ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 1, Issue 6, August 2012

Analysis of Data Grid in Grid Computing Based On Random Search Using Gridsim

R.Saranya

Department of ECE, Anand Institute of Higher Technology, Kazhipattur, Chennai

Abstract— **This paper firstly summarizes and defines data grid models and the process of job scheduling, simultaneously analyzes the time and cost of job execution in the data grid. The job scheduling strategy influences the QoS of the data grid immediately and then proposes a design proposal of the job scheduling simulator of data grid based on a grid simulator named GridSim and introduces the architecture, process and key technologies of the job scheduling simulator. By using this simulator we can determine the resource failure so that the performance of the data grid can be analysed. This involves implementation of job scheduling in a grid for handling resource failure using GridSim simulator. Moreover, the resources may suffer failures, and all of this obviously would affect the performance received by the users. In this paper we present an extension to one of the most popular grid simulators GridSim to support variable resource availability. Through which the overall performance can be determined. Finally, it proves that this scheduling simulator can satisfy the need of research on the data grid optimization in grid computing.**

Keywords- **Resource Failure, Allocation Policy, Execution Time, Random Search**

I. INTRODUCTION

A Grid is a collection of distributed computing resources available over a local or wide area network that appears to an end user or application as one large virtual computing system. Grid computing is a form of distributed computing that involves coordinating and sharing computing, application, data, storage, or network resources across dynamic and geographically dispersed organizations. Grid technologies promise to change the way organizations tackle complex computational problems. The Grid can in fact improve the quality of service for the end-user. As mentioned above, a typical example for a computational Grid is currently the collaboration of high-performance computing sites. The participating compute centres have usually a local user community which gains access to additional remote compute resources by joining the Grid. Such a sharing of computational jobs is a major application of Grids. While Grid technology is becoming more common and several computational Grids have already been established, little is yet known about the current and future user demand in such a Grid environment.

II.SYSTEM ARCHITECTURE

We employed a layered and modular architecture for Grid simulation to leverage existing technologies and manage them as separate components. A multi-layer architecture and abstraction for the development of GridSim platform and its applications is shown in Figure1. The first layer is concerned with the scalable Java interface and the runtime machinery, called JVM (Java Virtual Machine), whose implementation is available for single and multiprocessor systems including clusters .The second layer is concerned with a basic discrete-event infrastructure built using the interfaces provided by the first layer. One of the popular discrete-event infrastructure implementations available in Java is SimJava. Recently, a distributed implementation of SimJava was also made available. The third layer is concerned with modeling and simulation of core Grid entities such as resources, information services, and so on; application model, uniform access interface, and primitives application modeling and framework for creating higher level entities. The GridSim toolkit focuses on this layer that simulates system entities using the discrete-event services offered by the lower-level infrastructure. The fourth layer is concerned with the simulation of resource aggregators called Grid resource brokers or schedulers. The final layer is focused on application and resource modeling with different scenarios
using the services provided by the two lower-level layers for evaluating scheduling and resource management policies, heuristics, and algorithms.

III. SCHEDULING MODELS IN DATA GRID

Based on the grid entity defined above, simulating process of data grid job scheduling is shown. Firstly it sets experiment parameters, initializes GridSim package, then employs the user entity to create jobs; after created job, submit jobs to the user broker in the form of event transmission, then carries out the operations of job scheduling, distributing and recycling in turn. This simulator employs a method of message inquiry to realize the job scheduling simulation. If resource failure occurs, the job allocated to the particular node will be assigned to the other node .that allocation is based on the performance of the system. Here the performance is based on the system configuration. After the user broker receives a new job, it inquires each known computing resource broker about time and cost, if the job executing in this resource. After all resource brokers return their time and cost information, the user broker carries out the job scheduling strategy to distribute jobs. To overcome the disadvantage we use random search algorithm. It will consider based on the performance of the system in the cluster. If the performance is higher, then that node will be taken into account and that node will be assigned a job. It is a simple methods based on the notion of randomly searching over the domain of interest. Random search methods are perhaps the simplest methods of stochastic optimization in such a setting and can be quite effective in many problems. Their relative simplicity is an appealing feature to both practitioners and theoreticians. These direct random search methods have a number of advantages relative to most other search methods. The advantages include relative ease of coding in software. The computing environments comprise heterogeneous resources (PCs, workstations, clusters, supercomputers), fabric management systems (single system image OS, queuing systems) and policies, and applications (scientific, engineering, and commercial) with varied requirements (CPU, input/output (I/O), memory and/or network intensive). The users: producers (resource owners) and consumers (end-users) have different goals, objectives, strategies, and demand patterns. More importantly both resources and end-users are geographically distributed with different time zones.

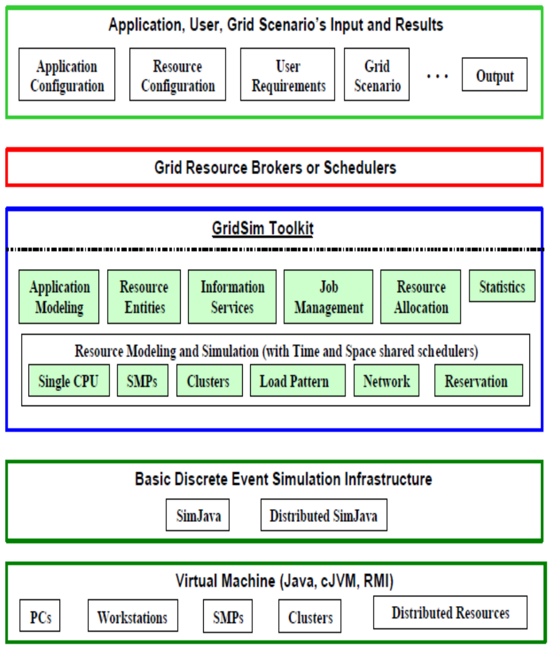

Figure 1. Architecture for GRIDSIM Platform

IV. RANDOM SEARCH ALGORITHM

In artificial intelligence or decision-making adversarial searching representing the possible positions with a tree is a common method. This generally works well when there are fixed rules and small state descriptions, such as what is encountered with a board game. Nodes of the tree typically represent the board positions. The game tree is the set of possible future board positions, where each node is a description of the board configuration. Each link in the tree represents a move by one of the players down to the next level and each alternating level represents the other player's possible situations. An exhaustive search is often impossible, even on powerful machines. Instead, a look-ahead procedure that will evaluate upcoming moves is needed. This requires the identification of all possible legal moves, perhaps some filter to sort out patently wrong moves that can be immediately identified, a function to evaluate the outcome, and a way to prune bad moves. Legal moves may be selected using some kind of random procedure, or biased in terms of some metric or rule or predefined knowledge. One evaluation procedure, known as minimax, focuses on a manner of position evaluation where one player has a better position through some kind of a heuristic. The goal is minimizing the opponent's score while maintaining as high a score as possible. The metrics involved can be weighted to consider the differences between move freedom and piece count, for example. This method assumes players will select the best option available. It functions by checking the bottom of the tree and performing evaluation, then going back up to select paths that will prevent or avoid the best moves for the opponent. Because tree searches alternate layers of players' moves, the minimax search alternates between *mini*mizing levels and *max*imizing layers of the tree. By reverse engineering the moves that led to the enemy's worst follow-up in this manner and allowed for the player's best advantage a successful path is deduced and the appropriate move for the level is made.

V. SYSTEM MODULES

The initialization module is responsible for the initialization operation of entire simulation, including initializing GridSim package and setting system parameters. Grid resources creation module creates resources for the simulation of grid job scheduling algorithms, including computing resource, storage resources as well as file resources. The network topology creation module is responsible to create the network topology. The topology can be edited by the user artificially, or be loaded automatically by files and randomized algorithms. The grid user creation module is responsible to create grid users, create a certain amount of jobs for each grid user, and simultaneously describe parameters of job execution. The job scheduling module firstly receives jobs submitted by users, update status of grid resources and user jobs, then execute assigned job scheduling algorithm to carry on the simulation. The statistical analysis module records the performance indicators of the job scheduling algorithm, and outputs them in the form of text or files to analyze and evaluate. And also gives the entire process of data grid job scheduling simulation: After starting GridSim, firstly carries on initialization of GridSim packages and configure system parameters including user quantity, resource quantity, statistical parameters and so on. Then creates resources, network environment and users. After user submitting jobs to the job scheduling module, loads job scheduling algorithm to carry on the simulation; finally analyzes the simulation result by the statistical analysis module and returns the result to the user.

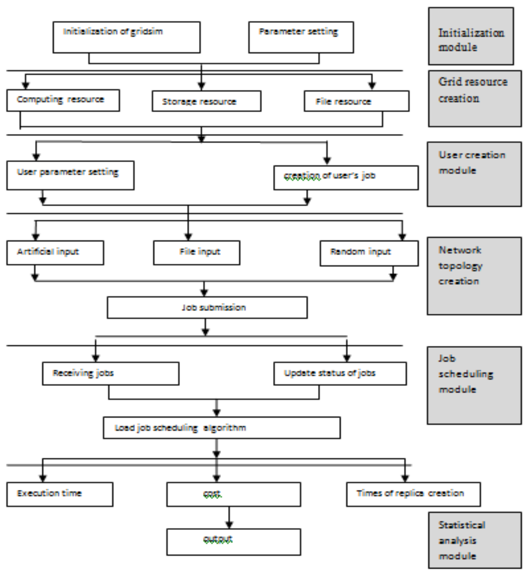

Figure 2. Modules of the system

VI. KEY COMPONENTS OF A SYSTEM

Grid simulator GridSim is based on the discrete event package SimJava2, defines multiple network entities, including the user, the resource, the job, the replica supervisor, the replica catalog and so on. These entities in GridSim communicate through event transmission mechanism. In order to realize simulation of the data grid job scheduling, some related components should be defined on the basis of existing entities. Below the paper introduce them separately. In the actual grid environment, the user broker is equal to the grid job scheduling. The user broker is used to receive jobs submitted by users, execute the job scheduling strategy to submit jobs to corresponding resources. Meanwhile it is responsible to log job status and execution. In this simulator, a class named DataGridBroker is defined as the user agent entity. In this class, there are following methods, schedule() load scheduling algorithm carry on the job scheduling and dispatch() according to the scheduling result, distribute jobs to corresponding resources. The resource broker records resource attributes and the status information, predicts cost of job executing in this resource according to its load information, which provides the reference for the resource scheduling. In this simulator, a class named BrokeDataGridResource is defined to realize related functions of the resource broker. To convenient, this class extends the class BrokeResource in the gridbroke. It realizes prediction of the transmission time and cost of data-intensity jobs. Because the length is limited, the process to realize the resources agent is no long to be introduced.

VII. STATUS OF JOBS

Through analysis above, the execution of data intensity jobs can be regarded as two parts of data transmission and job computing. Accordingly, this paper defines the data grid job status as the five statuses of created, blocked, ready, executing and success. Created status after the job creates and the blocked job has submitted but its data needed have not yet been prepared. In the ready status the data needed have already be prepared, but the computing resource has not yet gained. Executing the status of executing job. Success status after the job executed successfully. This simulator mainly statistics the job execution time, the execution cost, and the resource distribution result. Execution time includes job submission time, beginning time of data transmission end time of data transmission, beginning time of job computing, end time of job computing as well as actual computing time. Execution cost includes file transmission cost and job computing cost. Resource distribution includes the computing resource distributed to the job as well as the file request. Its represented in figure 3 and figure 5.

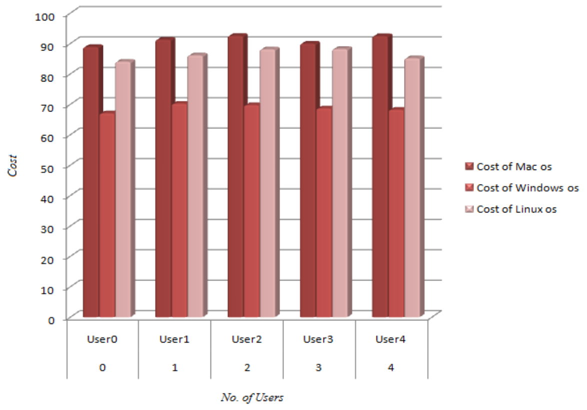

Figure 3. Cost comparison diagram

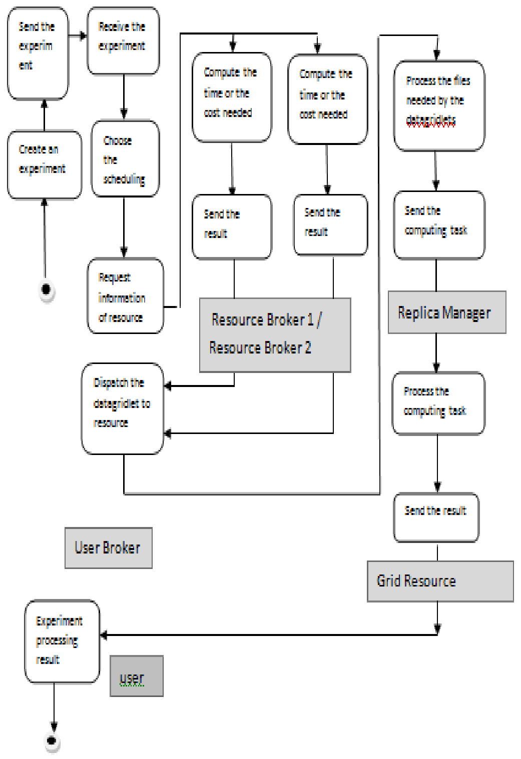
Send the experiment
Receive the experiment
Create an experiment
Choose the scheduling
Request information of resource
Compute the time or the cost needed
Compute the time or the cost needed
Send the result
Send the result
Resource Broker 1 / Resource Broker 2
Process the files needed by the datagridlets
Send the computing task
Replica Manager
Process the computing task
Send the result
Dispatch the datagridlet to resource
User Broker
Grid Resource
Experiment processing result
user

Figure 4. Simulation flowchart of scheduling process

The GridSim toolkit allows modelling and simulation of entities in parallel and distributed computing (PDC) systems-users, applications, resources, and resource brokers (schedulers) for design and evaluation of scheduling algorithms. It provides a comprehensive facility for creating different classes of heterogeneous resources that can be aggregated using resource brokers. for solving compute and data intensive applications. A resource can be a single processor or multi-processor with shared or distributed memory and managed by time or space shared schedulers. The processing nodes within a resource can be heterogeneous in terms of processing capability, configuration, and availability. The resource brokers use scheduling algorithms or policies for mapping jobs to resources to optimize system or user objectives depending on their goals.

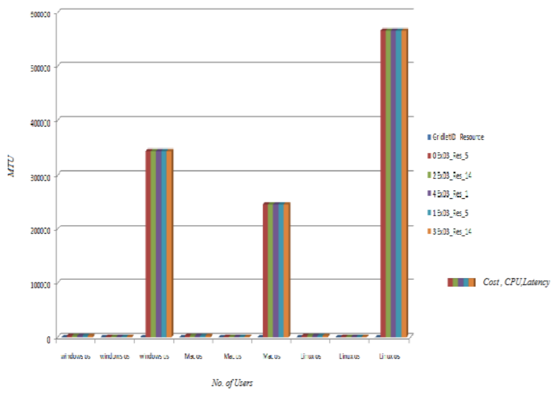

Figure 5. Time comparison diagram

VIII. CONCLUSION

To simulate a job scheduling strategy using this simulator, the user should sets related parameters of the simulator, adds the new scheduling strategy to the user broker, and then starts the simulator to carry on the performance analysis of the strategy. The GridSim simulator, added a new function of simulating the data grid job scheduling and proved its feasibility and rationality by experiment simulation. By monitoring the performance of data grid we need more resources to determine the user data. Here when comparing with the FCFS algorithm, the performance of random search algorithm is much better in determining the resource failure. Next step, some related problem on evaluating the fault tolerance scheduling strategies of the data grid will be researched.

IX. REFERENCES

[1] Cameron D, Millar P, Bell W, Capozza L, Stockinger K, Zini F.(2002) 'Simulation of dynamic Grid replication strategies in OptorSim', Proceedings of the 3rd International Workshop on Grid Computing (GRID), Baltimore, U.S.A. IEEE CS Press: Los Alamitos, CA, U.S.A., 18

[2] Kyong Hoon Kim, Rajkumar Buyya and Jong Kim.(2007), 'Power Aware Scheduling of Bag-of-Jobs Applications with Deadline Constraints on DVS-enabled Clusters', Proceedings of the 7th IEEE International Symposium on Cluster Computing and the Grid, IEEE CS Press, pp.14-17

[3] XING Chang-ming, LIU Fang-ai, CHEN Kun.(2009), 'A Job Scheduling Simulator in Data Grid Based on GridSim', Proceedings on IT in Medicine & Education.

[4] Chervenak A, Foster I, Kesselman C.(2001), 'The data grid towards architecture for the distributed management and analysis of large scientific data sets', Network and computer Applications, pp.187-200.

[5] P.K.Suri, Manpreet Singh.(2009), 'Resource Failure Impact on Job Execution in grid',International Conference on Advances in Recent Technologies in Communication and Computing.

[6] Rajkumar Buyya , Manzur Murshed (2002), 'GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing', concurrency and computation : practice and experience, 14, pp.1175-1220.

[7] Sulistio A, Buyya R. "A toolkit for modeling and simulating data Grids: an extension to GridSim", Concurrency and Computation: Practice and Experience 2008,20, pp.1591–1609.

[8] Srikumar Venugopal and Rajkumar Buyya, "An SCPbased Heuristic approach for Scheduling Distributed Data- Intensive Applications on

Global Grids", Journal of Parallel and Distributed Computing, Elsevier Press, Amsterdam,2008,68(4),pp. 471-487.

[9] Ahmed Ibrahim Saleh , Amany M. Sarhan ,Amr M. Hame," A New Grid Scheduler with Failure Recovery and Rescheduling Mechanisms: Discussion and Analysis", Springer Science+Business Media B.V. 2012

[10] Yu, Z., Shi, W.: An adaptive rescheduling strategy for grid workflow applications. In: Parallel and Distributed Processing Symposium. IPDPS 2007. IEEE International, pp. 1–18 (2007)

[11] Bharadwaj, V., Ghose, D., Mani, V., Robertazzi, T.: Scheduling Divisible Loads in Parallel and Distributed Systems. IEEE Comput. Soc. Press (1996)

[12] Negri, A., Poggi, A., et al.: Dynamic grid tasks composition and distribution through agents. Concurr. Comput. Pract. Exp. 18(8), 875–885 (2006)

[13] Silberschatz, A., Galvin, P., Gagne, G.: Process Scheduling. Operating System Concepts, 8th edn., p. 194. Wiley, Asia. ISBN 978–0–47023399–3. 5.3.4 Round Robin Scheduling (2010)