ISSN No. 0976-5697

Volume 4, No. 9, July-August 2013

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Implementation of FMFRS (Fault Tolerant Most fitting Resource Scheduling) algorithm in Real time system

Harkiran Kaur
M. Tech Computer Science,
Department of CSE, RBIET, Kharar
Punjab Technical University, India.
harkiran7237@gmail.com

Er. Deepankar Verma
AP M. Tech Computer Science,
Department of CSE, RBIET,
Kharar
Punjab Technical University, India.

Abstract: In computational Grid, fault tolerance is an imperative issue to be considered during job scheduling. Due to the widespread use of resources, systems are highly prone to errors and failures. Hence fault tolerance plays a key role in grid to avoid the problem of unreliability. The two main techniques for implementing fault tolerance in grid environment are check pointing and replication. This paper proposes a real time approach to a replication technique named as FMFRS (Fault Tolerant most fitting resource scheduling algorithm) to improve the fault tolerance of the fittest resource scheduling algorithm. The proposed method is to improve the fault tolerance by using fittest resource scheduling algorithm, by scheduling the job in coordination with job replication when the resource has low reliability and checking the parameters like Fault Tolerance capacity and Node's Reliability. Based on the reliability index of the resource, the resource is identified as critical.

Keywords: Computational Grid, Fault tolerance, Task Replication, Job Scheduling

I. INTRODUCTION

A. Grid Computing:

Grid computing is a form of distributed computing that involves coordinating and sharing computational power, data, and storage and network resources across dynamic and geographically dispersed organizations. Management of these resources becomes complex as the resources are geographically distributed, heterogeneous in nature, owned by different individual or organizations with their own policies, have different access models, and have dynamically varying loads and availability [1].

B. Fault Tolerance:

Fault tolerance is the ability of a system to perform its function correctly even in the presence of faults and it makes the system more dependable. It is a survival attribute to preserve the delivery of expected services despite the presence of fault caused errors within the system itself, errors are detected and corrected, and permanent fault are located and removed while the system continues to deliver acceptable service. [3]

C. Task scheduling:

The scheduling problem in grid is one of the most studied problems in the research community. In order to perform the scheduling process, the grid scheduler has to follow a series of steps:

a. Collecting information of jobs submitted to the grid,
b. Collecting available resource information from resource broker,
c. Computation of the mapping of jobs to selected resources,
d. Jobs allocating according to the mapping, and
e. Monitoring of job completion [7].

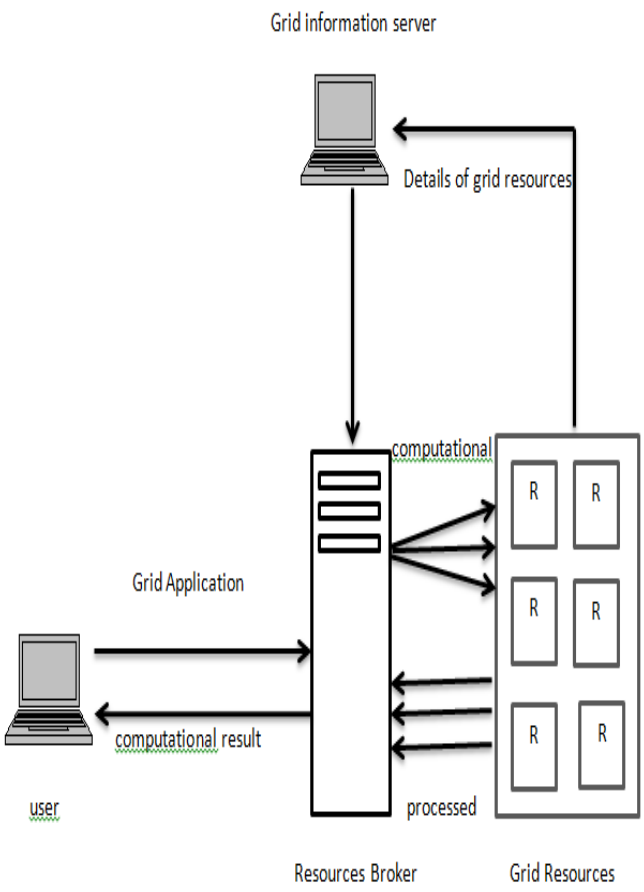

Figure 1: Basic grid scheduling system

D. Socket Programming:

A **socket** is a communication connection point (endpoint) that you can name and address in a network. Sockets allow us to exchange information between processes on the same machine or across a network, distribute work to the most efficient machine and allow access to centralized data easily. [8]

a. How do sockets work?:

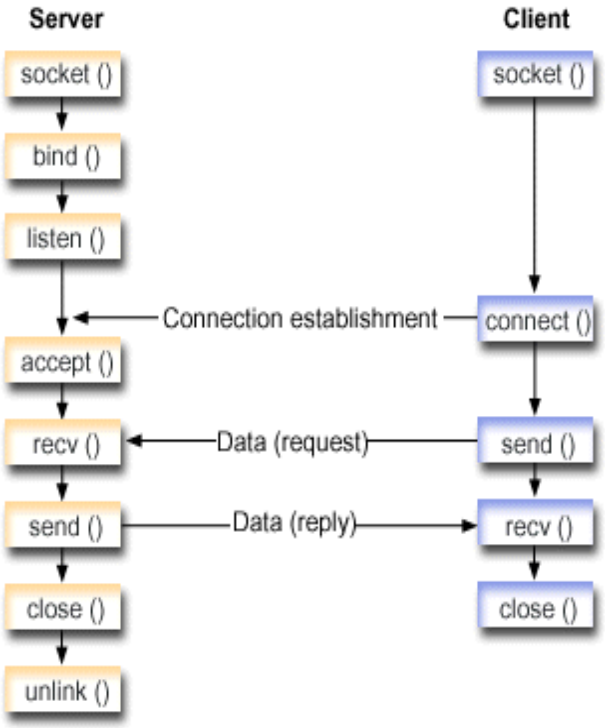

Figure 2: working of Socket

II. RELATED WORK

Grid resource scheduling is one of the most significant research issues today. Most of the existing scheduling algorithms do not consider the resource failure during scheduling which would eventually increase the execution time of the task. Some of the resources scheduling algorithms are as follows:

Fault Tolerant most Fitting Resource Scheduling Algorithm (FMFRS) for Computational Grid(A. Shamila Ebenezer, K. Baskaran

In this proposed method, to improve the fault tolerance of the fittest resource scheduling algorithm, by scheduling the job in coordination with job replication when the resource has low reliability. Based on the reliability index of the resource, the resource is identified as critical. The tasks are scheduled based on the criticality of the resources [1].

Scheduling Tasks on Most Suitable Fault tolerant Resource for Execution in Computational Grid (Jairam Naik K., K. Vijaya Kumar, N. Satyanarayana**)**

To improve the performance and throughput of the system, it is required to assign the tasks to a suitable and lower fault rate processor. Therefore, if a task is allocated to a resource without considering the performance factor, resource fault rate, the overall execution time will increases and throughput will decreases obviously. To solve this problem, a task scheduling algorithm that finds suitable and low fault rate resource for task execution was introduced in this paper [2].

An Efficient approach to Task Scheduling in Computational Grids (Dr. G. Sudha Sadasivam, Viji Rajendran.V**):**

The objective of this paper is to develop a scheduling strategy using job groups that optimizes the utilization of processing capabilities of grid resource and reduces the total time taken to process user jobs [6].

Charlotte: Metacomputing on the Web(Arash Baratloo, Mehmet Karaul, Zvi Kedem, Peter Wyckoff):

Metacomputing prototype Charlotte uses the eager scheduling mechanism where the tasks are assigned to the free nodes randomly to keep them busy. There is a manager process which schedules the jobs. The parallel jobs having concurrent routines are split and given to different volunteers for execution. When all the routines are allocated and still if volunteers are free the unfinished jobs will be reassigned to the volunteers. If a machine fails, the job is migrated and re executed in another machine automatically [5].

In literature, many other scheduling algorithms are also proposed, such as RR (Round Robin), FPLTF (Fastest Processor to Largest Processor First), WQR (Work Queue without Replication), and MFTF (Most Fit Task First). The common features among these algorithms are the following:

a. All tasks are queued before the processors are available.
b. When processors are available, tasks are broken down and processed according to the algorithms' specifications.
c. These processes are repeated until all tasks are completed.

The only difference among them is the priority setting up mechanism for task processing. In the following, we will introduce some scheduling algorithms grouped separately by whether it is a static and a dynamic scheduling algorithm [4].

III. PROPOSED WORK

The previous work has implemented the Fault Tolerant Most Fitting Resource Scheduling (FMFRS) Algorithm in java based simulating environment. In our proposed method, we are going to implement the following algorithm in real time using grids by making grid of 4 computers.

FMFRS Algorithm:

Step 1: The task named as 'Ti' is submitted for scheduling.

Step 2: The resources are categorized into different discrete levels based on the predicted execution time taken by each resource for a particular task Ti.

Step 3: The Reliability Index value for each resource is calculated. The resource with higher reliability index value tends to be more reliable. In order to decrease the average waiting time for task Ti and to prevent resource failure, the resource with higher reliability index value is used first.

Step 4: The closest level Li Based on the desired execution time of the Task Ti is identified.

Step 5: Any node Ri within the Level Li with minimum load is allocated to the Task Ti.

Step 6: If the resource Ri is marked Critical, then Task Ti is replicated and submitted to another resource Rj with minimum load in the same level Li.

After implementation of this algorithm, we will test our system on the following parameters- crash, transient failure, execution time, node's reliability, fault rate.

IV. CONCLUSION

Earlier, the fault tolerant most fitting resource scheduling (FMFRS) algorithm was implemented in a java

based grid simulator and the results were evaluated. The algorithm was tested for transient failures only. In our proposed work, we are going to implement the same algorithm using socket programming in java and are going to test the system on the parameters i.e. crash, transient failure, execution time, node's reliability, fault rate.

V. REFERENCES

[1]. A. Shamila Ebenezer, K. Baskaran (Sept 2012) "Fault Tolerant most Fitting Resource Scheduling Algorithm (FMFRS) for Computational Grid", European Journal of Scientific Research ISSN 1450-216X Vol. 86 No 4 September, 2012, pp.468-473

[2]. Jairam Naik K., K. Vijaya Kumar and N. Satyanarayana, " Scheduling Tasks on Most Suitable Fault tolerant Resource for Execution in Computational Grid", International Journal of Grid and Distributed Computing Vol. 5, No. 3, September, 2012

[3]. P. Latchoumy, P. Sheik Abdul Khader, "Survey on fault Tolerance in grid computing", International Journal of Computer Science & Engineering Survey (IJCSES) Vol.2, No.4, November 2011

[4]. Ruay-Shiung Chang, Chun-Fu Lin, Jen-Jom Chen, "Selecting the most fitting resource for task execution", Future Generation Computer Systems 27 (2011) 227–231

[5]. Baratloo, M. Karaul, Z. Kedem, P. Wyckoff, "Charlotte: Metacomputing on the Web", Proceedings of the 9th International Conference on Parallel and Distributed Computing Systems, 1996.

[6]. Dr. G. Sudha Sadasivam, Viji Rajendran.V, "An Efficient approach to Task Scheduling in Computational Grids", International Journal of Computer Science and Applications, 2009 Technomathematics Research Foundation Vol. 6, No. 1, pg. 53 – 69

[7]. Abhang Swati Ashok and Durole Pankaj Hari, "Grid Computing: Various Job Scheduling Strategies", Emerging Trends in Computer Science and Information Technology - 2012(ETCSIT2012) Proceedings published in International Journal of Computer Applications® (IJCA)

[8]. Retrieved from http://pic.dhe.ibm.com/infocenter/iseries/v6r1m0/index.jsp?topic=/rzab6/rzab6soxoverview.htm