Dynamic Cluster Scoring Job Scheduling algorithm for grid computing

S. Dineshbabu[1], S. Supriya[2]

[1] PG Scholar, SNS College of Engineering, Coimbatore, TamilNadu, India.

[2] PG Scholar, SNS College of Engineering, Coimbatore, TamilNadu, India.

Abstract— **Grid is a parallel and distributed system, shares the resources among multiple administrative domains. Job scheduling is one of the major issues in the grid environment. Effective Job scheduling algorithm is proposed to overcome the scheduling problems so as to decrease the overall execution time and to achieve the efficient utilization of grid resources. Effective scheduling in grid environment can reduce the amount of data transferred between nodes by submitting a job to a node where most of the requested data files are available. In this paper we propose Cluster scoring job scheduling (CSJS) algorithm for the grid. It can decrease the overall makespan and increases the utilization of idle resources.**

Keywords—grid, job scheduling, makespan, CSJS

I. Introduction

Grid Computing is a processor architecture that combines computer resources from various domains to reach a main objective. In grid computing the computers on the network can work on a task together, thus functioning as a super computer. The aim of grid computing is use to perform sharing, selection and aggregation of resources distributed across multiple administrative domains.

Grid achieves the same level of computing power of super computer, but at a much reduced cost. In grid computing the computers on the network can work on a task together, thus functioning as a super computer. Grid is a heterogeneous system. Scheduling independent tasks on it is more complex one. In order to utilize the power of grid computing completely, we need an efficient job scheduling algorithm to assign jobs to resources. This paper focuses on the efficient job scheduling considering the makespan of job in a grid.

The purpose of job scheduling algorithm is to minimize the completion time and enhance the system throughput. The status of grid environment may change at any time, the traditional job scheduling algorithm, e.g."First Come First Serve"(FCFS) [1],"First Come Last Serve (FCLS), etc, may not adapt to the dynamic grid environment well. This paper proposes a new framework and scheduling algorithm to decrease jobs make span in a grid environment. Local update and global updates are used to get the newest status of resources in the grid. Based on these update results, we can schedule the jobs more dynamically and appropriately.

A new job scheduling algorithm is proposed and is called Cluster scoring job scheduling algorithm (CSJS), we compare SJSA with Ant colony Optimization (ACO), Most Fit Task First Scheduling algorithm (MFTF) and Dynamic Processor Largest Task First Scheduling algorithm (DPLTF) selection method in the experiments. According to the Results CSJS is capable of decreasing the completion time of jobs better than other job scheduling algorithms mentioned above.

II. RELEATED WORK

The scheduling algorithms used in grid environment are classified into two types: batch mode and on-line mode.

In batch mode Jobs are queued and collected into a set when they arrive .They will be scheduled by the scheduling algorithm. Batch mode algorithms are suitable for the environment with the same type of resources.

First-Come, First-Served Scheduling Algorithm (FCFS) the Jobs are executed based on the sequence of submitted jobs. The second job will be executed after the completion of first job, and FCFS [1] has a serious problem called convoy effect. The convoy effect will occur when there is a job with large workload in the Front of the job queue. All other small workload jobs have to wait until the larger one completes its execution.

Fastest Processor to Largest Task First Scheduling Algorithm (FPLTF) [2] performs the scheduling of jobs based on the workload of jobs and computing power of resources. In Min–min scheduling algorithm [3], each job will be always assigned to the resource with earliest completing time. The Max–min scheduling algorithm gives the highest priority to the job with the maximum earliest completion time.

The Work queue with replication (WQR) [4] scheduling algorithm uses the concept of replication with Work queue (WQ) scheduling algorithm. WQR scheduling algorithm, each task will be replicated into a predefined numbers of times and transfer to the available resources. If one resource completes the job, all other replications which are executed by other resources will be canceled. The performance of the WQR scheduling algorithm is good and it wastes extra computing power on executing the replications, if the size of the jobs is large the replication takes lots of time.

In online mode algorithms jobs are scheduled when it arrives. Online mode scheduling algorithms are more suitable for heterogeneous resources with dynamic environment.

Dynamic fastest processor largest task first scheduling algorithm (DFPLTF) [5] is the modified version of the fastest processor largest task first scheduling algorithm (FPLTF). Dynamic FPLTF has a good ability to adapt to the dynamicity and heterogeneity of the environment.

Most Fit Task First Scheduling algorithm (MFTF) [6] assigns the most suitable resource to the task with the help of value called fitness. The range of the fitness is between 100,000 and 0. It always assign jobs to resources with the largest fitness value and therefore, the load of resources with the fastest speed will become heavy and many jobs will be queued, this wastes time.

$$Fitness(i,j) = 100{,}000/(1+[\ Wi / Sj- Ei]\ ) \quad (1)$$

Where Wi indicates the workload of the job i, Sj indicate the CPU Speed of the resource j, Ei is the the expected execution time of the job i.

$$Ei=A+N*S \quad (2)$$

Where A is the average response time, N is the non-negative real number and S is the standard deviation of task response time. When the estimated execution time is closer to Ei, it means that the node is more suitable for the task.

Ant Colony Optimization (ACO) [7] used for solving the scheduling problems in grid. Xu et al. proposed simple grid simulation architecture and modified the basic ant algorithm for job scheduling in grid. The ACO scheduling algorithm needs some information such as the number of CPUs, Million Instructions per Second (MIPS) of every CPU for job scheduling. A resource must submit these information's to the resource monitor. The resource suitability calculated based on the pheromone value.

$$Pr_j=m\times P+C/S \quad (3)$$

Where Pr indicates the initial path between the resource j and the resource monitor indicates the number of CPU, C is the size of the parameter and S is the transfer time between resource j to the resource monitor. Encourage- factor, Punish-factor are important factors in ant algorithm, when resource j completes the job successfully then the Encourage-factor will be updated. If resource j does not complete the job successfully means then it update the Punish- factor. Encourage- factor and punish-factor are used to adjust the every resource pheromone value and select the suitable resource for the job. The main drawback of the ACO algorithm is load of the better resources is much more heavy compare with other resources.

III. THE CLUSTER SCORING JOB SCHEDULING ALGORITHM(CSJS)

The aim of the CSJS algorithm is decrease the job's execution time. Here we consider computing power of the resources and transmission power of each cluster's in the grid environment. The computing power of the resources indicates the CPU speed and available CPU percentage and the transmission power defined by the bandwidth between different clusters. CSJS uses the status of each resource in the grid as parameters to initialize the cluster score of each cluster. The cluster score of each cluster will be updated by using local update and global update. The system will submit a job to the most suitable resource based on the cluster scores.

A. Architecture

Grid architecture identifies fundamental system components, specifies the purpose and functions of these components, and indicates how these components interact.

The grid system framework composes of four main components, a Portal, a Job Scheduler, Grid Information Server (GIS) and Grid Resources (GR). The Portal acts as an interface to the users for job submission and shows the execution status of the jobs submitted. The Grids Information Server will store the information's about the status of grid resources.

Local resource manager (LRM) which is a component of cluster (Cluster Head) monitors the status of grid resources such as TCP/IP performance, available CPU percentage, and available memory and reports them back to the Grid Information Server. The Resource Broker splits the jobs into tasks and allocates the task to the most suitable resource with the help of resource information's available in grid information server. Once a job is completed, the time taken for executing each job will be obtained and stored in GIS.

B. CSJS algorithm

In CSJS, users can submit different types of jobs at the same time containing computing-intensive jobs or data-intensive jobs. The computing-intensive jobs needs lots of computing power to finish and data – intensive jobs needs lots of bandwidth to transmit files.

User gives following information's types of jobs, number of computing-intensive and data–intensive jobs. When the user completes the job specifications, the Portal will send the request to the Job Scheduler. Job Scheduler will assign the jobs to the resources with the highest cluster score.

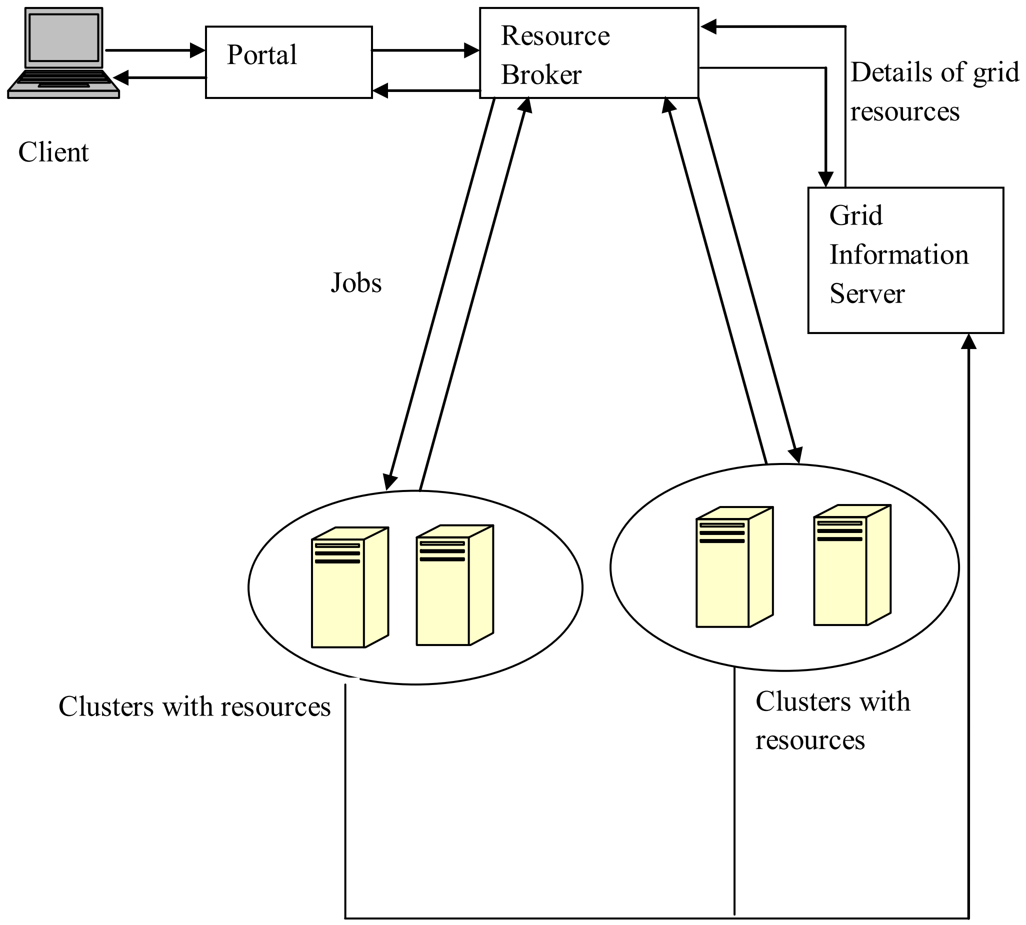


Fig 1. Grid Architecture

After Job Scheduler sends the job to the resource selected, it continues to schedule the next job until completing all jobs. After a resource receives a job, it starts to execute. The status of the resource will change, and therefore local update will be applied to adjust the cluster score of the cluster containing the resource. Once a job is completed by a resource, the result will be sent back and stored in the Information Server. At the same time, the global update will be applied to adjust the status of the entire grid system.

C. Cluster score

After identifying the job type, Job Scheduler will initialize the score of each cluster.

The cluster score [8] is defined below:

$$CSi = a.ATPi+b.ACPi \qquad (4)$$

Where CSi is the cluster score for cluster i, a and b are the weight value of ATPi and ACPi respectively, the sum of a and b is 1, ATPi and ACPi are the average transmission power and average computing power of cluster i respectively.

ATPi [1] means the average available bandwidth the cluster i can supply to the job and is defined as:

$$ATPi=\frac{\text{Bandwidth available } (i,j)}{m-1} \quad i\neq j \qquad (5)$$

Where Bandwidth available (i,j) is the available bandwidth between cluster i and cluster j, C is the number of clusters in the overall grid system. ACPi [1] means the average available CPU power cluster i can supply to the job and is defined as

$$ACPi=\frac{\text{CPU Speed}_k*(1\text{-load}_k)}{n} \qquad (6)$$

Where CPU Speed$_k$ is the CPU speed of resource k in cluster i, load$_k$ is the current load of the resource k in cluster i, n is the number of resources in cluster i. computing power calculated based on the following formula

$$CP_k = \text{CPU-speed}_k \times (1\text{-load}_k) \qquad (7)$$

CP_k is the available computing power of the resource k.

The bandwidth between resources in the same cluster is very large. Here we only consider the bandwidth between different clusters. The scale of the value of ATPi and ACPi are different. To normalize, we map them into a scale of 1–10 before using them to calculate the cluster score of each cluster. The scale of normalization is uniform one. Let Inc. = (maximum–minimum)/10. Then value between minimum and minimum + Inc. should be normalized to 1, value between minimum + Inc. to minimum + 2*Inc. should be normalized to 2, and so on.Local update and global update are used to update the cluster score. After a job is submitted to a resource, the status of the resource will change and local update will be performed to adjust the cluster score of the cluster containing the resource. What local update does is to get the available CPU percentage from Grid Information Server and recalculate the ACP, ATP and CS of the Cluster. After a job is completed by a resource, global update will get information of all resources in the entire grid and recalculate the ACP, ATP and CS of all clusters.

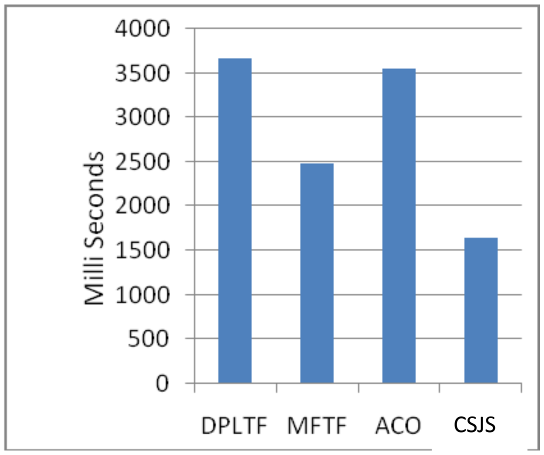
IV. IMPLEMEMTATION

We implement SJSA in GridSim Toolkit. The total number of jobs simulated is 500. In this experiment, we compare CSJS with ACO, MFTF, and DPLTF algorithms. There are two sections in this experiment. First we need to define the coefficient values of ACP (a), ATP (b). Second compare the performance of CSJS with ACO, MFTF, and DPLTF algorithms. The sum of the coefficient of ATP and the coefficient of ACP is equal to 1. Here we compare the total execution time when the ratio of a (coefficient of ACP) to b (coefficient of ATP) is 3:7.

A. Results

We compare the total execution time of CSJS with other scheduling algorithms. According the results CSJS uses less time to execute all the jobs because CSJS method assigns jobs to resources depending on the status of the resources. A cluster with the largest cluster score indicates that it is the fittest cluster for the job right now.

The cluster score of each cluster will be recalculated based on the newest status of resources included in it by local update and global update rules. Therefore, CSJS can select a better resource for each job and take less time to complete all jobs than other methods.



V. CONCLUSION

In this paper, we propose cluster scoring method to schedule jobs in grid environment. CSJS selects the suitable resource to execute the jobs according to the status of resources. Local and global update rules are used to know the status of each resource. The experimental results show that CSJS is capable of decreasing the overall execution time of jobs and the performance of CSJS is better than other methods.

REFERENCES

[1] Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, Operating System Concepts, eighth ed., John Wiley & Sons, 2011.

[2] Network Weather Service (NWS) website, <http://nws.cs.ucsb.edu/ewiki>.

[3] M. Maheswaran, S. Ali, H.J. Siegel, D. Hensgen, R. Freund, Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing system, Journal of Parallel and Distributed Computing 59 (1999) 107–131.

[4] Ruay-Shiung Chang,Chih-Yuan Lin,Chun-Fu Lin, An Adaptive Scoring Job Scheduling algorithm for grid computing,Elsevier Information Sciences 207(2012)79-89.

[5] Ruay-Shiung Chang, Jih-Sheng Chang, Po-Sheng Lin, Balanced job assignment based on ant algorithm for computing grids, in: Asia-Pacific Service Computing Conference, 11–14 December 2007, pp. 291–295.

[6] Sheng-De Wang, I-Tar Hsu, Zheng-Yi Huang, Dynamic scheduling methods for computational grid environment, International Conference on Parallel and Distributed Systems 1 (2005) 22–28.

[7] Zhihong Xu, Xiangdan Hou, Jizhou Sun, Ant algorithm-based task scheduling in grid computing, in: CanadianConference on Electrical and Computer Engineering, vol. 2, 4–7 May, 2003, pp. 1107–1110.