International Scientific Journal of Contemporary Research in Engineering, Science and Management (ISJCRESM)
Vol. 1, Issue 1, March-May 2016. ISSN : 2456–1134

Grid Computing for Effective performance of Job Scheduling

[1]B. Kalaiselvi*, Assistant Professor, Mahendra Engineering College for Women Tamilnadu, India
[2]Dr. M. Thangamani, Assistant Professor, Kongu Engineering College, Tamilnadu, India

ABSTRACT

Grid computing is becoming the most important research area in the high performance computing. Under this concept, the job scheduling in Grid computing has more complicated problems. However, the major problem is the optimal job scheduling in which Grid nodes need to allocate the appropriate resources for each job. In this paper, jobs are classified based on its "Job Type". The Job types are "Data intensive" i.e. the jobs which require more data access power and "Computational intensive". The jobs which require more CPU power respectively. The resources are classified based on the CPU speed, baud rate and failure rate. The failure rate is calculated based on the number of successfully executed jobs and number of failures while executing a job in the resource. The computationally intensive jobs are scheduled to the resources that have more CPU speed and less failure rate. Similarly the data intensive jobs are scheduled to the resources that have high baud rate and less failure rate. The simulation toolkit "Gridsim" is used for creating resources, jobs and for scheduling the jobs to the resources. This paper has shown that the scheduling algorithm will allocate the jobs more efficiently. Thus it reduces makespan and increases the throughput.

Keywords: Grid Computing, Gridsim, Scheduling Algorithm, CPU speed

1. INTRODUCTION

In the last decade, even though the computational capability and network performance have gone to a great extent, there are still problems in the fields of science, engineering, and business, which cannot be effectively dealt with using the current generation of supercomputers. The emergence of the Internet as well as the availability of powerful computers and high-speed network technologies as low-cost commodity components is rapidly changing the computing landscape and society. These technology opportunities have led to the possibility of using wide-

area distributed computers for solving large-scale problems, leading to what is popularly known as Grid computing.

Grids enable the sharing, selection and aggregation of a wide variety of resources including super computers, storage systems, data sources and specialized devices that are geographically distributed. Grid computing, most simply stated, is distributed computing taken to the next evolutionary level. The grid can be thought of as a distributed system with non-interactive workloads that involve a large number of files. Grid size can vary by a considerable amount. Grids are a form of distributed computing composed of many networked loosely coupled computers acting together to perform very large tasks.

In general, the grid scheduling is divided into 3 phases. They are Resource Discovery, Scheduling, and Execution. In the first phase it finds the available resources along with its characteristics. In the phase two it finds the best matching resources for the submitting jobs. The third phase is to execute a job in Gridsim. The computational grid has a dynamic and unpredictable behavior. They are

- Computational performance of each resource varies from time to time
- The Network connection may be unreliable
- The resources may join or leave the grid at any time
- The resource may be unavailable without a notification

The execution time of a job on different machine will have a different range of completion time. So scheduling the grid resources to minimize the job execution time is a major issue. The two different goals for task scheduling are high throughput and high performance computing. High throughput is minimizing the execution time of each application. High performance scheduling is scheduling a set of independent tasks to increase the processing capacity of the systems over a long period of time. This approach is to develop a high throughput computing scheduling algorithm. Ruay-Shiung Chang et al. [1-3] had explained the basics of scheduling. The task-scheduling algorithm should work better in the grid environment, in which the structure of the grid is static. It should reduce the total execution time of jobs submitted to the grid by effectively scheduling the jobs on to the appropriate resources in the grid. The execution time of the jobs is the main criteria for scheduling the jobs on to the resources.

1.1 Gridsim Toolkit

The GridSim toolkit provides a comprehensive facility for simulation of different classes of heterogeneous resources, users, applications, resource brokers, and schedulers. Rajkumar Buyya et al. [4] had illustrated about the Gridsim toolkit. It can be used to simulate application schedulers for single or multiple administrative domains distributed computing systems such as clusters and Grids. Application schedulers in the Grid environment, called resource brokers, perform resource discovery, selection, and aggregation of a diverse set of distributed resources for an individual user.

A multi-layer architecture and abstraction for the development of GridSim platform. 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. The third layer is concerned with modeling and simulation of core Grid entities such as resources, information services, and so on; 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 . The final layer is focused on application and resource modeling with different scenarios

SimJava Discrete Event Model **:** SimJava is a general purpose discrete event simulation package implemented in Java. Simulations in SimJava contain a number of entities, each of which runs in parallel in its own thread. An entity's behavior is encoded in Java using its body() method.

GridSim Entities: GridSim supports entities for simulation of single processor and multiprocessor, heterogeneous resources that can be configured as time- or space-shared systems. It allows setting of the clock to different time zones to simulate geographic distribution of resources. It supports entities that simulate networks used for communication among resources. During simulation, GridSim creates a number of multi-threaded entities, each of which runs in parallel in its own thread. An entity's behavior needs to be simulated within its body () method, as dictated by SimJava. Each instance of the User entity represents a Grid user. Each user is connected to an instance of the Broker entity. Every job of a user is first submitted to its broker and the broker then schedules the parametric tasks according to the user's scheduling policy. Before scheduling the tasks, the broker dynamically gets a list of available resources from the global directory entity. Every broker tries to optimize the policy of its user and therefore, brokers are expected to face extreme competition while gaining access to resources. The scheduling algorithms used by the brokers must be highly adaptable to the market's supply and demand situation.Each instance of the resource entity represents a Grid resource. Each resource may differ from the rest of the resources with respect to the following characteristics:

- Number of processors
- Cost of processing
- Speed of processing
- Internal process scheduling policy, e.g. time-shared
- Time zone

The resource speed and the job execution time can be defined in terms of the ratings of standard benchmark Millions of Instructions Per Second (MIPS). They can also be defined with respect to the standard machine. Upon obtaining the resource contact details from the Grid information service, brokers can query resources directly for their static and dynamic properties.

Provides resource registration services and keeps track of a list of resources available in the Grid. The brokers can query this for resource contact, configuration and status information. The flow of information among the GridSim entities happens via their Input and Output entities. Every networked GridSim entity has I/O channels or ports, which are used for establishing a link between the entity and its own Input and Output entities. Note that the GridSim entity and its Input and Output entities are threaded entities, i.e. they have their own execution thread with body() method that handles events. The use of separate entities for input and output enables a networked entity to model full duplex and multi-user parallel communications.

1.2 Method for Simulating Application Scheduling

In this section present high-level steps, to demonstrate how GridSim can be used to simulate a Grid environment to analyze scheduling algorithms. First step is to create Grid resources of different capabilities and configurations (a single or multiprocessor with time/space-shared resource manager). It also needs to create users with different requirements (application and quality of service requirements). The second step is to model applications by creating a number of Gridlets and define all parameters associated with jobs. The Gridlets need to be grouped together depending on the application model. Then, create a GridSim user entity that creates and interacts with the resource broker scheduling entity to coordinate execution experiment. It can also directly interact with Grid Information Service (GIS) and resource entities for Grid information and submitting or receiving processed Gridlets. However, for modularity sake, it need to encourage the implementation of a separate resource broker entity by extending the GridSim class. Finally, implement a resource broker entity that performs application scheduling on Grid resources. First, it accesses the GIS, and then inquires for resource capability including cost. Depending on the processing requirements, it develops a schedule for assigning Gridlets to resources and coordinates the execution. The scheduling policies can be systems-centric or user-centric.

2.3 Motivation for Use of Grid Simulation Tools

All of them including Application Developer, Tool Developer or Grid Developer need to test and verify their applications, tools & services respectively for fulfillment of intended design goals before the end product or logic is put in Real-Time Grid Computing Environment. The motivation for using simulation instead of directly using the Grid test-bed, especially in analyzing models and algorithms can be drawn from the following reasons. Setting up a Grid test-bed is expensive, resource intensive, and time consuming. Even if one is set up, it is mostly limited to some local area environments. Using a real test-bed with real jobs is time consuming as well. Hours of real job time can be simulated in seconds provided the simulation is having sufficient processor power. The real test-bed does not provide a repeatable and controllable environment for experimentation and evaluation of scheduling strategies. Simulation works well, without making the analysis mechanism unnecessary complex, by avoiding the overhead of co-ordination of real resources. Simulation is also effective in working with very large hypothetical

problems that would otherwise require involvement of a large number of active users, which is very hard to coordinate and build at a large-scale research environment for investigation purposes. Simulation allows analyzing existing as well as new economic models and scheduling algorithm

3. PROBLEM DEFINITION

In the existing system, the Job Type approach is used which is a scheduling algorithm that selects the best resource for the submitting jobs and minimize the job execution time. Each job can be categorized into two types namely, data intensive and computational-intensive in a specific ratio. The computational jobs are the jobs that require more processing time. However, data intensive jobs are the jobs that require more data access time. The job execution time and the data access time for each job is monitored and computed to provide the Job Ratio for the new job. The Job Ratio determines the exact job percentage requirement of either computation or data-intensive jobs. Each Job has its own ratio. Based on the job ratio, the jobs are scheduled and submitted to the site which provides a solution stating that the job's turnaround time is reduced but it is not optimal.

4. EXISTING SYSTEM

As known, the general problem of global resource scheduling is NP-complete. A large number of algorithms have been designed to schedule jobs to computational grid nodes. Fangpeng Dong et al. [5] had discussed the existing system in the literature. Several commonly used typical algorithms are listed as follows.

UDA- User-Directed Assignment (UDA) assigns each job, in arbitrary order, to the node with the best-expected execution time for the job.

OLB -Opportunistic Load Balancing (OLB) assigns each job, in arbitrary order, to the next available node.

Fast Greedy-Fast Greedy assigns each job, in arbitrary order, to the node with the minimum completion time for that job.

Greedy Heuristic-The Greedy heuristic is literally a combination of the Min-min and Max-min heuristics by using the better solution.

Min-Min-In Min-min, the minimum completion time for each job is computed with respect to all nodes. The job with the overall minimum completion time is selected and assigned to the corresponding node. The newly mapped job is removed and the process is repeated until all jobs are mapped.

Max-Min-The Max-min heuristic is very similar to the Min-min algorithm. The set of minimum completion time is calculated for every job. The job with overall maximum completion time from the set is selected and assigned to the corresponding node.
Genetic Algorithm-The Genetic algorithm is used for searching large solution space. It operates on a population of chromosomes for a given problem. The initial population is generated randomly. A chromosome could be generated by any other heuristic algorithm. When it is generated by Min-min, it is called "seeding" the population with Min-min.
Simulated Annealing-Simulated Annealing is an iterative technique that considers only one possible solution for each meta-task at a time. It uses a procedure that probabilistically allows solution to be accepted. These attempts to obtain a better search of the solution space based on a system temperature.

4.1 Drawbacks in Existing System

Though the above mentioned algorithms have various advantages, they also have some drawbacks.

- The drawback of UDA is that too many jobs are assigned to a single grid node. This leads to overloading and the response time of the job is not assured
- OLB has a drawback that load balance is not achieved leading to hard calculation of minimum completion time for a job
- Other algorithms including Genetic Algorithm, Simulated Annealing and GSA are very difficult to implement

5. PROPOSED SYSTEM

As the scheduling is performed statically, all necessary information about the resources and tasks are assumed to be available. Essentially the execution time of each individual job on the processor must be known and this information is stored in an Execution Time Calculation (ETC) matrix based on fault occurrence rate, Job type and CPU speed. Asef AL-Khateeb et al. [6] had dealt with this type of approach. A row in an ETC matrix contains the execution time for a single job on each of the available processors. With this ETC matrix the best job for each resource is compared and assigned.

Benefits Of Proposed Algorithm

- It selects best resource for the job
- System throughput is increased
- Makespan is reduced

6. EXPERIMENTS RESULT

Proposed system is measured my job ratio and calculation of execution time.

Job Ratio

As the scheduling is performed statically all necessary information about the resources and tasks are assumed to be available. For the given number of tasks and the resources, the job ratio, the expected execution time and turnaround time is calculated and the information are stored in a database. For each resource, the resource characteristics like speed, success rate, failure rate, cost and bandwidth are assumed to be available. Asef AL-Khateeb et al [6] used this ratio to schedule the job. According to the Job Type approach, the expected Job Execution Time (JET) is computed by the following formula

$$JET = JTT - JIOT \quad (1)$$

Where, JTT is the Job Turnaround time, which is the period of time when the job is submitted into the site until the job finishes the execution and JIOT is Job Input and Output Time, which is the total time required for I/O operations where the job is being executed.Likewise the Job Input and Output Time (JIOT) is also measured by the following equation.

$$JIOT = JTT - JET \quad (2)$$

Accordingly, the job ratio has the form Computational Job Ratio (CJR) : Data Intensive Job Ratio (DIJR) is computed as follow

$$JIOT = \left(\frac{JET}{JTT}\right) * 100 \quad (3)$$

$$\text{DIJR} = \left(\frac{\text{JIOT}}{\text{JTT}}\right) * 100 \qquad (4)$$

CJR and DIJR are multiplied by 100% for clarity and to deal with CJR and DIJR as percentage ratios. Therefore, the Job Ratio (CJR: DIJR) are the values that reflects the exact job required time for execution and the required time for data access. Such that CJR represents the execution time required by a job and DIJR represents the I/O operations time required by a job

Calculation of Execution Time

The Execution Time Calculation Matrix ETC_{ij} of task t_i on machine $m_{j,}$ is defined as the amount of time taken by $m_{j,}$ to execute t_i . Given that, m_j has no load t_i when assigned.

The expected Completion Time is (CT_{ij})

$$CTij = Bi + ETij \qquad (5)$$

Where, B_i = Beginning time of t_i on machine $m_{j..}$ The result of this research is to schedule the jobs to best resource and minimize the execution time. The resource characteristics like speed, cost and bandwidth and the job ratio are considered for calculating the expected ETC Matrix. Along with these parameters the failure rate of the resources should be considered for scheduling the job to the resources in the ETC Matrix. For the job having the highest CJR, it is scheduled to the resource having the high speed, cost and minimum bandwidth, as it requires more processing time. In the same way, the job having the highest DJR is scheduled to the resource hiving the high bandwidth, as it requires more data accessing time.

This is scheduling the jobs by considering only the job type. Along with this the failure rate of resources is considered. Leyli Mohammad Khanli et al. [7] had demonstrated the resource fault occurrence approach. Yi-Syuan et. al [8] produced a genetic algorithm for solving the problem of task scheduling. In this case a new initialization strategy to generate the first population and new genetic operators based on task and processor assignments to preserve the good characteristics of the found solutions. Effective scheduling for computational grid [9] planned by discrete particle swarm optimization. For the resource having the highest failure rate, the number of jobs scheduled to that resource should be minimum and the total completion time should also be minimum. The result will be in the following format (Task, machine, starting time, ending time).

The first step is to collect all the necessary information about the jobs (n) and resources (m) of the system in the database. The second step is to set the resource characteristics calculate job ratio. The next step is to generate the expected execution time calculation matrix expected ETC_{ij} , where for each task t_i on machine m_j, is defined as the amount of time taken by m_j, to execute t_i.Finally, the jobs are scheduled by considering the parameters like resource's speed, cost, bandwidth, failure rate and job ratio also considering the total completion time to be minimum.

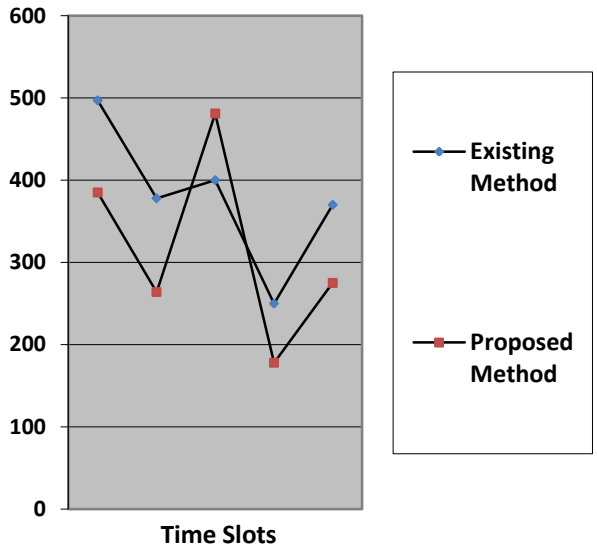

Fig.1 Execution time calculation for proposed system at different Time Slots

7. CONCLUSION AND FUTURE DIRECTION

Since the structure of the grid dynamically changes, there is no particular scheduling algorithm which can effectively utilize all the resources in a grid. But, the algorithm should be distributable, scalable and fault tolerant. In Grid environments, task execution failures can occur for various reasons. In this research, failure rate of the resource is considered as the main parameter for scheduling the jobs to the resources. Thus it reduces the selecting chance of the resources which have more failure probability. In addition to failure rate, resource's CPU speed and baud rate are considered and the jobs were successfully scheduled to the resources. This increases the performance of scheduling. Thus the simulation results indicate that the proposed strategy decreases makespan and increases the throughput. In this research job type and failure rate of the resources are considered as the main criteria but future research can be done using many factors like disk space, memory and so on. Instead of submitting the jobs one by one to the resources, submit the jobs as batches. Another research direction is to create different scheduling algorithms for problem arising in grid computing.

References

1. Ruay-Shiung Chang, Chun-Fu Lin and Shih-Chun Hsi, " Accessing Data from Many Servers Simultaneously and Adaptively in Data Grids," Future Generation Computer Systems, Vol. 26, Issue 1, pp. 63-71, Jan. 2010,
2. Ruay-Shiung Chang and Min-Shuo Hu, "A Resource Discovery Tree Using Bitmap for Grids," Vol. 26, Issue 1, Future Generation Computer Systems, Jan. 2010, pp. 29-37.
3. Ruay-Shiung Chang, D.J. Deng, and CH Ke, "Internet Resource Sharing and Discovery Preface," Journal of Internet Technology, Vol. 11, Issue 2, March, 2010.
4. Rajkumar Buyya, Economic-based Distributed Resource Management and Scheduling for Grid Computing, Ph.D. Thesis, Monash University, Melbourne, Australia, April 12, 2002.
5. Fangpeng Dong and Selim G. Akl , Scheduling Algorithms for Grid Computing: State of the Art and Open Problems, pp.1-55, 2006.
6. Asef AL-Khateeb, Rosni Abdullah and Nur'Aini Abdul Rashid , Job Type Approach for Deciding Job Scheduling in Grid Computing Systems, Journal of Computer Science, Vol. 5, No.10, pp. 745-750, 2009.
7. Leyli Mohammad khanli, Motahareh Ahmadi, A New Algorithm for Scheduling Parallel Tasks in Utility Grids using Reserve Resources for Tasks with Deadline, International Journal of Computer Science and Information Technologies, Vol. 2 , No. 3, pp .1300-1304, 2011.
8. Yi-Syuan Jiang, Yi-Syuan, Wei-Mei Chen, The Journal of Supercomputing, Vol.71, No. 4, pp. 1357-1377, 2015
9. M. Christobel, S. Tamil Selvi, and Shajulin Benedict, Efficient Scheduling of Scientific Workflows with Energy Reduction Using Novel Discrete Particle Swarm Optimization and Dynamic Voltage Scaling for Computational Grids, Scientific World Journal, 2015

Authors Biography

Ms. B. Kalaiselvi has completed Master of Engineering in Computer Science and Engineering in Anna University Application. Her research expertise covers Medical data mining, machine learning, cloud computing, big data, fuzzy, soft computing and ontology. She has presented 10 papers in national and international conferences in the above fields. She is currently working as Assistant Professor in Mahendra Engineering College for Women.

Dr. M. Thangamani possesses nearly 20 years of experience in research, teaching, consulting and practical application development to solve real-world business problems using analytics. Her research expertise covers Medical data mining, machine learning, cloud computing, big data, fuzzy, soft computing, ontology development, web services and open source software. She has published nearly 70 articles in refereed and indexed journals, books and book chapters and presented over 67 papers in national and international conferences in the above field. She has delivered more than 60 Guest Lectures in reputed engineering colleges on various topics. She has got best paper awards from various education related social activities in India and Abroad.

She has organized many self-supporting and government sponsored national conference and Workshop in the fields of data mining, big data and cloud computing. She continues to actively serve the academic and research communities. She is on the editorial board and reviewing committee of leading research journals, which includes her nomination as the Associate Editor to International Journal of Entrepreneurship and Small & Medium Enterprises at Nepal and on the program committee of top international data mining and soft computing conferences in various countries. She is also seasonal reviewer in IEEE Transaction on Fuzzy System, international journal of advances in Fuzzy System and Applied mathematics and information journals. She has been nominated as chair and keynote speaker in international conferences in India and countries like Malaysia, Thailand and China. She has Life Membership in ISTE, Member in CSI, International Association of Engineers and Computer Scientists in China, IAENG, IRES, Athens Institute for Education and Research and Life member in Analytical Society of India. She is currently working as Assistant Professor at Kongu Engineering College at Perundurai, Erode District.