Research Article Impact Factor: 0.621 ISSN: 2319-507X
Shrikant Rathod, IJPRET, 2014; Volume 2 (8): 519-528 IJPRET

INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

A SURVEY OF SCHEDULING ASPECT IN GRID COMPUTING

MR. SHRIKANT DHARMA RATHOD[1], PROF. S. R. JADHAV[2]

1. ME First Year, Dept. of Computer Science & Engg, BabaSaheb Naik College of Engineering, Pusad.
2. Dept. of Computer Science & Engg, BabaSaheb Naik College of Engineering, Pusad.

Accepted Date: 15/02/2014 ; Published Date: 01/04/2014

Abstract: When the trend in human culture advances, the problems in their science and engineering is increased to solve those problems lot of computing power is needed. Grid is a heterogeneous system that allows sharing of resources. Grid computing is a technology that works what super computer does. Efficient utilization of the grid environment requires large amount of computing power. Scheduling independent jobs to the resources is not an easy task. Scheduling schedules maximum number of jobs to the minimum amount of resources which is a very tedious task. Many scheduling algorithms exist to focus either on the job side or on the resource side. Schedulers either target system-centric metrics, such as utilization and throughput, or prioritize jobs based on utility metrics provided by the users. In the proposed work, to utilize the power of grid completely, both the job and resources are taken into account. Jobs are prioritized based on the common location sum that considers both user and system priority Resource management and job scheduling are two important and difficult tasks in grid computing. Scheduling and execution of jobs in a dynamic environment like Grid often calls for efficient algorithms to schedule the resources required for successful execution of the jobs.

Keywords: Grid computing, Meta-scheduling, job scheduling, adaptive scheduling.

PAPER-QR CODE

Corresponding Author: PROF. S. S. ASOLE

Access Online On:

www.ijpret.com

How to Cite This Article:

Shrikant Rathod, IJPRET, 2014; Volume 2 (8): 519-528

INTRODUCTION

Grid computing is a technology that couples disparate and distribute heterogeneous software and hardware resources to provide uniform computing environment to solve data and computational-intensive problems. It enables the sharing to resources to solve a problem that work towards our common goal. The resources of many computers in a network can be applied to solve a single problem at the same time. the process that needs large number of computer processing cycles can use.[5]

This technology to solve their problem. Grid computing can be a form of heterogeneous, distributed and large-scale cluster computing. It needs to schedule the process to the appropriate computation resources So that it can divide and form the pieces of a program to as many thousand computers as possible in virtual organization The different type of computer resources can be applied to different type of the problem. Grid computing can experience difference in resource utility because, they can be owned by an independent organization but they are utilized by another set of organization. It provides the good infrastructure for the user communities to access and to solve their specific problems with the theme of obtaining the objective. [2]

Grid computing works towards the three common objectives:

(1) More cost-effective with the given amount of computer resources,

(2)As a way to solve problems with less Amount of computing power,

(3)Whole resources are working towards our common objective. Not only with these Objective it can also focus towards the standards in certain areas like scheduling, resource discovery, load balancing, security, system management and so on. Because of the heterogeneity nature of the grid, the global grid forum has been organized as a working body for designing standards for the grid. [13]

1.1. Scheduling:

Grid scheduling and resource management play a critical role in building an effective and efficient grid environment. There are three different paradigms that are centralized distributed and hierarchical helps to perform the scheduling.

Fig.1. Centralized Scheduling [7]

In centralized scheduling, a central machine that acts as are source manager to schedule jobs to all the surrounding nodes that are a part of the grid environment. It is used in situations like a computing center where resources that have the similar characteristics and usage policies. Jobs are first submitted to the central scheduler who then dispatches the jobs to the appropriate nodes those jobs that cannot be started on a node are normally stored in a central job queue. In distributed computing, multiple localized schedulers interact with each other in order to dispatch jobs to the participating nodes. There are two mechanism for a scheduler to communicate with other schedulers are direct and indirect communication. It overcomes the scalability problems which are incurred in the centralized paradigm in addition it can offer better fault tolerance and reliability. However the lack of a global scheduler which has the necessary information on available resources, usually leads to suboptimal scheduling decisions. In hierarchical scheduling, a centralized scheduler interacts with local schedulers for job submission. The centralized scheduler is a kind of a meta-scheduler that dispatches the submitted jobs to local schedulers. Similar to the centralized scheduling, hierarchical scheduling can have scalability and communication bottlenecks. However, compared with the centralized scheduling, one advantage of hierarchical scheduling is that the global scheduler and local scheduler can have different policies in scheduling jobs [7]

2. Scheduling works:

Scheduling can work under four main stages. They include

- Resource discovery
- Resource selection
- Schedule generation
- Job execution [7]

2.1. Resource discovery:

The goal of resource discovery is to identify a list of authenticated resources that are available for job submission .In order to cope with the dynamic nature of the grid, a scheduler needs to have some way of incorporating dynamic state information about the available resources into decision making process. Grid environment mostly uses the pull, push or pull –push model for resource discovery.[7]

2.2. Resource selection:

Once the targeted resource is known, the second phase is to select the resources that best suit the user constraints and conditions imposed by the user, such as CPU usage, RAM available or disk storage. The result of resource selection is to identify a resource list in which all resources can meet the minimum requirements for a submitted job or a job list.[6]

2.3. Schedule generation:

The generation of steps involves two steps, selecting jobs and producing resource selection strategies .The goal of job selection is to select a job from a job queue for execution.[5]

2.4. Job execution:

Once a job and resource are selected, the next job is to submit the job to the resource for execution. Job execution maybe as easy as running a single command or as complicated as running a series of scripts that may include staging.[7]

3. RELATED WORK

Most of the scheduling system considers two main issues called data access cost and the job waiting in queue. Several improvements had been taken place to improve their overall performance. the job type to achieve the minimum turnaround time and load balancing and slowly to the user priority. The data access cost of the job is combined with the waiting time of the job in queue to reflect the minimization of the overall time taken to complete the given job[8].

The methods and software for a Grid resource manager. They are responsible for resource brokering and scheduling in early production Grids. In the proposed system, they try to match the job with the appropriate resources to reduce the job completion time by using the prediction method .The matching of the job to the appropriate resources can be done by the decentralized brokering system.[9]

The existing scheduling algorithm of the data intensive jobs based on the data access cost however, the influencing factors on access cost should be reconsidered, since existing algorithms neglect the importance of potential behaviors of jobs in the waiting queue. The potential behaviors can be predicted roughly based on the queue length of waiting jobs and the local replica replacement strategy. So to verify the potential behaviors of jobs in the waiting queue have an effect on the evaluation of access cost. In the proposed system, a scheduling algorithm based on access cost with potential behaviors are taken and given a decentralized feedback mechanism to replica report. [10]

An efficient job scheduling algorithm to execute jobs in less time and to maintain the grid load based on the minimum execution time this is used to fetch the job from the queue. The main usage and consideration in this algorithm is the job waiting in the queue and their corresponding access cost [11]

The static scheduling algorithms Jobs are queued and collected into a set when they arrive in the batch mode. The available number of queue can be calculated by using difference between the total queue length and the already available jobs the queue. [12]

Scheduling system that considers job type and priority and the resource selection based on their processing power, job queue access cost, data access cost. The overall performance can be improved up to the 48 percent, when compared to the previous existing system. An Adaptive Scoring Job scheduling algorithm, they have used to calculate the average transmission power and average computing power based on the CPU available, CPU speed and Load. Jobs are submitted to the best cluster based on the average transmission power and the best resources are found out by using the average computing power. [5]

4. Characteristics of a Computational Grid

There are many desirable properties and features that are required by a grid to provide users with a computing environment. They are as follows:

- Heterogeneity

The grid involves a number of resources that are varied in nature and can encompass a large geographical distance through various domains.

- Scalability

The grid should be tolerant to handle a large number of nodes without any performance degradation.

- Adaptability or Fault Tolerant

In a grid unexpected computational aborts, hardware or software faults etc are high. These faults are generally handled by Resource Managers.

- Security

All the user participating computers should be protected from any malicious manipulations or interventions.

5. Grid Scheduling Model

5.1 Performance Model

This section defines the performance model of tasks and processors. The length of a task represents its workload and is decided by the number of instructions in the task. The computational power provided by a processor depends on the usage of the original user. If the user occupies more resources, the computer will supply less computing power to the server. The usage of a processor can be defined as *up,t*.

The processor and *t* is the time. Besides, a grid is heterogeneous, so processors have various capabilities by nature. According to the utilization and the capability of a computer, the capability of a processor can be calculated. This capability value shows the workload completed by a processor per unit time. The capability of processors and the length of tasks must have the same standard to model their performance, and this criterion is the number of instructions. In a word, the capability of each processor is the excess computational power of the processor which is not used by his capability of processor p during time interval (*t, t+1)*. If the load by the original users is very heavy or the computer is powered off, may be zero. [2]

5.2 Criteria of a Schedule:

Two major criteria, make span and total processor power cost, are used to evaluate the performance of the proposed scheduling system. [4]

5.3 Scheduling Model:

In this section, a dynamic scheduling algorithm that modifies Round-robin algorithm is illustrated. In the grid, requests are dynamically submitted to the agent, and new tasks are continuously sent to the task queue during the scheduling process. Besides, the number of nodes that provide computing power also varies overtime. The proposed algorithm considers these dynamic characteristics and makes the scheduling adaptive for the grid. The proposed scheduling algorithm uses two queues to manage all tasks. One is the waiting queue, and the other is the execution queue. The waiting queue manages the tasks that are not yet allocated to processors. When the algorithm assigns a task to computing node, this task is removed from the waiting queue and moved to the execution queue. Each task in the execution queue has a specific order, and no task has the same order as any other task. The execution queue has a pointer which is used to point a task in the queue. Using this pointer, tasks in the execution queue can be scanned in the ringed round-robin fashion. Initially, this pointer is pointed to the task with the lowest order in the execution queue. The task pointed by the pointer is called the current point, and the next point and the last point can be defined as follows. If the current point is the task with the highest order in the execution queue, the next point will be the task with the lowest order in the queue. Otherwise, the next point will be the task with the minimum order of the tasks with higher order than the current point. If the current point is the task with the lowest order in the execution queue, the last point will be the task with the highest order in the queue. Otherwise, the last point will be the task with the maximum order of the tasks with lower order than the current point. When a task is moved to the execution queue, it is inserted before the pointer. If the task pointed by the pointer is removed, the pointer will advance to the next point. The state diagram of the proposed algorithm the scheduling process starts at S1 to wait for tasks' arrival. when tasks are submitted to the scheduler, the process advances on S2. Each processor is assigned one task of the waiting queue respectively. The tasks that have been assigned are removed from the waiting queue and moved to the execution queue. Each task of the execution queue has to maintain a processor list. This list records the processors that are dedicated to this task. when a processor completes a task, the server receives the result, this task is removed from the execution queue, and the scheduler assigns another unassigned task of the waiting queue to this free processor. If new tasks are sent to the scheduler, they will be queued in the waiting queue. These processes are repeated until all tasks in the waiting queue are assigned. Then the scheduling process goes to S3.

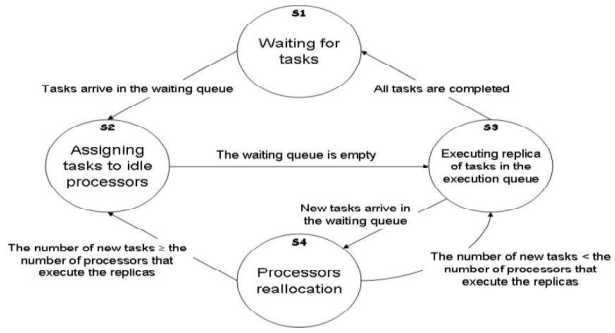

Figure 2. The state diagram of the proposed scheduling algorithm [2]

The state diagram of the proposed scheduling algorithm In S3, the waiting queue is empty, and several tasks in the execution queue remain uncompleted. The pointer points to the task with the lowest order. When a task is completed, this task is removed from the execution queue, and the processor recorded in the processor list is released. The free processor is assigned to execute the replica of the current point, and the processor tag is recorded in the processor list of the task. Then, the point moves to the next point. Whether the original or the replica of a task is finished, the execution on all processors should be killed. When the task with replicas is completed, the server receives the result, this task is removed from the execution queue, and all processors dedicated to this task are released. The free processors are assigned in the ringed round-robin fashion to execute the replicas of tasks in the execution queue. While all tasks are completed, the scheduling process returns to S1 [2]

5.4 Scheduling System:

The scheduling system schedules all tasks of the queue and performs two scheduling processes at the same time. Max-min scheduling process receives prediction information of task lengths and processor capabilities from the prediction system. Then the earliest completion time of each task can be calculated, and [4]

Max-min gives the highest priority to the task with the maximum earliest completion time. On the other side, the scheduling process of the proposed algorithm doesn't need any prediction information and gives priorities just like Round-robin. The adaptive scheduling model selects the proper scheduling strategy according to the accuracy of the prediction system. [3]

If the prediction accuracy is low or even the prediction system doesn't work, the scheduling strategy of the proposed algorithm will be adopted Else, the proposed model will adopt the scheduling strategy of Max-min algorithm. Both scheduling processes assign replicas of tasks to

idle nodes and consider dynamic characteristics of grid tasks and processors. The agent assigns tasks to idle processors according to the strategy. [2]

6. CONCLUSION

Grid scheduling and resource management play a critical role in building an effective and efficient grid environment .Grid computing is one of the major key concerns to develop the computing system that have the major ability to self configuration and optimization. Numerous algorithms in the literature has been studied and focused on scheduling aspect.

7. REFERENCES

1. Guiyi Wei, Yun Ling, Athanasios V. Vasilakos, Bin Xiao, Yao Zhen "PIVOT: an adaptive information discovery framework for computational grids," Information Sciences 180 (23)

2. Lee, Liang, and Hung-Yuan Chang "An Adaptive Task Scheduling System for Grid Computing "Proceedings of The Sixth IEEE International Conference on Computer and Information Technology "2009.

3. Kun-Ming Yu, Cheng-Kwan Chen "An Evolution-based Dynamic Scheduling Algorithm in Grid Computing Environment" Computer Science and Information Engineering Department, Chung-Hua University, HsinChu 2008.

4. Gong, Jiong, Hou Yong, Liu "User QoS and System Index Guided Task Scheduling in Grid Computing "2008

5. Yi-Lun Pan Yueh-Ching Lee Fan Wu 2009] "Job Scheduling of Savant for Grid Computing on RFID EPC Network"

6. Fufang Li, Deyu Qi, Limin Zhang, Xianguang Zhang, and Zhili Zhang "Research on Novel Dynamic Resource Management and Job Scheduling in Grid Computing"

7. "Hybrid Adaptive Meta-Scheduling System For Grid Computing

8. ManjotKaur Bhatia, S. K. Muttoo and M. P. S. Bhatia "Secure Requirement Prioritized Grid Scheduling Model "International Journal of Network Security, Vol.1 2013

9. Elmroth E, Tordsson J "Grid resource brokering algorithms enabling advance reservations and resource selection based on performance predictions," Future Generation computer System 24(6), 2008.

10. Jin H, Shi X, Qiang W, Zou D " An adaptive meta-scheduler for data intensive applications," International Journal Grid Utility Computing 1(1),2005.

11. M. Maheswaran, S. Ali, H.J. Siegel, D. Hensgen, R. Freund, "Dynamic matching and scheduling of a class of independent tasks on to heterogeneous computing system," Journal of Parallel and Distributed Computing 59:107–131,1999

12. Al-khateeb A, Abdullah R, Rashid NA "Job type approach for deciding job scheduling in grid computing system" s. Journal Computer Science 5:745–2008

13. I. J. Computer Network and Information Security, 2014, 2, 16-22 Published Online January 2014 in MECS (http://www.mecs-press.org/) DOI: 10.5815/ijcnis.2014.02.03 Scheduling in Grid Systems using Ant Colony Algorithm