http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 8 (3): 372-377, 2009
ISSN 1812-5638

A Bandwidth-Aware Job Grouping-Based Scheduling on Grid Environment

T.F. Ang, W.K. Ng, T.C. Ling, L.Y. Por and C.S. Liew
Department of Computer System and Technology,
Faculty of Computer Science and Information Technology,
University of Malaya, 50603 Kuala Lumpur, Malaysia

Abstract: This study explores the feasibility of job scheduling strategies and extend the job grouping-based approach using the idea of bandwidth-awareness. As today's best-effort network generally experiences low bandwidth and high delay, we aim to maximize the Grid resource utilization and reduce the delay by considering the bandwidth criterion. A simulation environment using GridSim is developed to model job scheduling process. Exploiting the simulation environment, a job scheduling strategy that encompasses the job grouping concept coupled together with bandwidth-aware scheduling is proposed and evaluated. The proposed scheduling strategy focuses on grouping independent jobs with small processing requirements into suitable jobs with larger processing requirements and schedules them in accordance with indeterminist network conditions. The simulation result demonstrates that the proposed strategy succeeds in minimizing the total processing time by at most 82% as compared to its counterpart.

Key words: Grid computing, GridSim, job scheduling

INTRODUCTION

The emergence of the accessibility to interconnected powerful computers and the availability of high-speed network technologies have opened up vast amount of opportunities in using distributed computers to conduct various job processing. The concept of forming pool of resources has led the way into creating large computing facilities which are currently popular known as Grid computing (Foster and Kesselman, 2003). Chetty and Buyya (2002) termed the Grid as an analogy to a power Grid that offers consistent, pervasive, dependable and transparent access to electricity irrespective of its source.

In a network-based parallel computing system, job scheduling is one of the most challenging problems. In the Grid environment, the situation is even more difficult in which resources may be located at different geographically areas with different administrative control. The computational resources in a Grid are usually high-performance computers such as clusters or parallel machines. Shan *et al.* (2003) asserted that the most common objective of job scheduling is to match suitable jobs into parallel computing system and minimize the total execution time.

Conventional parallel system models are connected to homogeneous computing nodes in a geographically small area network such as LAN. Therefore, the consideration on the communication cost is not significant. On the other hand, the Grid cannot ignore various computing performance at each node and the communication cost as the nodes in Grid are heterogeneous and they are distributed over many different places. Grid computing is a type of distributed computing that involves wide-area networks. This new breed of computing has enabled large amount of possibilities in conducting various processing and collaborations. However, the situation has also posed several problems especially in the area of job scheduling. Dail *et al.* (2003) observed that one of the adversities which the job scheduling in Grid has to confront is the problem posed by network properties between users and resources. Thus, new scheduling strategies need to be proposed to overcome the challenge. The new scheduling strategies may use some of the conventional scheduling concepts to merge them together with some network aware strategies to provide solutions for better and more efficient job scheduling.

Muthuvelu *et al.* (2005) discovered that the Grid implicitly favours coarse-grained jobs with heavy computational component, so that the Computation-Communication Ratio (CCR) encourages distribution of these jobs to remote resources. On the other hand, an application with large amount of fine-grained jobs is not favoured because the total communication time for

Corresponding Author: Ang Tan Fong, Department of Computer System and Technology,
Faculty of Computer Science and Information Technology,
University of Malaya, 50603 Kuala Lumpur, Malaysia

transmitting each job between the host and the resource seems to exceed the total computation time of each job at the resource. Muthuvelu *et al.* (2005) proposed the Dynamic Job Grouping strategy which concentrates on maximizing the utilization of Grid resource processing capabilities and reducing the overhead time and cost taken to execute the jobs through a batch mode dynamic scheduling. In this algorithm, user jobs are submitted to the scheduler and the scheduler collects the required characteristics of the available resources. Next, it selects a specific resource and multiplies the resource processing capability specified in Million Instructions per Second (MIPS) with the granularity size, 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 size. Subsequently, the scheduler groups the user jobs by accumulating the MI of each user job based on the comparison made between the resulting job total MI and the resource total MI. If the total MI of the user jobs exceeded the resource MI, the last MI added to the job total MI will be removed from the job total MI. Then, a new job of accumulated total MI will be created with a unique ID and matched with a selected Grid resource. This grouping process continues until all the jobs are put in groups and assigned to the Grid resources. When the grouping and assigning of jobs have completed, the scheduler sends the grouped jobs to the corresponding resources for computation. The Grid resources compute the received job groups and send back the results of the jobs to the user.

Besides job grouping, the current throughput will affect the job completion time. The basic idea of bandwidth-aware scheduling was used for performing load balancing at the Stream Control Transmission Protocol (SCTP) layer. The main objective of this study is to maximize the chance of in-order delivery over multiple paths. In this approach, a rough estimation of the bandwidth available on each round-trip path is performed by sending pairs of SCTP heartbeats on each path. The corresponding heartbeat-acks are sent back to the receiver and evaluated by using the Packet-Pair Bandwidth Estimation (PPBE) technique (Carter and Crovella, 1996). In this approach, an association comprising K link-disjoint multi-hops are considered as $\Pi_1, \Pi_2, \ldots, \Pi_k$ and the transmission time of packets on the bottleneck link and their one-way propagation are assumed to be the main delays on each path. Thus, each $Đ_i$ is regarded as an estimated available bandwidth on the bottleneck link B_i and its single way propagation time T_i. A wall-time clock C is available and path Π_i is associated with a delivery time C_i at each endpoint. C_i indicates the earliest time when the opposite endpoint becomes idle after entirely received the last bit of data in flight on Π_i. Hence, Π_i which guarantee the fastest delivery for a packet of size D can be selected easily. If the local-end of $Đ_i$ is still busy sending a packet, $C_i = C_i+D/B$. Otherwise, $C_i = C+D/B_i+T_i$. With this idea, Casetti and Gaiotto (2004) proposed an enhancement of the SCTP protocol that aims at balancing load across multiple connections on disjoint paths in an accurate fashion, through the knowledge of bandwidth estimation on each path.

THE PROPOSED ALGORITHM

This study presents and evaluates an extension to dynamic job grouping-based scheduling strategy that concentrates on improving the scheduling of jobs with small scale processing requirements by reducing delaying factors in network environment as well as maximizing the utilization of grid resources. The reason for choosing dynamic job grouping strategy is that this approach is able to effectively handle the scheduling of multiple independent jobs with small scale processing requirements without causing the time and cost overheads to heavily affect the job transmission and processing. In short, the essence of bandwidth-awareness strategy combined together with the dynamic job grouping strategy is to further improve the reduction of transmission delay and overhead.

In the proposed algorithm, the scheduler retrieves information on the resources' processing capability and the bottleneck bandwidth to reach each of them. Then, the scheduler selects the resource with the largest bottleneck bandwidth and groups independent fine-grained jobs together based on the chosen resource's processing capability. These jobs are grouped to maximize the utilization of the resource. By grouping the fine-grained jobs, these jobs are represented in a coarse-grained form which will reduce the network latencies. The grouping process is repeated until all the jobs are in groups and every group is allocated with a resource. Then, these job groups are sent to the corresponding resources based on Largest Job First (LJF) strategy (Li, 2004) and the results of the processing are sent back to the user after they have been computed at their respective resources. The bandwidth-aware scheduling is taken as a part of the primary principle used in the proposed scheduling algorithm. The principle behind the bandwidth-aware scheduling is the scheduling priorities taking into consideration not only their computational capabilities but also the communication capabilities of the resources. Meanwhile, the LJF strategy is also implemented in the proposed algorithm to allow grouped jobs with longer

```
1. receive JobList, JListv
2. receive ResourceList, RListv
3. sort(RListv, BandwidthDescendingOrder)
4. for all jobs, ji in JListv
5. for all resources, ri in RListv
6. TotalJobMI, JMI = 0
7.        TotalResourceMI, RMI = ri_MIPS *
          Granularity_Size
8.        while JMI≤RMI and i≤JListv_Size – 1
9.        JMI = JMI+ji _MI
10.       i++
11.       endwhile
12.       i--
13.       if JMI>RMI then
14.       JMI = JMI-ji _MI
15.       i --
16.       endif
17.       create a new job, gji with JMI size
18.       insert gji into GroupedGridletList, GJListv
19.       insert job, gji and corresponding resource, ri
          information into TargetResourceList, TRListv
20. endfor
21. endfor
22. sort(GJListv, GridletLengthDescendingOrder)
23. while all grouped jobs, gji in GJListv
24. find gji_id in TRListv to get ResourceID, ri_id
25.       send gji to ri
26. endwhile
27. // After jobs are processed at the resources
28. while all grouped jobs, gji in GJListv
29. receive gji from ri
30. endwhile
```

Fig. 1: Bandwidth-aware job grouping-based scheduling algorithm pseudo code

execution time to be transmitted before the grouped jobs with shorter execution time. This creates the situation where grouped jobs with longer execution time are executed concurrently with the grouped jobs with shorter execution time. Therefore, this strategy would probably improve the overall execution time of the jobs. The pseudo code for the algorithm is presented in Fig. 1.

EVALUATION

Setup of simulation environment: GridSim, (Buyya and Murshed, 2002) has been used to create the simulation environment. An adequate network topology that is based on Belle Analysis Data Grid (BADG) testbed in

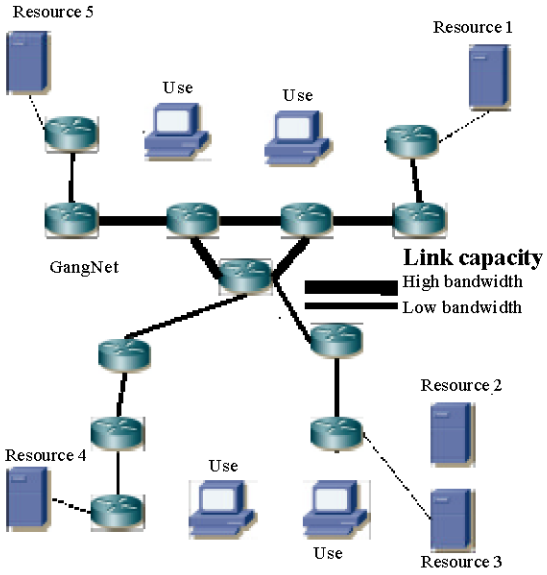

Fig. 2: BADG network topology

Australia is created. The BADG is a testbed used for analysing high-energy physics experiment data. Some parts of the BADG network are connected to the GrangeNet, a wide-area network with Grid services in Australia. Figure 2 shows the topology which is used for the evaluation of job scheduling strategy.

The BADG consists of 11 routers with 3 levels of link capacity. The actual bandwidths of all the links in the BADG topology have been adjusted and scaled down to reduce the simulation volume as well as to match the study purposes. Therefore, the resulting bandwidth of low capacity link is less than 1 Mbps, medium capacity link is between 1 and 10 Mbps and high capability link is more than 10 Mbps. For simplification of the setup, bandwidths for other links between users/resources and routers are connected with 1-100 Mbps. The actual values of all these links are generated within the range of their link capability using uniform distribution. The links in topology are assumed to be symmetric where the upload and download speed are at equivalent transfer rate. In addition, all links share the same characteristics such as Maximum Transmission Unit (MTU) size of 1500 bytes and latency of 10 m sec. The BADG adopts Open Shortest Path First (OSPF) with QoS extension as the routing protocol to select widest-shortest paths.

There are five resources created in four different locations in the Grid simulation environment. The characteristics of the resources are as stated in the Table 1. The processing ability of a resource's CPU is measured by Million Instructions per Second (MIPS) as per Standard Performance Evaluation Corporation (SPEC) CPU (INT) 2000 benchmark.

Table 1: Grid resources

Name	Resource type	A SPEC rating
R1	IBM eServer with dual Intel Xeon 2.6GHz, 2GB RAM	1050
R2	PC with Intel Pentium 2.0GHz, 512MB RAM	684
R3	IBM eServer with dual Intel Xeon 2.6GHz, 2GB RAM	1050
R4	IBM eServer with dual Intel Xeon 2.6GHz, 2GB RAM	1050
R5	IBM eServer with dual Intel Xeon 2.6GHz, 2GB RAM	1050

For creating a sufficiently functioning Grid environment, 5 users are created at each of the four locations. One user with the OSPF listener is chosen for the job scheduling evaluation purposes and the other 19 users are used for distributing additional loads into the network using First-Come-First-Served list scheduling. These 19 users create 10 jobs with each job processing requirement equals to 200 MI during the light load condition while 30 jobs with each job processing requirement equals to 200 MI during the heavy load condition. The purpose of these 19 users is to provide loads for the network and resources so that the simulation environment is capable of modelling a real world situation. Apart from that, background traffic is also created by each user to all other users and resources with inter-arrival time using a Poisson distribution approach with mean of 16 simulation seconds. The total number of packets for each interval is uniformly distributed in (50..100).

In the simulation, the total processing time is calculated in simulation seconds. When a grouped job is formed during job scheduling, the execution time, t^*_i, involves computational time, $t_{i,comp}$ and communication time, $t_{i,comm}$.

$$t^*_i = t_{i,comp} + t_{i,comm} \quad (1)$$

Where:

t^*_i = Execution time of a grouped job
$t_{i,comp}$ = Computation time of a grouped job
$t_{i,comm}$. = Communication time of a grouped job

Thus, the total processing time for a single job can be summarized as:

$$t_{proc} = t_{submit} + t^*_i/m + t_{receive} \quad (2)$$

Where:

t_{proc} = Total processing time
t_{submit} = Time taken to submit a job
$t_{receive}$ = Time taken to receive a processed job
m = Number of jobs in a group

However, for r_i grouped jobs $J_{i,1}$, $J_{i,2}$, …, $J_{i,ri}$, the effective execution time is the maximum time taken to execute all the grouped jobs from the time of the first job sent at the user until the final job received at the user after all the jobs are executed at the resources.

To summarize:

$$t^*_{effective} = T_{i\ end} - T_{i\ start} \quad (3)$$

Where:

$t^*_{effective}$ = Effective execution time
$T_{i\ end}$ = Time when the last job received
$T_{i\ start}$ = Time when the first job sent

The total processing time is calculated based on the time taken to group the jobs, to submit all the grouped jobs, to receive all the processed jobs and the effective execution time.

$$t_{proc} = t_{grouping} + t_{submit} + t^*_{effective} + t_{receive}, \quad (4)$$

Where:

t_{proc} = Total processing time
$t_{grouping}$ = Time taken to group jobs
t_{submit} = Time taken to submit all the grouped jobs
$t^*_{effective}$ = Effective execution time
$t_{receive}$ = Time taken to receive all the processed jobs

The grouping of jobs depends on a specific granularity size. Granularity size is the time within which a job is processed at the resources. As stated by Muthuvelu *et al.* (2005), this value is used to measure the total amount of jobs that can be completed within a specified time in a certain resource. It is one of the main factor in job grouping strategy that influences the way job grouping is performed to achieve the minimum job execution time and maximum utilization of the Grid resources.

RESULTS AND DISCUSSION

Simulation on different number of jobs in light traffic load condition: The simulations are conducted in light traffic load condition to explore the differences in the total processing time using job scheduling without grouping, job grouping-based scheduling and bandwidth-aware job grouping-based scheduling. The results of the simulations for three types of scheduling approach using jobs with processing requirement of average 200 MI along with deviation percentage of 20 and granularity size of 10 simulation seconds is shown in Fig. 3. From the Fig. 3 the bandwidth-aware job grouping-based scheduling has reduced the total processing time by 7 to 81% compared to the job grouping-based scheduling and has lessened the total processing time by 9 to 77% compared to the job scheduling without grouping.

The results show that the total processing time for using bandwidth-aware job grouping-based scheduling is increasing steadily below the total

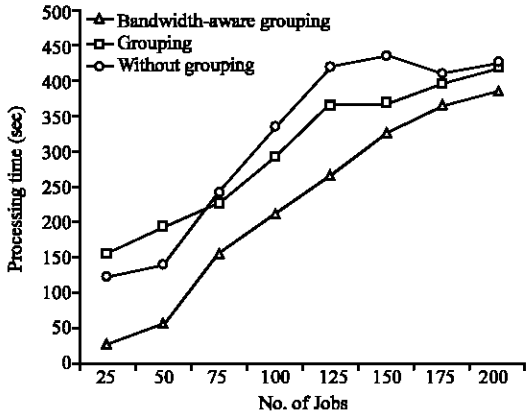

Fig. 3: Simulation of different No. of jobs in light traffic load condition

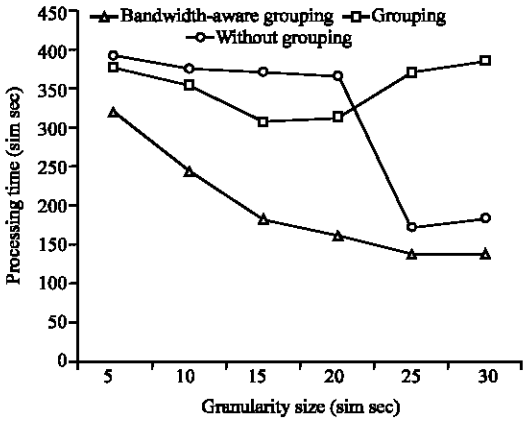

Fig. 5: Simulation of different granularity sizes in light traffic load condition

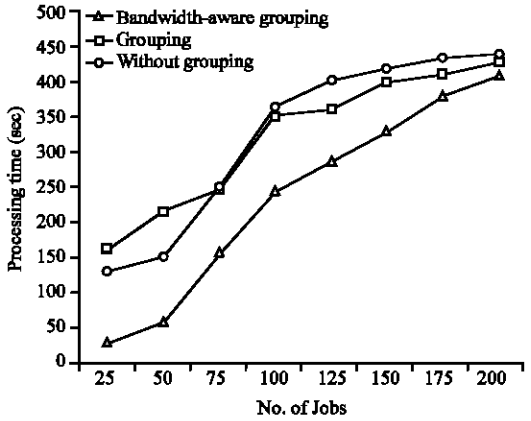

Fig. 4: Simulation of different No. of jobs in heavy traffic load condition

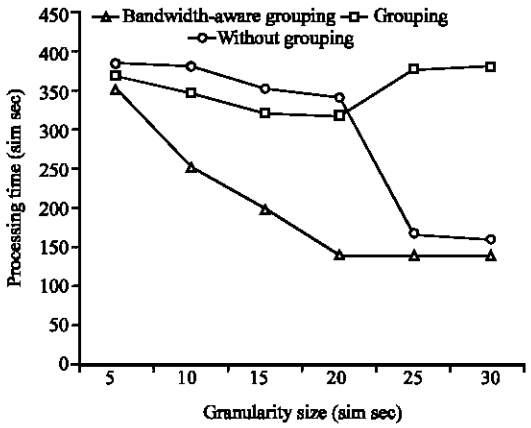

Fig. 6: Simulation of different granularity sizes in heavy traffic load condition

processing time of the job grouping-based scheduling and job scheduling without grouping as the amount of jobs increases.

Simulation on different number of jobs in heavy traffic load condition: The simulations are also carried out in heavy traffic load condition to analyse the differences in the total processing time using job scheduling without grouping, job grouping-based scheduling and bandwidth-aware job grouping-based scheduling. The simulations are conducted by using jobs with processing requirements of 200 MI, MI deviation percentage of 20 and granularity size of 10 simulation seconds.

As indicated in Fig. 4, the bandwidth-aware job grouping-based scheduling has decreased the total processing time by 4% to 82% as compared to the job grouping-based scheduling and has minimized the total processing time by 6 to 78% compared to the job scheduling without grouping.

The results shown in Fig. 4 indicated that the total processing time for using bandwidth-aware job grouping-based scheduling is increasing gradually below the total processing time of the job grouping-based scheduling and job scheduling without grouping as the amount of jobs increases.

Simulation on different granularity sizes in light traffic load condition: We conducted the following simulation to investigate the effect of granularity size for scheduling jobs in light traffic load condition by using three different approaches. A total of 120 jobs/gridlets with processing requirement of 200 MI and 20% MI deviation are used during each simulation. As shown in Fig. 5, the total job processing time for bandwidth-aware job grouping-based scheduling decreases when the granularity size increases. When the granularity size is small, the processing of jobs involves many resources as each resource only able to process small amount of processing instructions. Thus, it

increases the possibility where grouped jobs may be caught in queues at different resources. Meanwhile, when the granularity is large, more processing instructions may be process at each resource. Therefore, the amount of resources involved in processing the same amount of jobs may decrease. With the bandwidth-aware job grouping approach, it uses the links with the best bandwidth; thus, the amount of time for sending grouped jobs to the resources is reduced and subsequently reduced the total job processing time. For job grouping and without grouping approaches, the total processing is more than the bandwidth-aware job grouping-based approach because bottleneck bandwidth factor is not taken into consideration for job scheduling.

Simulation on different granularity sizes in heavy traffic load condition: We also conducted the same simulation as above in heavy traffic load condition. Figure 6 shows that in the heavy load condition, the bandwidth-aware job grouping-based scheduling still shows decreasing of total processing time as the granularity size increases.

CONCLUSION AND FUTURE WORK

In this study, we proposed a new job scheduling strategy in grid environment that incorporate bandwidth-awareness into dynamic job grouping. The new job scheduling strategy has reduced the total job processing time in both light and heavy traffic load condition when compared to job scheduling without grouping and job grouping scheduling strategies. Furthermore, the new proposed scheduling strategy has also reduced the total job processing time when different granularity sizes are used.

In the future, we are planning to implement an effective load balancing scheme during the job scheduling process. Besides, we are also intending to improve the job receiving strategy and incorporate the QoS requirements to the job scheduling strategy.

ACKNOWLEDGMENTS

Special thanks to the GridSim team, Dr. Rajkumar Buyya and Anthony Sulistio at the University of Melbourne and Gokul Poduval and Dr. Tham Chen Khong at the National University of Singapore for sharing the wonderful Grid simulation toolkit.

REFERENCES

Buyya, R. and M. Murshed, 2002. GridSim: A toolkit for the modelling and simulation of distributed management and scheduling for grid computing. Concurrency Comput: Practice Exp., 14: 1175-1220.

Carter, R.L. and M.E. Crovella, 1996. Measuring bottleneck link speed in packet-switched networks. Perform. Eval., 27-28: 297-318.

Casetti, C. and W. Gaiotto, 2004.. Westwood SCTP: Load balancing over multipaths using bandwidth-aware source scheduling. Proceedings of IEEE 60th Vehicular Technology Conference, September 26-29, Los Angeles, pp: 3025-3029.

Chetty, M. and R. Buyya, 2002. Weaving Computational Grids: How analogous are they with electrical grids? Comput. Sci. Eng., 4: 61-71.

Dail, H., F. Berman and H. Casanova, 2003. A decoupled scheduling approach for grid application development environments. J. Parallel Distrib. Comput., 63: 505-524.

Foster, I. and C. Kesselman, 2003. The Grid 2: Blueprint for a New Computing Infrastructure (The Elsevier Series in Grid Computing). 2nd Edn., Morgan Kaufmann, UK., ISBN: 1558609334.

Li, K., 2004. Experimental performance evaluation of job scheduling and processor allocation algorithms for grid computing on metacomputers. Proceedings of the 18th International Parallel and Distributed Processing Symposium, April 26-30, Santa Fe, New Mexico, pp: 170-177.

Muthuvelu, N., J. Liu, N.L. Soe, S. Venugopal, A. Sulistio and R. Buyya, 2005. A dynamic job grouping-based scheduling for deploying applications with fine-grained tasks on global grids. Proceedings of the 2005 Australasian workshop on Grid Computing and E-Research, January 30-February 4, Newcastle, Australia, pp: 41-48.

Shan, X.H., S. Xian-He and G.V. Laszewski, 2003. QoS guided min-min heuristic for grid task scheduling. J. Comput. Sci. Technol., 18: 442-451.