, Almora, Uttarakhand, India. Email: enpareek@gmail.com

In large grid computing system it is unwieldy for an individual to select the resources manually. So resource selection mechanism and scheduling of tasks into machines are required for better performance. Resource selection usually occurs after resource discovery phase. While the first phase filters out unwanted resources, this phase should determine from this large list the best set of resource(s) chosen to map the application. Resource selection is one of the important and key concepts in grid. The selection process made by the requester is based on the scheduler. The Grid scheduling problem is viewed in two-level, at the first level meta-scheduler allocates tasks to sites and the second level local scheduler schedules and assigns the task to the local computing elements. In the higher level scheduling the central scheduler will decide where the task to be assigned and the lower level scheduler will decide where the task to be executed. The use of grid computing as a platform for running parallel applications is a promising area of research. The ability to allocate a huge amount of re-sources for parallel application (thousands of Internet connected machines) and to do so at lower cost than traditional alternatives (based parallel supercomputers) is one of the main attractive in the grid computing.

Resources in grid systems are heterogeneous, geographically distributed, belong to different administrative domains and apply different management policies. The roles of resource selection mechanisms are to identify, select and allocate the most suitable resources for a given set of tasks. So in this paper taxonomy was presented that facilitates identifying and classifying the mechanisms used in the implementation of grid resource selection process, as well as describing the most significant features of grid resource selection mechanisms. The benefit of this taxonomy was to highlight the main aspects of the selection mechanisms, which can help researchers and developers of grid resource management systems.

Related Work

QoS [2] based task scheduling approach is best suit for the batch model heuristics [3]. In batch mode, all tasks are groped as batches and then scheduled as group, that's why it have better advantage to map task to best resource based on job and resource characteristics. So following are some of important batch heuristics model of Grid scheduling which have been taken into consideration in this research.

Min-Min

The completion time of all of the tasks which are unassigned ($1 \leq j \leq n$) on all of the present computing machines ($1 \leq i \leq m$) is utilised for the calculation of the minimum completion time (MCT_i) of task Ti on machine Mi*. Afterwards, the task which proves out to have the minimum value of MCT_i is identified as T*={min(MCT_i) for ($1 \leq i \leq m$) } and then it is assigned to the machine M*. Simultaneously the other task T* is eliminated from the prepared list of tasks that are unassigned and the workload for machine M* is updated again. The above described procedure is repeated again and again until the unassigned task list gets completely eliminated [4].

Max-Min

In this particular method, in accordance to the similarities to min-min method, completion time (CT) of all of the jobs that are unassigned ($1 \leq j \leq n$) on the machines that are available ($1 \leq i \leq m$) is then taken into account to calculate the minimum completion time (MCT_i) of task T_j on machine M_i*. Afterwards, of all the tasks the task which gives maximum of MCT_i will be identified asT*= {max (MCT_i) for ($1 \leq i \leq m$) } and will then be provided to the machine M*. Simultaneously the other task T* is tried and removed from the list of tasks that are unassigned and then the workload of machine M* is updated again. The procedure defined above is tried and repeated until the unassigned task list gets completely eliminated [4].

Suffrage

The difference between the second minimum completion time and the first minimum completion time for a particular job can be defined as the suffrage for that job. The Suffrage method tries and allocates the jobs that suffer the most in terms of expected completion time. Suffrage is calculated out for all of the unassigned jobs in this defined method and out of these jobs the job which consists of the maximum amount of suffrage value is provided to the available machine which gives out the very first minimum completion time. That job is then removed from the list of unassigned jobs, the machine workload is finally updated and the above defined job allocation cycle is repeated again and again till the complete list of the jobs that are unassigned gets completely eliminated [4].

Fault Tolerance in Grid

In Grid Environment failure can be due to resource overloaded condition, system running out of memory and fault in network and fabric computing component. Fault can be tolerate in two level Task level and workflow Level[5].Task-level technique handle failure in workflow of task or running time of task, while workflow-level technique manipulate workflow structure for dealing with failure[5].Task-level techniques widely use in parallel and distributed systems for handling failures. They can be categorized into *retry, alternate resource,* ***Checkpoint*** and *replication.* Workflow-level techniques include *alternate task*, *redundancy*, *user-defined exception handling* and *rescue workflow* [5].

Proposed Work

In this section author propose QoS based heurist task scheduling algorithm preventing fault.

```
Proposed Work
I.  Calculating the computation time for task.
II. Calculating the communication time for task.
III.    Calculate Communication to Computation Ratio (CCR)
IV. If CCR is low
    a.  Task is computational based and can be submit on Remote site.
    b.  Call FT (Ti).
V.  Else if CCR is high
    a.  Task is communicational based and can be submit on Local site.
    b.  Call FT (Ti).
VI. End
```

```
Algorithm FT (Ti)
1.  Rank the resource based on previous performance.
2.  Submit the job having highest rank.
3.  If failure occur
    a.  Recover failure using Checkpoint Fault Recovery.
    b.  If not success go to step 3.
4.  End.
```

Basic Terminology Used in Proposed Algorithm:

1. **Communicational Based Task:** These type of task required high bandwidth, like transferring a file in grid environment.

2. **Computational Based Task:** These type of task required high power CPU, like solving complex computation based problem.

3. **Communication Time:** Communication Time consider that time in which the communication based jobs are completed.

4. **Computational Time:** Computational Time consider that time in which the computational based jobs are completed.

5. **CCR (Communication-computation Ratio:** The ratio of Communicational cost and computational cost.

Experimental Results

In this section author give experimental result for given proposed algorithm which is simulated on grid environment using GridSim 5.0 simulator, he used Pentium-4 based intel processor with CPU clock speed 2.00 GHz, 2 GB RAM running on Window 7

e-ISSN: 2321-1814, ISBN (Print): 978-1-62951-791-9

operating system. Computational results of existing batch mode heuristics i.e. Min-Min, Max-Min and Suffrage have been discussed in the form of 4 scenarios. For every scenario, the number of jobs has been increased while the number of resources has been kept constant.

Scenario 1: For scenario 1, the number of jobs which has been taken into consideration is 100 while the number of resources is 10.

			Performance	
	No of Jobs	**No of Resources**	**Makespan**	**Resource Utilization**
Min-Min	100	10	520	85%
Max-Min	100	10	500	82%
Suffrage	100	10	540	90%
Proposed QoS Scheduling	100	10	490	93%

Table 1: Result of Scenario 1.

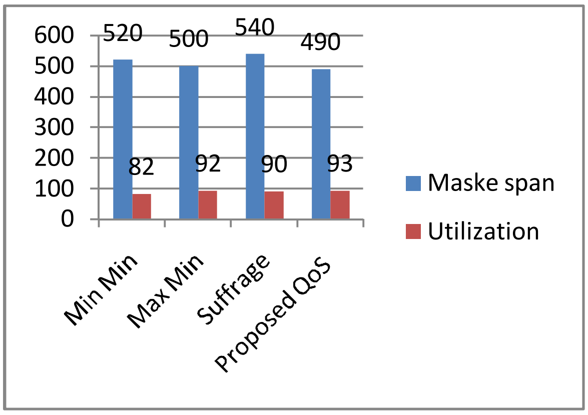

Figure 1: Result of Scenario 1.

Scenario 2: For scenario 2, the number of jobs which has been taken into consideration is 300 while the number of resources is 10.

			Performance	
	No of Jobs	**No of Resources**	**Makespan**	**Resource Utilization**
Min-Min	300	10	1750	84%
Max-Min	300	10	1720	91.7%
Suffrage	300	10	1500	86.2%
Proposed QoS Scheduling	300	10	1495	91.9%

Table 2: Result of Scenario 2.

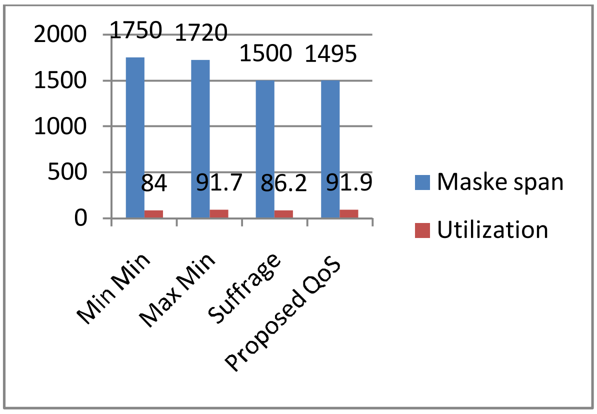

Figure 2: Result of Scenario 1.

Scenario 3: For scenario 3, the number of jobs which has been taken into consideration is 500 while the number of resources is 10.

			Performance	
	No of Jobs	No of Resources	Makespan	Resource Utilization
Min-Min	500	10	2915	78%
Max-Min	500	10	2500	96.2%
Suffrage	500	10	2875	80.3%
Proposed QoS Scheduling	500	10	2480	97.3%

Table 3: Result of Scenario 3.

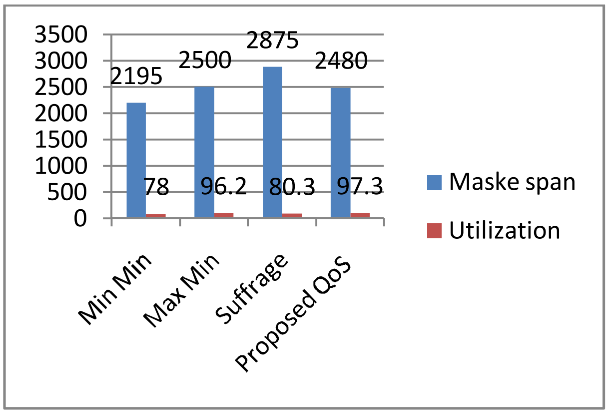

Figure 3: Result of Scenario 3.

e-ISSN: 2321-1814, ISBN (Print): 978-1-62951-791-9

Scenario 4: For scenario 4, the number of jobs which has been taken into consideration is 1000 while the number of resources is 10.

			Performance	
	No of Jobs	**No of Resources**	**Makespan**	**Resource Utilization**
Min-Min	1000	10	5000	70%
Max-Min	1000	10	5635	95.8%
Suffrage	1000	10	5725	72%
Proposed QoS Scheduling	1000	10	4890	96.1%

Table 4: Result of Scenario 4.

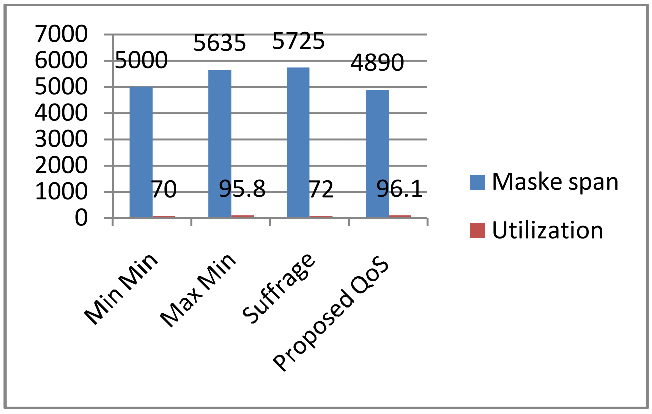

Figure 4: Result of Scenario 4.

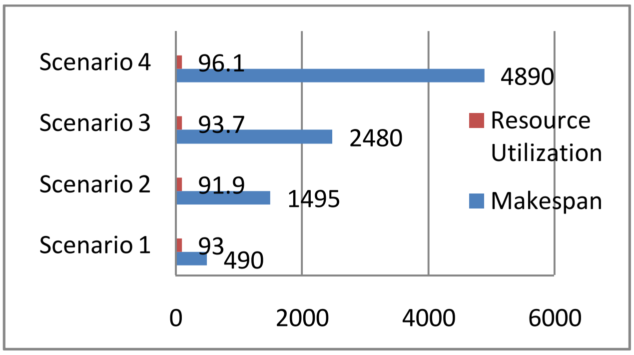

Figure 5: Comparison of all four scenarios.

From the above diagram (Figure 5) it is clear that performance of proposed algorithm is better than previous heuristic batch algorithms based on makespan and resource utilization.

Conclusion and Future Work

In this research, we describe the implementation of a quality service providing fault-tolerance heuristic task scheduling for computational grids. We present algorithms for provisioning and migration of virtual machines and tasks, are the main strategies for maximum utilization of resource and recovery from fault in efficient way. We consider the performance demand models, reliability of resource models, computational resources to determine good fault tolerance strategy. The main goal was to provide quality of service in grid environment, in real time as the time and the expected probability of success, and do it in a way that is transparent to the scientists.

In the future, we plan to use more than basic network services reliability estimates to perform more reliable Fault tolerance policy on grids. We also plan to extend our algorithms to other types of workflows such as bioinformatics, which have different characteristics; many small workflow tasks / applications unlike weather workflows, which have a small number of large workflows tasks / applications.

References

[1] I. Foster, C. Kesselman, *The Grid Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Pub., 1998.

[2] Hai Jin, Hanhua Chen, Zhipeng Lu, QoS Optimizing Model and Solving for Composite Service in CGSP Job Manager, Chinese Journal of Computers, 28(4): 578-588, 2005.

[3] Fatos Xhafa, Ajith Abrahamb, *Computational models and heuristic methods for Grid scheduling problems*, Future Generation Computer Systems 26 (2010) 608-621, Springer, pp.173-214, 2008.

[4] Rajendra Sahu and Anand K Chaturvedi, *Many-Objective Comparison of Twelve Grid Scheduling Heuristics*, International Journal of Computer Applications (0975 – 8887),Volume 13– No.6, January 2011

[5] Jia Yu and Rajkumar Buyya*, A Taxonomy of Workflow Management Systems for Grid Computing,* Grid Computing and Distributed Systems (GRIDS) Laboratory, Dec 2014 http://www.gridbus.org .