Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 7, July 2013, pg.190 – 195

RESEARCH ARTICLE

An Efficient Resource Management and Scheduling Technique for Fault Tolerance in Grid Computing

Deeptanoy Ghosh[1], Ramandeep Singh[2], Jimmy Laishram[3]
[1]CSE, Lovely Professional University, India
[2]CSE, Lovely Professional University, India
[3]CSE, Lovely Professional University, India

[1] deeptanoyghosh@gmail.com; [2] ramandeepsingh@lpu.co.in; [3] jimmy.laishram@gmail.com

Abstract— In Grid computing, scheduling techniques are based on queues, easy to implement but not very effective in mapping jobs after well determined parameters. If a certain user needs a specific resource, usually she/he has to log on directly on specific resource and to access the scheduling service. A more advanced method is to have a schedule-based approach. This approach is usually most effective when the submitted jobs are in batch mode (a new task will trigger either rescheduling, either will be placed in a temporary queue if it has a low priority, until a significant amount of jobs are queued, depending on the scheduling algorithm used). The proposed technique will give the user a single view where the user can log into the Grid resource management system, submit jobs and check the status of his jobs just like he does when he submits a job to any single resource.

Key Terms: - Alea2; EASY; EDF; FCFS; Grid

I. INTRODUCTION

In today's complex world of superfast computing, computers have become extremely stronger. Home-based personal computers are powerful enough to run complex applications. Numerous complex scientific experiments, genome matching, advanced modeling scenarios, a wide variety of simulations, astronomical research, real-time personal portfolio management, and complex scientific & business modeling scenarios which require huge amount of computational resources. Grid Computing is born to satisfy some of these aforementioned requirements. Grid computing is a distributed computing, emerging as a wide-scale infrastructure that promises to support resource sharing with coordinated problem solving in dynamic and multi-institutional virtual organization. In Grid, resources are heterogeneous and dynamic. In order to utilize dynamic and heterogeneous resources, a scheduling strategy should be able to continually adapt to the changes and properly distribute the workload and data amounts scheduled to each node. Researchers have stated that serious difficulty in concurrent programming of a distributed system has occurred in terms of dealing with scheduling and load balancing, that may consist of heterogeneous computers [3] [4] [5] [6]. This paper is organized as follows: Section II, provides the research background and related work for grid and cluster scheduling. Section III, formalizes the problems at a cluster node in a grid system. Section IV, proposes scheduling strategy with Alea2 simulator. In Section V, we discuss the performance of the algorithms and their workings and highlight their advantages with a simulation study. Section 6, provides the conclusions and lastly references.

II. LITERATURE REVIEW

In this literature review, we will provide a review of a limited number of studies related to the grid computing with resource management and scheduling. Law, 2000, proposed a randomized resource discovery algorithm called "absorption algorithm" based on a strongly connected graph[7]. Authors discussed, this algorithm is developed in two stages. In stage 1, a network of n nodes is partitioned into clusters; each cluster has its leader node and all nodes in a cluster know their leader. At this stage, an ultimate leader is determined in the network. In stage 2, the ultimate leader of the network broadcasts the pointers to the each member of the network in one time step. The authors claimed that this absorption algorithm takes O(log n) steps time complexity, can send O(nlog n) number of messages, and can pass O(n2 log n) number of pointers with high probability. Nithiapidary M, 2005, present a scheduling strategy that performs dynamic job grouping activity at runtime and convey the detailed analysis by running simulations [8]. He proposed job processing granularity size is introduced to facilitate the job grouping activity in determining the total amount of jobs that can be processed in a resource within a specified time. The job grouping strategy results in increased performance in terms of low processing time and cost if it is applied to a Grid application with a large number of jobs where each user job holds small processing requirements. Sending/receiving each small job individually to/from the resources will increase the total communication time and cost. K. Somasundaram, 2008, proposed a new scheduling algorithm called Optimum Resource Constraint (ORC) scheduling algorithm which includes the combination of both the Best fit allocation and Round Robin scheduling to allocate the jobs in queue pool [9]. ORC improved the efficiency of load balancing and dynamicity capability of the Grid resources. They measured the performance of ORC algorithm on a Grid computing with maximum number of jobs and nodes. The ORC scheduling reduces the waiting time of job and allocates the jobs to processor based on its capability and increase the processing time of job. The proposed method will calculate the capability of jobs and processor to allocate the jobs. It will reduce the waiting time of jobs in queue and turnaround time. Thus the overall performance of the Grid has increased. Rosemarry P, 2012, cited that, if we provide an optimize algorithm to queue of the scheduler using various scheduling methods like Shortest Job First, First in First out, Round robin then the job scheduling system is responsible to select best suitable machines in a grid for user jobs [10]. Management and scheduling system generates job schedules for each machine in the grid by taking static restrictions and dynamic parameters of jobs and machines into consideration. The main purpose was to develop an efficient job scheduling algorithm to maximize the resource utilization and minimize processing time of the jobs. All queues can be optimized by using various scheduling algorithms depending upon the performance criteria to be improved based on response time and throughput.

III. PROBLEM FORMULATION

Most of the scheduling techniques used in computing grids are based on queues, easy to implement but not very effective in mapping jobs after well determined parameters. If a certain user needs a specific resource (for performance reasons or special requirements) usually she/he has to log on directly on specific resource and to access the scheduling service. A more advanced method is to have a schedule-based approach. This allow advanced reservation of resources, dynamically remapping jobs on more appropriate resources and optimization of the schedule based on optimization methods (such as local search based methods). This approach is usually most effective when the submitted jobs are in batch mode (a new task will trigger either rescheduling, either will be placed in a temporary queue if it has a low priority, until a significant amount of jobs are queued, depending on the scheduling algorithm used).

IV. PROPOSED WORK

Experimenting with large numbers of grid scheduling scenarios is impractical in real world systems because of the limited number of various resources available for testing sessions. Moreover, the grid environment it is not controlled so multiple repetitive experiments are hard to deploy. This impact negatively the experimenter's capacity to evaluate the scheduling algorithms and strategies employed. A testbed platform can be used for testing, but is has visible disadvantages: it is small, quite expensive; it is time consuming to build and not too diverse. The possible tested scenarios are limited in scope and size. To avoid this, a framework for deterministic modeling and simulation of resources, applications, users and algorithms must be used for scheduling strategies evaluation. The most common way to achieve this is using a software simulator that will provide the necessary tools for modeling, simulation and visualization of the result data. In this research, Alea 2 simulator is used. Alea 2 is an event-based simulator used for testing scheduling algorithms under various conditions (different resources and job types, dynamic changes in the environment, etc.). This grid simulator is based on GridSim Toolkit and represents an extension that incorporates better tools for scheduling algorithm implementation visualization capability and a higher speed of simulations. In this paper, the approach followed consists of a use of multiple priority based scheduling with different algorithm used in each queue. The multiple queue

scheduling algorithms, allows a process to move between queues. Every queue has its own scheduling algorithm. There are three scheduling algorithm (First Come First Serve, EDF, EASY-BF) used. This scheme leaves I/O-bound and interactive processes in the higher-priority queues. In this paper, the three algorithms are used in loops for each queue in order to find out a best combination for a particular type of job and according to certain performance criteria.

Algorithm:

Step 1: Start

Step2: Load machine data, failure data, job data

Step 3: Submit job using Alea and Gridsim simulation

Step 4: Listen to machine failure

if (machine failure? YES)

a) Accept result and reschedule jobs using

i) FCFS: calculate machine usage per cluster/ machine

ii) EDF: ---do----

iii) EASY: ---do---

b) Submit reschedules jobs

c) Goto step 4

Else

goto step 5

Step 5: Continue with the job submitting

Step 6: If no data?

Print result (machine usage/ chart)

Step 7: End.

Flowchart of proposed algorithm:

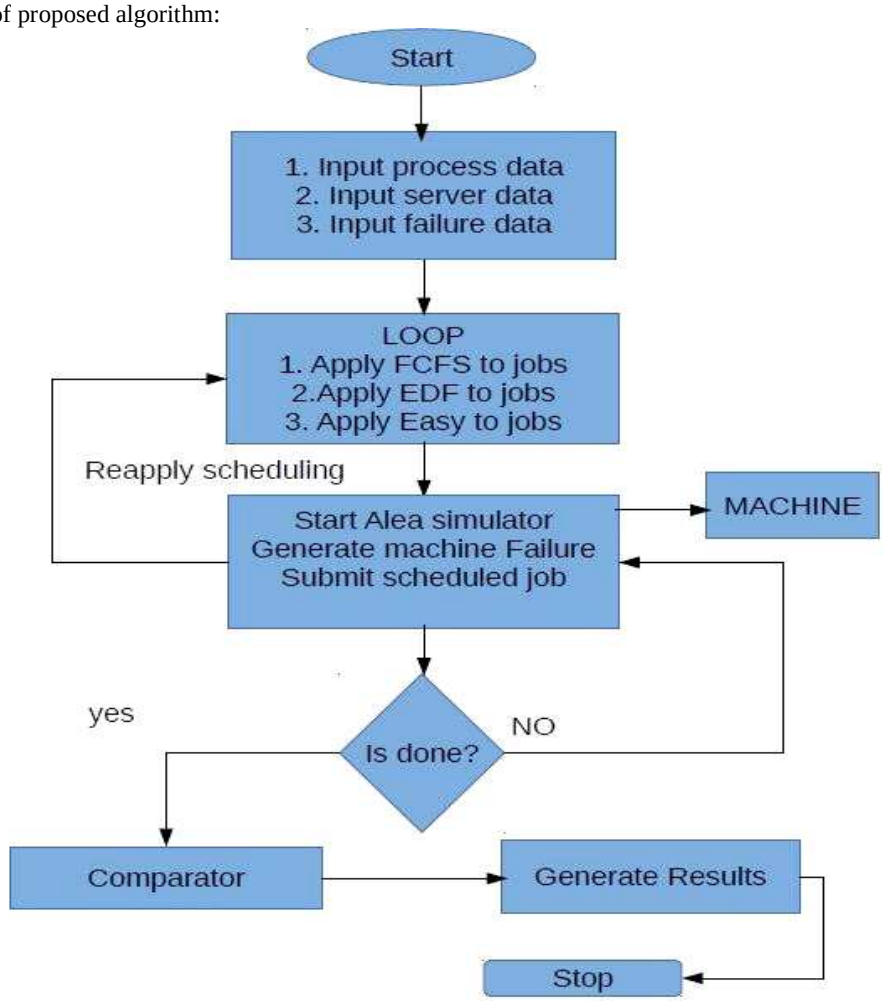

Figure 1: Flowchart of the algorithm

V. EXPERIMENTAL RESULTS

The algorithms were implemented in the Alea 2 grid simulator for testing. The settings assume that a job can span a resource, but cannot span multiple resources. A resource can run concurrently multiple jobs if there is enough processing elements to support the jobs' requirements. The tested set contains 12000 jobs processed by a FCFS algorithm [11] [12], EDF [13] and EASY [14] [15] algorithm. First, we simulate involving complex real-life data set from the Czech national Grid infrastructure MetaCentrum. MetaCentrum to demonstrate the simulation capability of the Alea 2. These data allow us to perform very realistic simulations, involving all characteristics. For example, we use precise information concerning the machine parameters, the information about machine failures and restarts, or the specific job requirements concerning the target machine properties.

All experiments were performed on Intel Centrino Duo 2.2GHz PC with 3GB of RAM. Unless otherwise indicated, the JVM (Java Virtual Machine) was limited by 1GB of available RAM. The experiment involved 12000 jobs that were originally executed on 14 clusters having 1152 processors.

Through the experiment, we have compared three different scheduling algorithms: FCFS, EDF and EASY-BF. The visualization for the implementation of these algorithms is described below in detail with appropriate screenshots.

Comparison of FCFS, EDF and EASY-BF using complex data set.

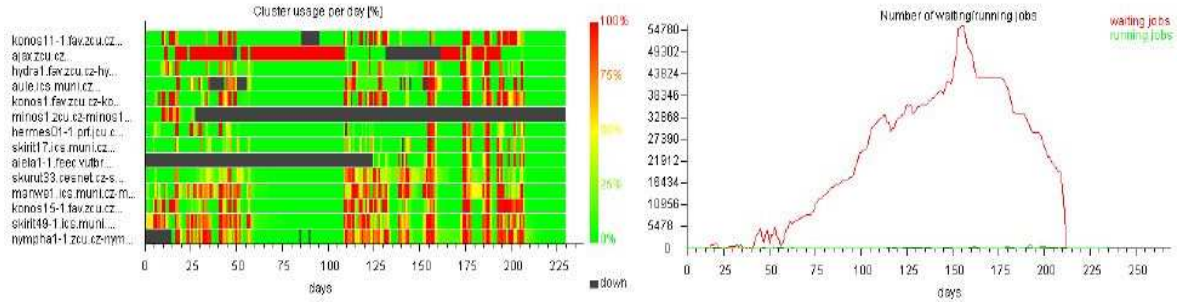

Figure 2: FCFS

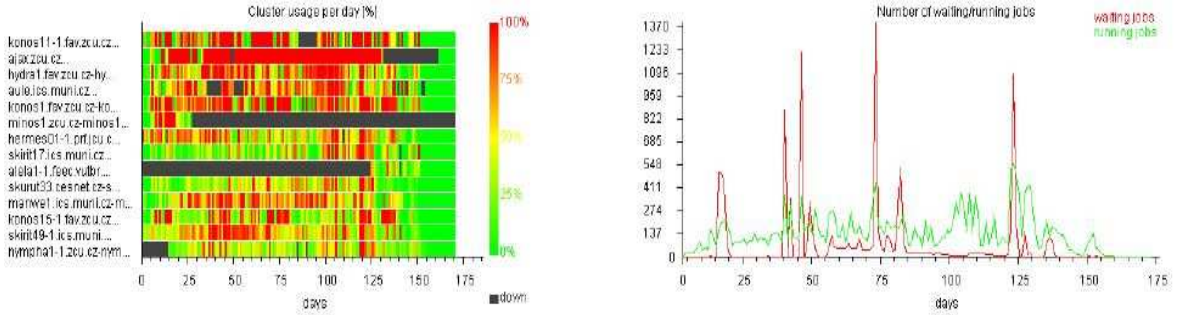

Figure 3: EDF

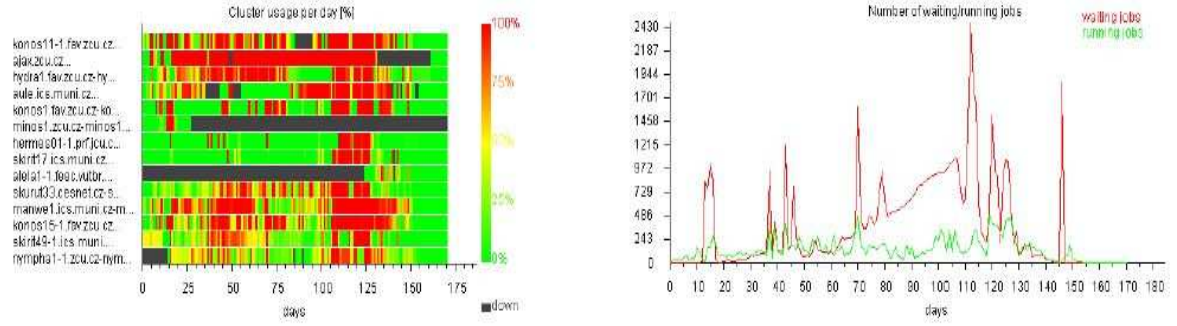

Figure 4: EASY-BF

From the simulation result, it can be observed that in terms of waiting jobs versus running jobs, FCFS algorithm have more queued jobs – meaning that more jobs can be delayed from execution and consequently the QoS level is lower (the quality of service restrictions are not meet). Moreover, because of better scheduling capabilities, the requests for processing units by the incoming jobs are much better balanced and the jobs are better placed to be executed and terminated on time. These graphs nicely demonstrate major differences among the algorithms. Concerning the machine usage as expected FCFS generates very poor results. FCFS is not able to utilize available resources when the first job in the queue requires some specific and currently unavailable machine(s). At this point, other more flexible jobs in the queue can be executed increasing the machine

utilization. This is the main goal of the EASY-BF algorithm. As we can see, EASY-BF is able to increase the machine usage by using the backfilling approach. Still, EASY-BF does not allow delaying the execution start of the first job in the queue, which restricts it from making more aggressive decisions that would increase the utilization even more. EDF algorithm does not apply such restrictions and thanks to the application of a more efficient schedule-based approach it produces the best results.

VI. CONCLUSION AND FUTURE SCOPE

For the simulation purpose, Alea 2 simulator has been successfully used in our work which focuses on the area of cluster and Grid resource scheduling and was able to simulate scheduling under different setups involving various data sets and objective functions [16] [17] [18]. Alea 2 has a visualization interface that allows faster debugging and tuning of the studied algorithms.. Through the experiment it has found that FCFS is not able to schedule jobs fluently. It generates huge peak of waiting jobs during the time. For the same reason, the resulting makespan is also much higher than in the remaining algorithms. On the other hand EASY-BF is capable of a higher resource utilization and reduction of the number of waiting jobs through the time. EDF produces the best results. As can be seen, these several other graphical outputs help the user to understand and compare the scheduling process of different scheduling algorithms. Since Grid Computing is the most popular research area in distributed environment now days because of the improving needs of users day by day. A standard still has to be set in the field. A number of techniques are available but with disadvantages of their own along with advantages. This paper will help in improving the efficiency of the grid system as well as fault tolerance capability.

REFERENCES

[1] Abramson D, Buyya Rajkumar, Giddy J, A computational economy for grid computing and its implementation in the Nimrod-G resource broker, Future Generation Computing Systems, 18(8), 2002, 1061-1074.

[2] Abhang Swati Ashok, Durole Pankaj Hari, Grid Computing: Various Job Scheduling Strategies, Emerging Trends in Computer Science and Information Technology, Proceedings published in International Journal of Computer Applications, 2012.

[3] Buyya R, Abramson D, Stockinger H, Giddy J, Economic models for resource management and scheduling in Grid computing, The Journal of Concurrency and Computation 14 (13-15), 2002, 1507-1542.

[4] Buyya R, Venugopal S, A gentle introduction to grid computing and technologies, CSI Communications 9, 9–19, 2005, July.

[5] Garg S.K, Siegel H, Buyya R, Time and cost trade-off management for scheduling parallel applications on utility grids, 2009, Future Generation Computer Systems.

[6] Safwana Haque and Isah Abdul Azeez, Resource Management in Grid Computing: A Review. Greener Journal of Science Engineering and Technological Research ISSN: 2276-7835 2012, Vol. 2 (1), pp. 024-031.

[7] Ching Law, Kai-Yeung Siu, An O(log n) Randomized Resource Discovery Algorithm, MIT-AUTOID-TR-004, 2000.

[8] Nithiapidary Muthuvelu, Junyang Liu, Nay Lin Soe, Srikumar Venugopal, Anthony Sulistio and Rajkumar Buyya, A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids, 2005.

[9] K. Somasundaram and S. Radhakrishnan, Node Allocation In Grid Computing Using Optimal Resource Constraint (ORC) Scheduling, International Journal of Computer Science and Network Security, VOL.8 No.6.

[10] Rosemarry P, Ravinder Singh, Dilip Sisodia and Payal Singhal, GROUPING BASED JOB SCHEDULING ALGORITHM USING PRIORITY QUEUE AND HYBRID ALGORITHM IN GRID COMPUTING, International Journal of Grid Computing & Applications, Vol.3, No.4, 2012.

[11] Zafril Rizal M Azmi, Wan Nurulsafawati Wan Manan, Kamalrulnizam Abu Bakar, Abdul Hanan Abdullah, and Mohd Shahir Shamsir, Scheduling Grid Jobs Using Priority Rule Algorithms and Gap Filling Techniques, International Journal of Advanced Science and Technology, 2011, Vol. 37.

[12] A.D.Techiouba, G.Capannini, Ranieri Baraglia, D.Puppin, and M.Pasquali, BACKFILLING STRATEGIES FOR SCHEDULING STREAMS OF JOBS ON COMPUTATIONAL FARMS.

[13] Zafril Rizal M.Azmi, Kamalrulnizam Abu Bakar, Abdul Hanan Abdullah, and Mohd Shahir Shamsir, Early Gap-Early Deadline First (EG-EDF) Scheduling Technique with Simulated Annealing Optimizer for Grid Computing.

[14] Ranieri Baraglia, Patrizio Dazzi, Giancarlo Pagano, and Gabriele Capannini, A Multi-criteria Job

Scheduling Framework for Large Computing Farms.

[15] Sivakumar Viswanathan, Bharadwaj Veeravalli, and Thomas G. Robertazzi, Resource-Aware Distributed Scheduling Strategies for Large-Scale Computational Cluster/Grid Systems, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, Vol. 16, 2007.

[16] Raksha Sharma, Vishnu Kant Soni, Manoj Kumar Mishra, Prachet Bhuyan, 2010, A Survey of Job Scheduling and Resource Management in Grid Computing, World Academy of Science, Engineering and Technology.

[17] Kaur H, A SYSTEMATIC APPROACH TO SIMULATE AND COMPARE THE PERFORMANCE OF STATIC VS DYNAMIC TASK SCHEDULING ALGORITHM ON GRIDSIM, International Journal of Computer Science & Information Technology, 2012, Vol. 2 No. 11, 880 ISSN No. 2231-2471.

[18] Mrs. Radha, and Dr.V.Sumathy, A Detailed Study of Resource Scheduling and Fault Tolerance in Grid, IJCSI International Journal of Computer Science Issues, 2011, Vol. 8, Issue 6, No 2, ISSN (Online): 1694-0814.

