Grouping Based on Priority for Job Scheduling in Grid Computing

Dr. S. Y. Amdani, Prof. S. R. Jadhao, Jayashree S. Chirde

***Abstract*— Grid computing has emerged as a distributed methodology that coordinates the resources that are spread in the heterogeneous distributed environment. Although Grids have been used for executing applications with compute-intensive jobs, there exist several applications with a large number of small jobs. The overall processing undertaking of these applications involves high overhead time and cost in terms of (i) job is transmission to and from Grid resources and, (ii) job is processing at the Grid resources. Therefore, there is a need for an efficient Priority job grouping-based scheduling system to dynamically assemble the individual small jobs of an application into a group of jobs, and send these coarse-grained jobs to the Grid resources. This dynamic grouping should be done based on the processing requirements of each job of application, Grid resources availability and their processing capability. In this paper a scheduling strategy that performs dynamic job grouping activity at runtime and convey the detailed analysis by running simulations. In addition, job processing granularity time is introduced to facilitate the job grouping activity in determining total amount of jobs that can be processed in a resource within a specified time. It will minimize the processing time and processing cost and increase the resources utilization.**

***Index Terms*— Grid computing, Job grouping, Job Scheduling.**

I. INTRODUCTION

The emerging computational Grids, as mentioned by Foster and Kesselman (1999), provide a new platform for executing large-scale resource intensive applications on a number of heterogeneous computing resources across political and administrative domains. Typically, an application requires an execution set that consists of several jobs, where each job is considered as the atomic

unit of computation. In a Grid computing environment, Berman, Fox and Hey (2002) describe that a scheduler is responsible for selecting the best suitable machines or computing resources in the Grid for processing jobs to achieve high system throughput.

To enhance the scheduling capability on cloud computing based software systems, simulations are used to facilitate the evaluations on different approaches under various runtime scenarios in a cloud environment. The study proposes a task grouping scheduling algorithm combined with shortest job first and bandwidth awareness algorithms in an attempt to reduce the waiting time and its associated processing costs(Jia Ru and Jacky Keung, 2013).

Grids consist of resources connected over high latency networks. Thus, they implicitly favour coarse-grained jobs with a heavy computational component or resources, so that the computation-communication ratio (CCR) motivates distributing them for processing on remote resources as referred to in Gray (2003).

The job grouping is done based on a particular granularity time. ***Granularity time*** is the time within which a job is processed at particular resources. It is used to measure total amount of jobs that can be completed within a specified time in a particular resource. Relationship between the total number of jobs, processing requirements of those jobs, total number of available Grid resources, processing capabilities of those resources and the granularity time should be determined in order to achieve the minimum job execution time and cost, and maximize utilization of the Grid resources. In order to evaluate the proposed job scheduler, GridSim toolkit, as discussed in Buyya and Murshed (2002), is used to model and simulate Grid resources and application scheduling.

MI : Million instructions or processing requirements of a user job
MIPS : Million instructions per second or processing capabilities of a resource
Processing Time : Total time taken for executing the user jobs on the Grid
Computation Time : Time taken for computing a job on a Grid resource
JobList : List of user jobs submitted to the broker
RList : List of available Grid resources
JList_Size : Total number of user jobs
RList_Size : Total number of available Grid resources
Job_Listi_MI : MI of ith user job
RListj_MIPS : MIPS of jth Grid resource
Granularity_time : Granularity time (time in seconds) for the job grouping activity
Total_JMI : Total processing requirements (MI) of a job group (in MI)
Total_RMIj : Total processing capabilities (MI) of jth resource
Total_RMIj = RListj_MIPS *Granularity_time
GJobList : List of job groups after job grouping activity
TargetRList : List of target resources of each job group

Figure 1: List of terms and their definitions.

II. ALGORITHM LISTING

Figure 1 shows the terms that are used throughout this paper and their definitions. The Priority job grouping and scheduling algorithm is presented in Figure 2.An example of priority job grouping and scheduling scenario where 30 user

Jayashree Chirde, ME in Computer Science and Engg from bncoe pusad
Dr.. S. Y. Amdani, Ph.D (CSE) de gree from SGB Amravati university, Amravati and Working as Head and Associate Professor in Deptt. Of CSE B.N.C.O.E., Pusad (India),
Prof. Satish R. Jadhao, Assistant Profes-sor in Computer Science & Engineering Department at B.N.College of Engineering, Pusad.

jobs with small processing requirements (MI) are grouped into two job groups according to the processing capabilities (MIPS) of the available resources and the granulariytime. The overall explanation of Figure 2 is as follows: once the user jobs are submitted to the broker or scheduler, the scheduler gathers the characteristics of the available Grid resources. Then, it selects a particular resource and multiplies the resource MIPS with the granularity time where the resulting value indicates the total MI the resource can process within a specified granularity time.

The scheduler groups the user jobs by accumulating the MI of each user job while comparing the resulting job total MI with the resource total MI. If the total MI of user jobs is more than the resource MI, the very last MI added to the job total MI will be removed from the job total MI. Eventually, a new job (job group) of accumulated total

MI will be created with a unique ID and scheduled to be executed in the selected resource. This process continues until all the user jobs are grouped into few groups and assigned to the Grid resources.

Fig 2: Grouping based on priority for job scheduling in grid computing.

Algorithm Grouping based on priority for job Scheduling Algorithm(GBPS)

```
m := 0;
 for i:= 0 to JobList_Size-1 do
 for j:=0 to RList_Size-1 do
 Total_JMI := 0;
Give the priority to the smallest job first based on MI
and sort into ascending order
Sort the resource Based on MIPS in descending order
 Total_RMIj :=RListj_MIPS*GT;
 while Total_JMI  Total_RMI j and i JobList_Size-1 do
Total_JMI := Total_JMI + JobListi_MI;
i++;
end while
i--;
. if Total_JMI > Total_RMIj then
Total_JMI := Total_JMI – JobListi_MI;
i--;
end if
Create a new job with total MI equals to Total_JMI;
Assign a unique ID for the newly created job;
 Place the job in GJobListm;
Place RListj in TargetRListm;
m++;
end for
end for
for i:= 0 to GJobList-1 do
Send GJobListi to TargetRListi for job computation;
end for
//Job computation at the Grid resources
for i:= 0 to GJobList-1 do
Receive computed GJobList i from TargetRList i;
end for
```

The scheduler then sends the job groups to their corresponding resources for further computation. The Grid resources process the received job groups and send back the computed job groups to the Grid user. The scheduler then gathers the computed job groups from the network through its I/O port or queue.The granularity time is set to 5 seconds for example. In this first we assign priority to smallest job, and job sort on ascending order.

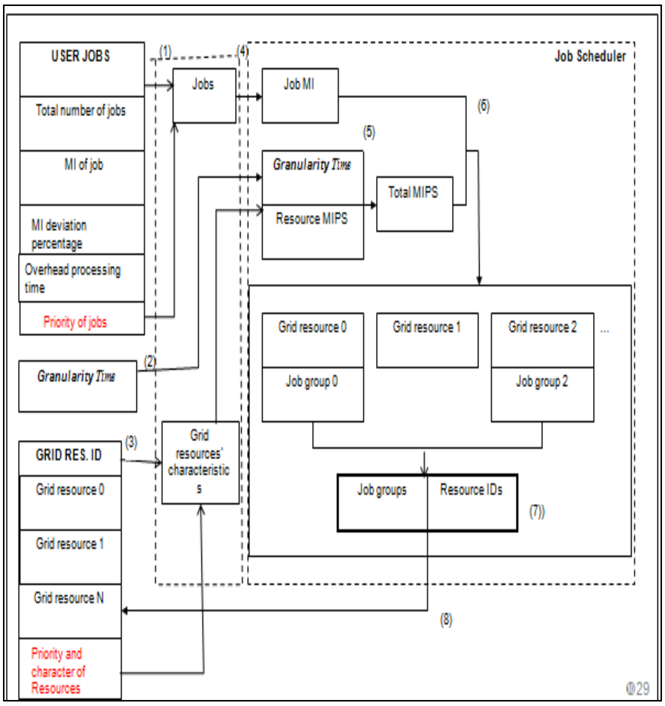

Fig3: Simulation strategy for job grouping-based schedule

Resource will sort in descending order .The scheduler selects a resource of 550.0 MIPS and multiply the MIPS with the given granularity time. In total, that particular resource can process 2750MI of user jobs within 5 seconds. The scheduler then gathers the user jobs by accumulating their MI up to 2750 MI. This process continues until all the jobs are grouped into a number of groups. Finaly, the scheduler sends the groups to the resource for job computation.

III. EVALUATION

Implementation with GridSim

GridSim toolkit is used to conduct the simulations based on the developed scheduling algorithm. Figure 4 depicts the simulation strategy of the proposed dynamic job grouping-based schdu

ler which is implemented using the GridSim toolkit. The system accepts total number of user

jobs, processing requirements or average MI of those jobs, allowed deviation percentage of the MI, processing overhead time of each user job on the Grid, granularity time of the job grouping ac-tivity and the available Grid resources in the Grid environment (step 1-3). Details of the available Grid resources are obtained from Grid Informa-tion Service entity that keeps track of the re-sources available in the Grid environment. Each Grid resource is described in terms of their vari-ous characteristics, such as resource ID, name, total number machines in each resource, total processing elements (PE) in each machine, MIPS of each PE, and bandwidth speed.

In this simulation, the details of the Grid re-sources are store in a file which will be retrieved during the simulations.

Example:
Job ID : 0 Length : 160.16078016819125 MI
Job ID : 1 Length : 160.54401925882078 MI
Job ID : 2 Length : 161.9805842486006 MI
Job ID : 3 Length : 162.12566887029269 MI
Job ID : 4 Length : 162.82616079189197 MI
Job ID : 5 Length : 164.57971263206613 MI
Job ID : 6 Length : 171.02895844180182 MI
Job ID : 7 Length : 173.6418583871137 MI
Job ID : 8 Length : 174.8793200690919 MI
Job ID : 9 Length : 176.61689635505815 MI
Job ID : 10 Length : 177.5711724199505MI
Job ID : 11 Length : 177.8897266470528 MI
Job ID :12 Length : 178.61914716710962MI
Job ID : 13 Length : 184.7461640796671 MI
Job ID : 14 Length : 187.5900139326515MI
Job ID : 15 Length : 190.1852696637688MI
Job ID : 16 Length : 193.4483522627478 MI
Job ID : 17 Length : 202.1728600796611 MI
Job ID : 18 Length : 204.2364668609247 MI
Job ID : 19 Length : 207.3668068770985 MI
Job ID : 20 Length : 208.2342176886634 MI
Job ID : 21 Length : 215.6950318068347MI
Job ID : 22 Length : 216.4239953426662 MI
Job ID : 23 Length : 221.6760252119536 MI
Job ID : 24 Length : 224.6646565163723MI
Job ID : 25 Length : 224.9990869099843 MI
Job ID : 26 Length : 225.78485097219695 MI
Job ID : 27 Length : 229.67799915569015 MI
Job ID : 28 Length : 234.69409125389174 MI
Job ID : 29 Length : 235.08232056973858 MI
Starting GridSim version 5.0

RESOURCE SORT IN DESENDING ORDER

Resource R1, with id = 17, cost = 100.0,
MIPS = 550.0
Resource R2, with id = 13, cost = 100.0,
MIPS = 380.0
Resource R3, with id = 9, cost = 100.0,
MIPS = 350.0
Resource R4, with id = 5, cost = 120.0,
MIPS = 220.0
=====START SIMULATION========
GroupGridlet 0; composed of 15 jobs, (job 0 to job 14); Total Length: 2574.8001834693605; Resource ID: 17 Resource Capacity : 2750.0
Resource Utilization : 93.62909758070401
GroupGridlet 1; composed of 9 jobs, (job 15 to job 23); Total Length: 1859.4390257943192; Resource ID: 13 Resource Capacity : 1900.0
Resource Utilization : 97.86521188391154
GroupGridlet 2; composed of 6 jobs, (job 24 to job 29); Total Length: 1374.903005377874; Resource ID: 9 Resource Capacity : 1750.0
Resource Utilization : 78.5658860215928

Process_Time = Overhead Time + Simulation Time
Overhead Time: The Processing Time required by a device prior to the execution of a command/job
Simulation Time : Total time required to Process Gridlet/Job In the simulation time we will add the gridlet grouping time ,time taken to submit all the groups to resources, total processing overhead time, gridlet processing time, time taken to receive all the processed gridlet.

Processing cost

Processing cost is computed based on the actual time taken for computing the gridlet at the grid resource and at the cost rate specified at grid resource.

Process_Cost = T*C

T=Total CPU Time for Job execution

and

C=Cost per second of the resource

Following graph show processing time and cost
The graph comparison of simple grouping ,grouping with granularity, Priority grouping granularity. Here job are 30 and granularity 5sec.

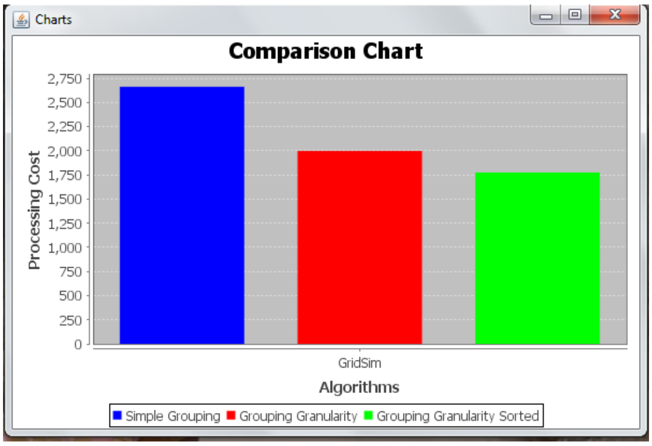

Fig3: Show comparison of processing cost.

Processing Costs: Simple Grouping=2664 Grouping Granularity=1998,Grouping Granularity Priority=1776.

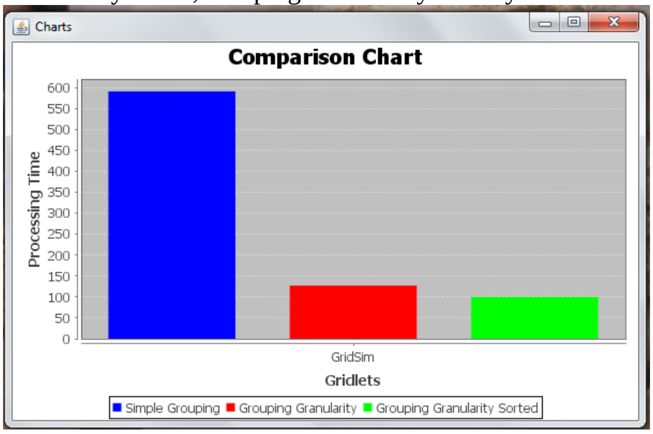

Fig4: Show comparison of processing time.

Processing time : Simple Grouping=591 Grouping Granularity=127,Grouping Granularity Priority=100.

IV. CONCLUSION

The Grouping based on priority strategy results in increased performance in terms of low processing time and processing cost and resource utilization if it is applied to a Grid application with a large number of jobs where each user job holds small processing requirements. Sending/receiving group job individually to/from the resources will increase the total processing time. Grouping based on priority strategy aims to reduce the impact of these drawbacks on the total processing time, processing cost and resource utilization. This strategy groups the small scaled user jobs into few job groups according to the processing capabilities of available Grid resources. This reduces processing overhead time of each user jobs and increases the resource utilization. (GBPS)Job grouping method allows total processing capabilities of available resource utilized during the job executions. GBPS minimize processing time and processing cost**.**

REFERENCES

[1] Foster and C. Kesselman, "The Grid: Blueprint for a New Computing Infrastructure, Morgan-Kaufmann", 1998.

[2] Jia Ru and Jacky Keung" An Empirical Investigation on the Simulation of Priority and Shortest-Job-First Scheduling for Cloud-based Software Systems" 2013 22nd Australian Conference on Software Engineering

[3] Nithiapidary Muthuvelu, Junyang Liu,Nay Lin soe,Srikumar Venugopal,Anthony Sulistio and Rajkumar Buyya,"A dynamic Job Grouping based Scheduling for Developing Application with Fine-Grained Tasks on Global Grids" Grid Computing and Distributed Systems (GRIDS) Laboratory Department of Computer Science and Software Engineering The University of Melbourne, Australia ICT Building, 111 Barry Street, Carlton VIC 3053

[4] Thu Zar Mon, and Cho Me Me Mg," MIPS Group Based Job Scheduling Model for Deploying Applications", International Conference on Advances in Engineering and Technology (ICAET'2014) March 29-30, 2014 Singapore.

[5] S. Gomathi, Dr.D.Manimegalai, "An adaptive grouping based Job Scheduling in Grid Computing", Proceedings of 2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN 2011.)

[6] Liang Yu, Gang Zhou, Yifei Pu," An Improved Task Scheduling Algorithm in Grid Computing Environment", Int. J. Communications, Network and System Sciences, 2011, 4, 227-231 doi:10.4236/ijcns.2011.44027 Published Online April 2011 (http://www.SciRP.org/journal/ijcns).

[7] Berman, F., Fox, G. and Hey, A. (2003): Grid Computting Making the Global Infrastructure a Reality. London,Wiley.

[8] Buyya, R. and Murshed, M. (2002): GridSim: A Toolkit for the Modeling, and Simulation of Distributed Resource Management, and Scheduling for Grid Computing. Journal of Concurrency and Computation:Practice and Experience (CCPE), **14**(13-15):1175-1220.

[9] P.SURESH, P. BALASUBRAMANIE," Grouping Based User Demand Aware Job Scheduling Approach For Computational Grid". IJEST Vol. 4 No.12 December 2012

[10] GridSim website. http://www.gridbus.org/gridsim

[11] K.Somasundaram, S.Radhakrishnan," Node Allocation In Grid Computing Using Optimal Resource Constraint (ORC) Scheduling ",IJCSNS International Journal of Computer Science and Network Security, June-2008.

[12] R.Manimala, P.Suresh," Load Balanced Job Scheduling Approach forGrid Environment",

[13] G.Malathy, Dr.A.Saradha," Mapping of Independent Task Classes onto GRIDSIM", International Conference on Advanced Computer Technology (ICACT) 2011Proceedings published by International Journal of Computer Applications® (IJCA).

[14] Thu Zar Mon, and Cho Me Me Mg," MIPS Group Based Job Scheduling Model for Deploying Applications", International Conference on Advances in Engineering and Technology (ICAET'2014) March 29-30, 2014 Singapore.

[15] S. Gomathi, Dr.D.Manimegalai, "An adaptive grouping based Job Scheduling in Grid Computing", Proceedings of 2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN 2011.)

[16] Liang Yu, Gang Zhou, Yifei Pu," An Improved Task Scheduling Algorithm in Grid Computing Environment", Int. J. Communications, Network and System Sciences, 2011, 4, 227-231 doi:10.4236/ijcns.2011.44027 Published Online April 2011 (http://www.SciRP.org/journal/ijcns).

[17] S. Gomathi, 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 (Online): 1694-0814www.IJCSI.org.

[18] Pinky Rosemarr1, Payal Singhal, Ravinder Singh," A Study of Various Job & Resource Scheduling Algorithms in Grid Computing"

[19] S.Gokul Dev, R.Lalith Kumar," Job Scheduling in Grid Computing using User Deadline", S.Gokul Dev et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, 1387-1392.

[20] Zahra Pooranian, Mohammad Shojafar, Jemal H. Abawajy, and Mukesh Singhal, "GLOA: A New Job Scheduling Algorithm for Grid Computing", International Journal of Artificial Intelligence and Interactive Multimedia, Vol. 2, N

Dr.. S. Y. Amdani received his Ph.D (CSE) de gree from SGB Amravati university, Amravati and Working as Head and Associate Professor in Deptt. Of CSE B.N.C.O.E., Pusad (India), and life member of Indian Society for Technical Education New Delhi. He has published and presented many research papers in International conferences & journals. He is also Edi-torial Board member of many International Journals.

Prof. Satish R. Jadhao is working as an Assistant Profes-sor in Computer Science & Engineering Department at B.N.College of Engineering, Pusad. He has completed his B.E and M.E in Computer Science & Engineering and has published various national & international papers. He is specialized in Design and Analysis of Algorithm (DAA), JAVA,J2EE, Grid Computing.

Jayashree Chirde hails from pusad and born on 3rd may 1989.she did her schooling K.d. High school pusad ,complete graduation (BE) in stream computer science and engg from bncoe pusad with 70% and Appeared for ME in Computer Science and Engg with same college and willing work on grid computing. She publish paper on in International conferences.