International Journal For Technological Research In Engineering
Volume 1, Issue 10, June-2014 *ISSN (Online): 2347 - 4718*

COARSE GRAINED JOB SCHEDULING WITH DYNAMIC STRATEGY IN GRID COMPUTING

Pankti Dharwa[1] (M. Tech Student), Harshvardhan Mathur[2] (Asst. Prof)
Sobhasaria Engineering College
Sikar, Rajasthan, India.

Abstract: Grid technologies are emerging as the next generation of distributed computing, allowing the aggregation of resources that are geographically distributed across different locations. To achieve the promising potentials of tremendous distributed resources, effective and efficient scheduling algorithms are fundamentally important. In grid computing main emphasis is given on job scheduling and resource management. . In this paper, scheduling algorithm is dynamically executed based on resources computational and communication capabilities. Coarse-grained jobs are grouped together based on the chosen resources characteristics, memory requirements by particular jobs and bandwidth to maximize resource utilization and minimize processing time and cost. Job scheduling is a decision process by which application components are assigned to available resources to optimize various performance metrics as per given strategy in this paper. The results show that the proposed scheduling algorithm efficiently utilizes resources at its best and reduces the processing time of jobs.

Keywords: Coarse grained job, Dynamic, Grid computing, Scheduling.

I. INTRODUCTION

The goal of grid computing is to create the illusion of a simple yet large and powerful self-managing virtual computer out of large collection of connected heterogeneous system sharing various combinations of resources. Grid computing is a way to enlist large no. of machines to work on multipart computational problem such as circuit analysis, mechanical design, special effects or any big project [1]. It harnesses a diverse number of machines and other resources to rapid processes to solve problem beyond an organization's available capacity. Once a proper infrastructure established by resource brokers, a user will have access to a virtual computer that is reliable and adaptable to the users [2] . For this, there must be standard for grid computing that will allow a secure and robust infrastructure to be built. Standards such as Open Grid Services Architecture (OGSA) and tools such as provided by Globus Toolkit provide the necessary framework [3]. Grid computing uses open source protocol and software called Globus. To achieve the promising potentials of tremendous distributed resources, effective and efficient scheduling algorithms are fundamentally important. The scheduler is responsible for the management of job, reservation, Network, Failure and data. As a summery when the user submits the job with the required specifications, the scheduler has to perform the Resource discovery and selecti-on, then the schedule will make some preparation for the listing of the resources .Once this is done , the job will be Submitted and executed on the selected resources . At last the job will be terminated [4]. Computational grids are among the first type of grid systems. They were developed due to the need to solve problems that require processing a large quantity of operations or data. In spite of the fact that the capacity of the computers continues to improve, the computational resources do not respond to the continuous demand for more computational power. Moreover, many statistical studies have shown that computers from companies, administration, etc. are usually underutilized. One of the main objectives of the computational grid is maximize the utilization of resources to benefit from the existence of many computational resources through the sharing [5]. In order to perform the scheduling process, the grid scheduler has to follow a series of steps[6]: (1) Collecting information of jobs submitted to the grid, (2) Collecting available resource information , (3) Computation of the mapping of jobs to selected resources, (4) Jobs allocating according to the mapping, and (5) Monitoring of job completion.

II. RELATED WORK

In the field of grid resource management and job scheduling, researchers have done much valuable work. Various algorithms have been proposed in recent years and each one has particular features and capabilities. In this section we review several scheduling algorithms which have been proposed in grid environment. Jobs submitted to a grid computing system need to be processed by the available resources. Best resources in terms of processing speed, memory and availability status are more likely to be selected for the submitted jobs during the scheduling process. Best resources are categorized as optimal resources. A framework for bandwidth-aware grouping-based scheduling [8] is proposed to improve the dissemination of jobs in Grid computing. The analysis conducted on the simulated framework has shown that the proposed framework is able to perform job scheduling using information of network and resource conditions. The proposed framework has also shown good comparative results in better job scheduling compared to a non-bandwidth-aware job grouping scheduling framework. This framework per se appears to be a viable alternative for delivering job scheduling process in the Grid environment with better job scheduling performance. A grouping-based fine-grained job scheduling model is presented in [9] by Liu and Liao, the fine-grained jobs grouped into forming coarse-grained are allocated to the available resources according to their processing capabilities and network bandwidth in Largest Job First(LJF) order. But here resource are selected in FCFS order, there is no priority for selecting resources. A dynamic job grouping-based scheduling algorithm [15] groups the jobs according to MIPS of the available resources. It reduces communication time and processing cost but does not focus on dynamic resource characteristics and grouping strategy to use resources. The fine grained job Scheduling Model [21] in Grid Computing is a grouping based job scheduling strategy that has taken memory constraint of individual jobs together with expected execution time at the job level into account rather than at the group level. The proposed model reduces the waiting time of the grouped jobs. Advantages of this algorithm compared with others are: It reduces the total processing time of jobs. It maximizes the utilization of the resource Grouping the jobs fine-grained into grouping course grained will reduce the network latencies. The above analysis of various grouping based job scheduling strategy presents some of their advantages and disadvantages The objective of this thesis is to present an efficient job scheduling approaches for coarse grained jobs in grid, which is based on previous research in grouping job scheduling. To solve the problems mentioned above, coarse grained job scheduling mechanism is presented in this approach.

III. COARSE GRAINED JOB SCHEDULING

The Grid Information Service (GIS) provides information about all the registered resources in a grid. This service keeps track of all of the resources characteristics in the grid. GIS collects resource characteristic information like operating system, system architecture, processing capability, network bandwidth and processing cost. It also provides users the availability information of the resources. The system collects jobs from the grid users specified by their Job Id, Job Length (in Million Instructions (MI)), Job Input size (in Mb) and total number of jobs submitted by the user. After collecting details of user jobs, scheduler collects all the available computational grid resources information specified by their Id, MIPS (computational power of the resource in Million Instructions per Second), BW (in Mb/sec.), Cost (in cost/sec.). Once scheduler receives the details of user jobs and the available resources, it will select a resource and multiplies the resource MIPS with the given granularity time [7], which is the time within which a job is processed at the resource. The value of this calculation produces the total Million Instructions (MI) for that particular resource to process within a particular granularity time. The system selects jobs according to Coarse Grain Job Scheduler (CGJS) which works according to following steps:

Step 1: $R1 = \sum_{i=1}^{n} \text{MIPS(Ri)}$, R stands for Resource
Step 2: $R2 = \sum_{i=1}^{n} \text{MEM(Ri)}$
Step 3: A1= R1 /n and A2=R2/n
where n= number of resources
Step 4: A= A1 + A2
Step 5: Bi = MIPS (Ri) + MEM (Ri)
where i= 1, 2… n
Step 6: <Bl> < A < <Bg>
where <Bl> = set of Bi < A and arranged in descending order
<Bg> = set of Bi > A and arranged in ascending order By implementing this policy in scheduling best resources are selected by taking boundary values one from each sorted list. New IDs are assigned to grouped jobs and scheduler submits the job groups to their respective resources for computation. After executing the group job, results goes to back to the corresponding users and the resource is again available to scheduler system.

```
Coarse grain job I = 0;
Taking summation of MIPS of each resources and Memory of each resources. $R1 = \sum_{i=1}^{n} \text{MIPS(Ri)}$ , $R2 = \sum_{i=1}^{n} \text{MEM(Ri)}$
 A1= R1 /n and A2=R2/n where n= number of resources
 A= A1 + A2
Bi = MIPS (Ri) + MEM (Ri)   where i= 1, 2… n
 <Bl>   < A <  <Bg>
where <Bl> = set of Bi < A and    arranged in descending order
     <Bg> = set of Bi > A and arranged in ascending order.
  for  i= 0 → RES_LIST_SIZE -1 do
  Take each resource from set1 and set2 one by one;
 (Coarse_grain_jobi)MI=0;
        Resource(MI) = Resource list(MIPS) * GS  ;
  Resource(BW) = Baudrate * TC ;
  for j=0→  finegrainjoblist_size -1  do
  while ( j <= joblist_size -1)
    if   (((CGJ)MI <=RiMI) &&  (CGJ)MS <=RiMS) && ((CGJ)BW <= RIBW))) then
    assign fine grain_job to coarse grain job
    else
 fine_grain job cannot group with coarsegrain job.
end if;
  end while
  end for Specify a unique id  for new generated coarse grain job; Place newly created job to target resource for computation. Receive completed coarse grain job for resources
 end for
```

The overall explanation of Algorithm is as follows: once the scheduler generate resource list by collecting characteristics of resources according to proposed policy. Then, jobs are allocated to resource from resource set. This Grouping strategy is based on processing capability (in MIPS), bandwidth (in Mb/s), and memory-size (in Mb) of the available resources. Jobs are put into the job group until condition 14 is satisfied. Where, MIPS (Million Instruction Per Second) is processing capability of the resource, MI (Million Instruction) is job's required computational power and Granularity size is user defined time which is used to measure total no. of jobs that can be completed within that specified time, CGJ_MS is required Memory Size of group jobs and R_MS is available Memory of the resource size of resource, Baud rate is the bandwidth capacity of resource with communication time. Condition (1) required computational power of grouped jobs shouldn't exceed to the resource's processing capability. In (2) Memory-size requirement of grouped job shouldn't exceed to the resource's memory-size capability. In (3) Memory-size of the grouped job shouldn't exceed to resource's transfer capability within a given time period. These are the main factors in job grouping strategy that influences the way job grouping is performed to achieve the minimum job processing time and maximum resource utilization of the Grid resources. Scheduler selects job and resource from resource set one by one from each set. If all three conditions are satisfy it will assign jobs to coarse grained job list. This step will be repeated to schedule remaining jobs. The grid resources process the received job groups and send back the computed job groups to the scheduler. The scheduler then gathers the computed coarse grained job and produce output.

IV. IMPLEMENTATION AND EVALUATION

The GridSim tool kit used to simulate grid environment is a Java based grid simulation tool. This tool kit supports modeling and simulation of heterogeneous grid resources. It also provides Application Programming Interface (API) or creation of user jobs, resource management and job schedulers [22]. It is assume that grid consists of nine resources for experiment purpose. Each resource contains three computing nodes, and each computing nodes contains four processing elements. Processing power (MIPS), network speed (Bandwidth), cost and total MIPS. In application model, it is assume that jobs which are submitted to the grid are independent tasks with no required order of execution. The jobs are of different computational size, each job also have different input files size requirements. The Job length of each job is presented in Millions Instructions (MI). Various performance parameters are used to evaluate the performance of algorithm. The criterion includes total execution time, computation time and processing cost. An average of ten runs is used in order to account realistic conditions. The figure 4.1 depicts the total computation time taken by the resources to compute the assigned jobs for all the algorithms. The proposed algorithm gives best performance compared to previous. The graph shows that the proposed scheduling algorithm is able to operate in conditions of different number of jobs with good improvement over an existing job grouping based scheduling algorithm.

Number of Gridlets	**CGJS**	**CGG**
100	91	126
200	134	157
300	174	209
400	193	239
500	238	281
600	281	335
700	299	366
800	335	401
900	370	441
1000	402	483

Table. 1: Computation of Total Computation Time of different numbers of jobs with average MI of 10 and granularity time 10 time unit

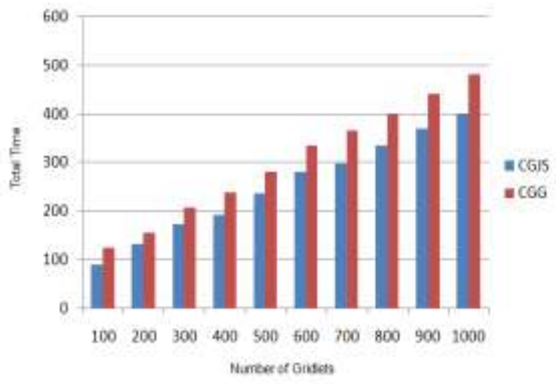

Fig. 1: Computation of Total Computation Time of different numbers of jobs

A. Processing Cost

In terms of processing cost, the total time each Gridlet spends at the grid resource of computation is taken into account for calculating the processing cost. The figure 4.3 depicts the graph of the results collected from the simulations. The proposed algorithm reduces total computation time for that reason it also reduce the processing cost.

Number of Gridlets	**CGJS**	**CGG**
100	14142	17361
200	27790	35050
300	44437	48746
400	59961	62839
500	75223	83968
600	90661	103124
700	104317	115674
800	120517	134682
900	133874	153645+
1000	149961	170688

Table. 2: Computation of Total Processing Cost of different numbers of jobs with average MI of 10 and granularity time 10 time unit

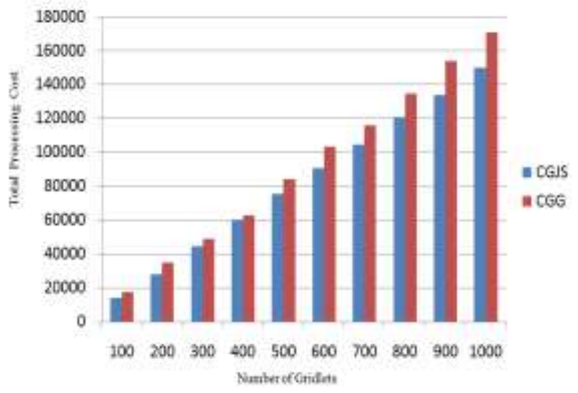

Fig. 2: Computation of Total Processing Cost of different numbers of jobs

Experimental results depict that the proposed Coarse grain Dynamic Grouping Based Scheduling algorithm reduces the total computation time and processing cost of the course grain job thereby utilizing the resources effectively.

V. CONCLUSION

A Coarse grained job scheduling algorithm with dynamic strategy has been proposed in order to reduce processing time and computational cost. The grouping algorithm improves the processing time of coarse grained job. The simulation results demonstrates efficiency and effectiveness of proposed algorithm in reducing processing time and computational cost and also shown that it is able to achieve the mentioned objectives in grid environment. The total number of Gridlet group should be created such that the processing load among the selected resources are balanced in order to avoid large number of Gridlets to one resource. The proposed approaches have been critically analyzed and few limitations have been observed. These limitations can be studied and refined. Additionally, the simulation environment is semi-dynamic and it can't reflect in the real computational grid environment sufficiently that promotes further research in the proposed area. In future, this work can be extended to design a parallel grid scheduler and some dynamic factors should be taken into account such as high priority, network delay, jobs with deadline, Quality of Service.

REFERENCES

[1] Ian Foster, Carl Kesselman, Steven Tuecke "The Anatomy of the Grid Enabling Scalable Virtual Organizations "Intl J. Supercomputer Applications, 2001.
[2] Fran Berman, Geoffrey Fox, and Tony Hey, The Grid: past, present, future Grid Computing – Making the Global Infrastructure a Reality, edited by F. Berman, G. Fox and T. Hey. 2002 John Wiley & Sons, Ltd.
[3] Rajkumar Buyya1, Manzur Murshed2, and David Abramson3,, A Deadline and Budget Constrained Cost-Time Optimization Algorithm for(2nd International Workshop on Active Middleware Services (AMS 2000), pp. 221-230, August 1, 2000, Pittsburgh, USA, Kluwer Academic Press USA, 2000.)
[4] Rajkumar Buyya and Srikumar Venugopal. A gentle introduction to grid computing and technologies. CSI Communications, 29(1) 19, July 2005. Computer Society of India (CSI).
[5] Xhafa and Ajith Abraham. Computational models and heuristic methods for grid scheduling problems. Future Generation Computer Systems, 26(4):608 - 621, 2010.
[6] Nithiapidary Muthuvelu, Junyang Liu, Nay Lin Soe, Srikumar Venugopal, Anthony Sulistio,and Rajkumar Buyya. A dynamic job grouping-based scheduling for deploying applications with _ne-grained tasks on global grids. In Proceedings of the 2005 Australasian workshop on Grid computing and e-research - Volume 44, ACSW Frontiers '05, pages 41-48, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.
[7] Ng Wai Keat, Ang Tan Fong, Ling Teck Chaw, and Liew Chee Sun. Scheduling framework for bandwidth-aware job grouping-based scheduling in grid computing. Malaysian Journal of Computer Science, 19(2):117-126, 2006.
[8] Quan Liu and Yeqing Liao. Grouping-based _ne-grained job scheduling in grid computing. In Proceedings of the 2009 First International Workshop on Education Technology and Computer. Science - Volume 01, pages 556-559, Washington, DC, USA, 2009. IEEE Computer Society.
[9] Rajendra Sahu and Anand K Chaturvedi. Article: Many-objective comparison of twelve grid scheduling heuristics. International Journal of Computer Applications, 13(6):9-17, January 2011. Published by Foundation of Computer Science.
[10] Yang Gao, Hongqiang Rong, and Joshua Zhexue Huang. Adaptive grid job scheduling with genetic algorithms. Future Gener. Comput. Syst., 21:151-161, January 2005.
[11] Ruay-Shiung Chang, Jih-Sheng Chang, and Po-Sheng Lin. An ant algorithm for balanced job scheduling in grids. Future Gener. Comput. Syst., 25:20-27, January 2009.
[12] Lei Zhang, Yuehui Chen, and Bo Yang. Task scheduling based on algorithm in computational grid. Intelligent Systems Design and Applications, International Conference on, 2:696-704, 2006.
[13] N. Muthuvelu, Junyan Liu, N.L.Soe, S.venugopal, A. Sulistio, and R. Buyya "A dynamic job grouping-based scheduling for deploying applications with fine-grained tasks on global grids," in Proc of Australasian workshop on grid computing, vol. 4, 2005,pp. 41–48
[14] T.F. Ang, W.K. Ng, "A Bandwidth-Aware Job Scheduling Based Scheduling on Grid Computing", Asian Network for Scientific Information, vol. 8, No. 3, pp. 372-277, 2009
[15] S. Gomathi and Dr. D. Manimegalai .An Analysis of MIPS Group Based Job Scheduling Algorithm with other Algorithms in Grid Computing. IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 6, No 3, November 2011 ISSN 1694-0814.

[16] Rajkumar Buyya and Manzur Murshed. Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing. CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE (CCPE), 14(13):1175-1220, 2000