SCHEDULING FRAMEWORK FOR BANDWIDTH-AWARE JOB GROUPING-BASED SCHEDULING IN GRID COMPUTING

Ng Wai Keat, Ang Tan Fong, Ling Teck Chaw, Liew Chee Sun
Department of Computer System and Technology
Faculty of Computer Science and Information Technology
University of Malaya, 50603 Kuala Lumpur, Malaysia.
Email: waikeat@perdana.um.edu.my, { angtf, tchaw, csliew} @um.edu.my

ABSTRACT

In recent years, Grid computing has emerged as an evolution from the existing distributed computing systems for delivering information, resources and services to users. This new computational infrastructure offers a remarkable increase in the number of available computation capabilities that can be delivered to applications. Grid computing is increasingly being used for aggregating resources across different geographical places. Therefore, job scheduling in the Grid computing environment has posed new challenges which demand for better computing performance and well-improved utilization of resources. This paper investigates the use of bandwidth-awareness in a scheduling framework to enhance the performance of job scheduling.

Keywords: Grid Computing, Job Scheduling, Scheduling Framework

1.0 INTRODUCTION

Grid computing has become the upcoming approach for parallel and distributed computing that groups distributed resources for processing various applications. The term Grid is an analogy to a power Grid that provides consistent, pervasive, dependable, transparent access to electricity irrespective of its source [1]. The creation of large resources through pooling of individual resources from different locations enables the Grid to provide powerful computing infrastructure. The Grid computing concept has created various possibilities for performing large-scale applications in an easily manageable way. In order to ensure the efficiency and good performance of job scheduling, an effective scheduling mechanism has to be implemented to cater for the needs in the Grid environment by promoting better job match and well utilized resources.

Conventional parallel system models are connected to homogeneous computing nodes in a geographically small area network such as LAN [2]. Therefore, the consideration on the communication cost is insignificant. 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. Since the communication networks act as the basic foundation for realizing Grid computing, the networking area inherits several characteristics which influence the scheduling process. By taking the network condition into account, implementation of a scheduling mechanism built upon a scheduling framework will undoubtedly provide better job matches.

In this work, a framework is proposed to facilitate the scheduling of jobs using bandwidth-awareness and job grouping concept [3]. The bandwidth-aware scheduling is a part of the primary principle used in the scheduling framework. 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. The bandwidth-aware scheduling approach uses the network bottleneck bandwidth of resources to determine the priority of each resource. The job grouping approach is also used in the framework where the scheduler retrieves information of the resources' processing capability. Then, the scheduler selects the first resource and groups independent fine-grained jobs together based on the chosen resource's processing capability. These jobs are grouped in such a way 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. Next, these grouped jobs are sent to the corresponding resources and the results of the processing are sent back to the user after they have been computed at their respective resources.

The GridSim toolkit has been used to test the framework in a simulated Grid environment. The toolkit, a Java-based discrete-event Grid simulation package, provides the modelling and simulation of heterogeneous time and spaceshared Grid resources, applications, users, resource brokers and schedulers in the Grid environment [4] [5]. Furthermore, primitives for creating application tasks, mapping of tasks to resources and resource management are also supported. Implemented using the object-oriented framework, GridSim is selected to help study the scheduling behaviours of the proposed scheduling framework.

2.0 RELATED WORKS

In this section, two related scheduling approaches are introduced namely: Dynamic Job Grouping-Based Scheduling and Bandwidth-Aware Scheduling.

Dynamic Job Grouping-Based Scheduling strategy focuses on improving the scheduling of application with a large number of jobs that need small scale processing requirements. When small scale (fine-grained) jobs are sent to the resource individually, it will lead to high communication cost and time. The sending of an individual job involves transmission and processing overheads. Thus, the computation-communication ratio (CCR) for this type of execution tends to be low, leading to poor utilization of resources. Targeting on this drawback, the Dynamic Job Grouping-Based Scheduling strategy creates a batch mode scheduling for maximizing the utilization of resource processing capabilities and reducing the overhead for time and cost. In this strategy, user jobs are submitted to the scheduler and the scheduler collects the required characteristics of the available resources. It then selects a specific resource and multiplies the resource, 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. The scheduler groups the user jobs by accumulating the MI of each user job based on comparisons 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. Subsequently, a new job of accumulated total MI will be created with a unique ID and scheduled to be executed in the selected resource. This grouping process continues until all the jobs are put in groups and assigned to resources. When the grouping and assigning of jobs are completed, the scheduler sends the job groups to the matching resources for further computation. The Grid resources process the receiving grouped jobs and send back the results to the user through their I/O ports or queue when jobs are completely computed [6].

There has been extensive usage of network information in various algorithms or strategies as the main key for improving different procedures involving interconnectivity between two or more computers. The concept of considering the network conditions has been proven effective by various algorithms in avoiding latencies and delays occurring in the network. One example is the usage of bandwidth information in scheduling strategy for Web servers. Scheduling policies for the Web servers is an important part to improve the servers' response time. The performance of Web servers is basically affected by Internet traffic. A scheduling policy, called the Fastest Connection First (FCF) for Web servers that exploits the high variability, is used to reduce the delay. In this approach, two characteristics are observed. These characteristics are the speed of the connections and the response file size. FCF policy gives priority to the processes based on some characteristics of the specific request issued. The next process with the highest priority is chosen for the resource. The priority is based on the throughput of the connection and the number of bytes to be processed. The main concept is to give priority to the request whose connection can be finished earlier, that is, the smallest request issued through the fastest connection (best transmission rate or bandwidth). The FCF policy consists of two steps. The first step is the searching of request queue for the request with the shortest remaining number of bytes to process. The second step is searching the queue again for all requests whose size is smaller than the smallest multiplies some constant, β. With this information, the one with the highest estimated throughput is chosen. β defines the range of the second search and the trade-off between size and throughput. When β is large, priority is given to throughput; otherwise, priority is given to the remaining size of the request. The FCF policy aims to ensure that the socket buffers of the fast connections will always have data to send as soon as the ack (acknowledgement) confirmation arrives. In order to allow packets to be ready before sending, the server must give priority to the requests at the CPU, at the disk and at the network interface. During the time when the load on the server is light or moderate, the connection buffers usually have the data to send as the ack arrives, regardless of which scheduling policy is used. However, when the server is overloaded, giving priority to the small requests issued through fast connections will ensure that the buffers of these requests will have data to transfer when the ack arrive. In essence, through deployment of FCF policy, priority is given to the requests issued through fast connections at the server. As a result, these requests will be completed quickly and resources will be released for other requests waiting at the queue [7].

3.0 METHODOLOGY

There are several methods to study a system or a concept of an implementation. Generally, there are two main approaches. A system can be studied through experiments on the actual system or by conducting experiments on a model that represents the actual system [8]. Experiments that are conducted via a model can be further divided into two approaches. There are the physical model and the mathematical model. However, constructing a physical model may require more time and cost, therefore a mathematical model would be a sufficient approach for conducting experiments that require less cost and quicker results. A mathematical model can be studied through analytical solution or simulation. The simulation approach has been chosen for conducting analysis on the proposed scheduling strategy over analytical solution because simulation allows investigation of different situations in a controllable condition. On the other hand, the analytical solution that models a system through mathematical formulas may not provide the complete representation of the actual system.

Simulation is the process of designing a model of a real or imagined system and conducting experiments with that model [9]. The main motivation for simulation is to allow newly designed systems to be studied for various issues and to ensure they can perform effectively before the real development takes place. Simulation has been an effective method used in modelling real world processes, applications and objects for analysing important subjects such as feasibility, reliability, performance and behaviour. Sometimes, problems in the real world are complex and difficult to be modelled exactly. Hence, simulation could be the source for providing acceptable approximations for study purposes. Moreover, simulation is more practical and less expensive than building the actual system for evaluations and studies [10].

In the simulation of parallel and distributed systems, various important studies can be conducted in a controllable manner in which different scenarios can be formed according to their objectives. Furthermore, complete studies can be performed systematically and repeatedly without the need to change configurations of each machine manually in the real system. In addition to that, configurations in a simulated environment are easier to manage than the real system because in the real system, machines might be under different administrative controls which complicate the forming of desirable environments for further investigations. Simulation also allows loads in the connections and in the machines to be organized according to specific estimations or calculations. However, this could not be done in a real system because the loads in the system are ever changing according to time and it is difficult to control them to perform specific situations.

3.1 Simulation Model

A model is an abstraction of a system intended to replicate some properties of that system [11]. Modelling objective of simulation is vital for the collection of properties in the model. This is because a model cannot represent the actual system but it acts as an abstraction which captures the main system characteristics relevant for simulation purposes.

Simulation that represents processes can be modelled as entities. In the entity-based model simulation, each entity performs its own tasks and communicates with other entities through messaging. Simulation can also be represented in an event-based model. In event-based model, each task in a modelled process is invoked via triggering of events. It is possible for simulation to be implemented using both entity and event because both models are capable of replicating the real-world activities of the actual objects.

3.2 Simulation Environment

Simulation is a technique for performing experiments and evaluations on a system without the necessity to construct a true actual system, which is usually too expensive and impractical to build for every evaluation. Simulation is a simpler yet effective approach for analyzing and evaluating designed mechanisms, protocols, algorithms or theories for systems.

GridSim is a simulation toolkit developed by [12] that provides facilities for creating Grid scheduling simulation environment powered by event-driven discrete event simulation engine called SimJava. The GridSim platform is deployed in a multi-layer architecture and abstraction. The first layer contains the Java's scalable interface and runtime called the Java Virtual Machine (JVM). This Java interface can be implemented in single and multiprocessor systems. The second layer consists of a basic discrete event infrastructure called the SimJava which is built using the interfaces supplied by the first layer. The third layer is concerned with modelling and simulation of core Grid entities such as resources, application model, information services, uniform access interface and primitives application modelling and framework for higher level entities. These simulation facilities are supplied by the GridSim toolkit using the discrete event service provided by the lower-level infrastructure. The fourth layer is the simulation of resource allocation services called Grid resource brokers and schedulers. The top layer relates to modelling of application and resource with different scenarios of management policies, heuristics and algorithms.

4.0 SCHEDULING FRAMEWORK

The scheduling framework illustrated in Fig. 1 depicts the design of the job scheduler and its interactions with other entities. The job scheduler is a service that resides in a user machine. Therefore, when the user creates a list of jobs in the user machine, these jobs are sent to the job scheduler for scheduling arrangement. The job scheduler obtains information on available resources and their network information from the Grid Information Service (GIS), and the external network information service such as routing information gained from routers. Based on the information, the job scheduling algorithm is used to determine the job grouping and resource selection for grouped jobs. Once all the jobs are put into groups with selected resources, the grouped jobs are dispatched to their corresponding resources for computation.

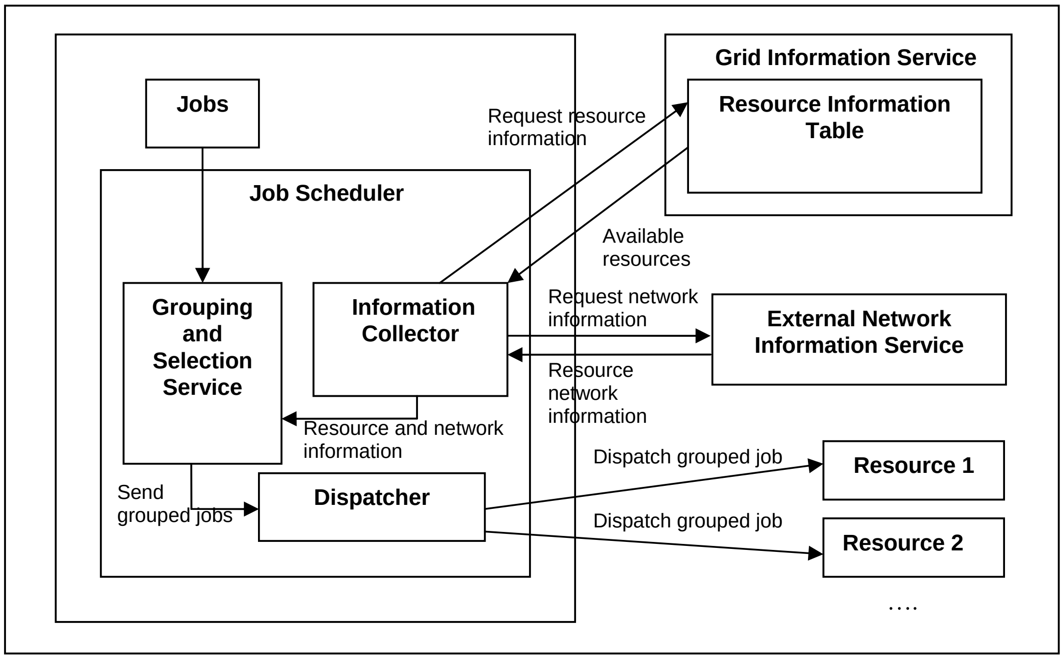
Grid Information Service
Resource Information Table
Jobs
Request resource information
Job Scheduler
Available resources
Grouping and Selection Service
Information Collector
Request network information
External Network Information Service
Resource network information
Resource and network information
Dispatch grouped job
Resource 1
Send grouped jobs
Dispatcher
Dispatch grouped job
Resource 2
….

Fig. 1: Bandwidth-aware Job Grouping-based Scheduling Framework

4.1 Grouping and Selection Service

The grouping and selection service serves as a site where matching of jobs is conducted. The strategy for matching jobs is based on the information gathered from the information collector. There are two steps involved during the matching of jobs i.e. job grouping and job selection. In the job grouping process, jobs submitted by the user to the scheduler are collected and they are grouped together based on the information of resources. The size of a grouped job depends on the processing requirement length expressed in Million Instructions. When a resource is chosen from the resource list, its processing capability expressed in the amount of Million Instructions processed in a second is used as the maximum limit for a grouped job. At the same time, job selection is also being conducted where a grouped job corresponds to the resource in question. The process is performed iteratively until all the jobs are grouped according to their respective resources.

4.2 Information Collector

The information collector collects information gathered from the Grid information service (GIS) and the external network information service. It assembles the resource availability and processing capability from the resource information table situated in the GIS. It also gathers information of the network bandwidth to reach each listed resource provided by the GIS from the network information service. The information collector acts as a repository for resources' information. It is used by the grouping and selection service to gather the necessary information to perform job selection via query or request for information.

4.3 Grid Information Service

The grid information service (GIS) is a facility that provides information about all the registered resources in a Grid. This service keeps track of all of the resources in the Grid. GIS collects resource information on operating system, system architecture, management policy and processing capability. In addition, it also provides information to users on the availability of the resources.

4.4 External Network Information Service

The external network information service serves as a hub for information on the network condition within the Grid. It gathers information on the network bandwidth of each resource. The information collected by this service is used by the grouping and selection service via the information collector to determine the scheduling of jobs.

4.5 Dispatcher

The dispatcher functions as a sender that sends grouped jobs to their respective resources. The dispatcher forwards the grouped jobs based on the schedule made during the matching of jobs with resources. The dispatcher also collects the results of the processed jobs from the resources through input ports.

5.0 SCHEDULING INTERACTIONS

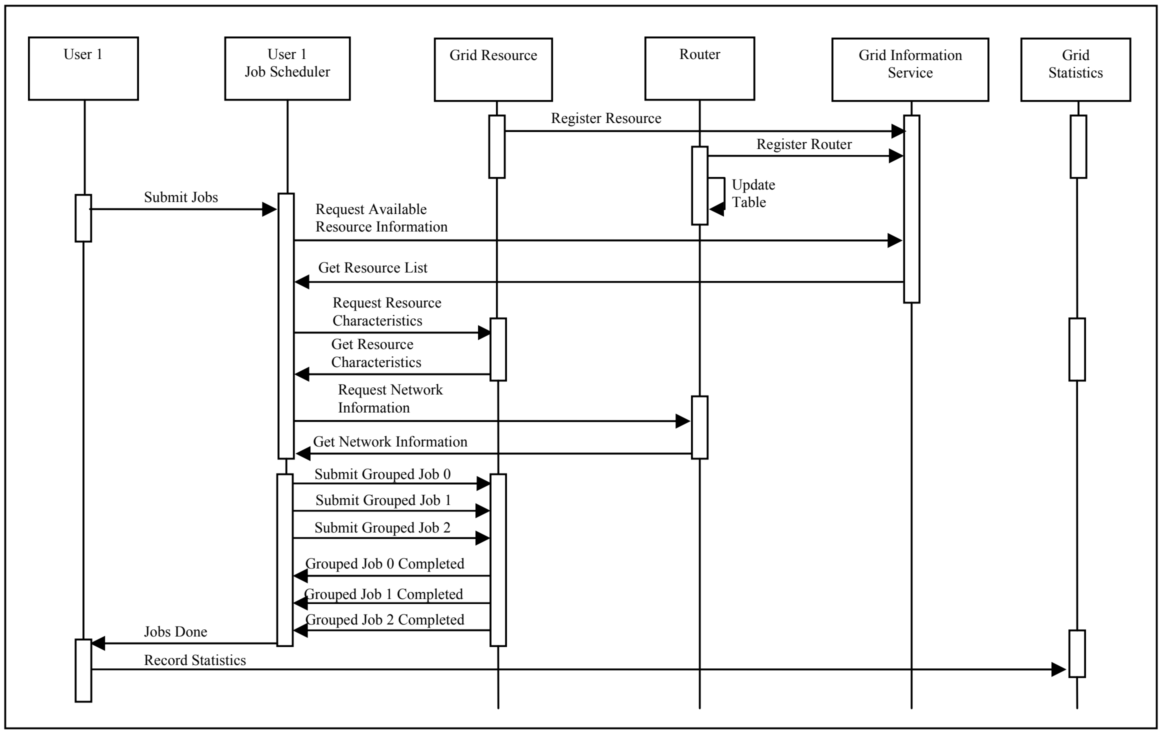

Fig. 2: Sequence Diagram for Bandwidth-aware Job Grouping-based Scheduling Framework

Fig. 2 depicts the sequence diagram for the Bandwidth-Aware Job Grouping-based Scheduling Framework. Before the beginning of the scheduling process, the resources and the routers register themselves to the Grid Information Service (GIS). Next, the routers perform updating of their routing information based on Open Shortest Path First (OSPF) with widest-shortest approach through the flooding mechanism. When the fine-grained jobs arrived at the job scheduler, the job scheduler performs query to the GIS to obtain information on available resources and requests for resource characteristics from the available resources. The job scheduler also acts as an OSPF listener that allows it to obtain routing information from the nearest OSPF router. The job scheduler carries out the job grouping and selection process. Subsequently, the grouped jobs are sent to their respective resources and these jobs are collected back after the resources have performed the job execution. Lastly, the job scheduling statistics are recorded for evaluation purposes.

6.0 SCHEDULING ACTIVITY

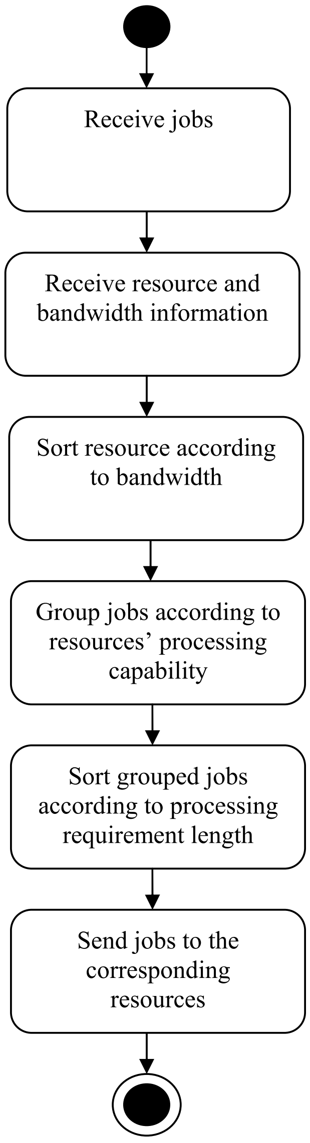
Receive jobs
Receive resource and bandwidth information
Sort resource according to bandwidth
Group jobs according to resources' processing capability
Sort grouped jobs according to processing requirement length
Send jobs to the corresponding resources

Fig. 3: Activity Diagram for Bandwidth-aware Job Grouping-based Scheduling Framework

Fig. 3 demonstrates the scheduling activity of the proposed framework. At the beginning of the scheduling activity, the scheduler receives jobs submitted by applications. After all the jobs are put into a queue, the scheduler performs a query to the Grid Information Service for information on the resources. Concurrently, information on the bandwidth to all available resources is gathered. Subsequently, the resources in a list are sorted according to the bandwidth in descending order. Jobs are grouped according to the processing capability of each selected resource. When all jobs are grouped, the grouped jobs are sorted based on processing requirement length in descending order. Lastly, the grouped jobs are sent to their respective resources.

7.0 SIMULATION AND DISCUSSION

A simulation environment with 5 Grid nodes as shown in Table 1 has been created to verify the improvement of proposed framework over other scheduling framework. Table 2 shows the simulation results of the proposed scheduling framework compared to a non bandwidth-aware job grouping list scheduling. The simulations involved jobs of processing requirement of average 200 Million Instructions (MI) along with a deviation percentage of 20 and granularity size of 10 simulation seconds. Fig. 4 depicts the graph of the results collected from the simulations. The bandwidth-aware job grouping-based scheduling has reduce the total processing time by 9% - 77% when compared to non bandwidth-aware job grouping list scheduling depending on the number of jobs. The graph shows that the

proposed scheduling framework is able to operate in conditions of different number of jobs with good improvement over an existing scheduling framework.

Table 1: Grid Nodes

Name	Million instructions per second (MIPS)
R1	1050
R2	684
R3	1050
R4	1050
R5	1050

Table 2: Simulations on Different Number of Jobs

Number of Jobs/Gridlets	Bandwidth-aware Job Grouping			Non Bandwidth-aware Job Grouping
	Number of Groups	Processing Time (sim sec)	MIPS availability (Total MIPS* granularity size)	Processing Time (sim sec)
25	1	27.930	10500	122.280
50	1	56.113	10500	136.680
75	2	154.610	21000	240.900
100	2	211.720	21000	333.800
125	3	271.090	31500	420.400
150	3	325.410	31500	436.230
175	4	367.660	42000	409.370
200	4	385.630	42000	424.090

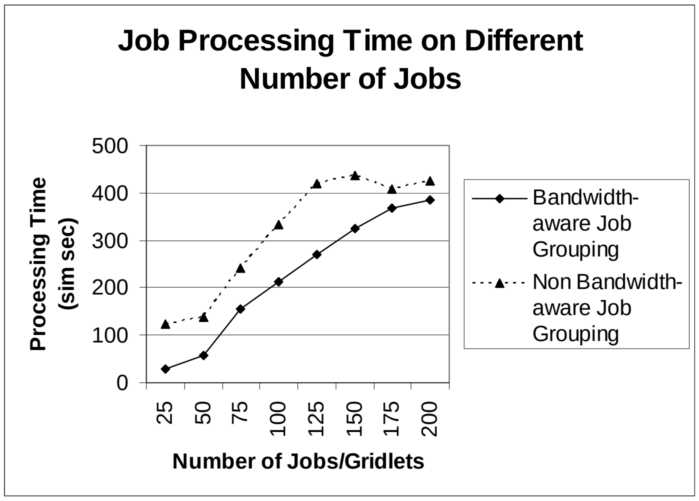

Fig. 4: Processing Time on Different Number of Jobs

Table 3 shows the simulation results where the number of jobs is 150 along with a deviation percentage of 20, and the granularity size is 10 simulation seconds. Fig. 5 depicts the graph of the results collected from the simulations. The bandwidth-aware job grouping-based scheduling has reduce the total processing time by 19% - 81% when compared to non bandwidth-aware job grouping list scheduling depending on the average MI. Again, the result shows that the proposed scheduling framework is able to perform against different job processing requirement lengths with better scheduling performance over a non bandwidth-aware job group list scheduling.

Table 3: Simulations on Different Job Processing Requirement Lengths

Average MI	**Bandwidth-aware Job Grouping**			**Non-Bandwidth-aware Job Grouping**
	Number of Groups	**Processing Time (sim sec)**	**MIPS availability (Total MIPS * granularity size)**	**Processing Time (sim sec)**
25	1	22.404	10500	117.970
50	1	46.680	10500	118.450
75	2	125.290	21000	301.640
100	2	127.590	21000	341.360
125	2	205.890	21000	337.990
150	3	250.940	31500	323.390
175	3	291.570	31500	374.850
200	3	330.810	31500	413.320

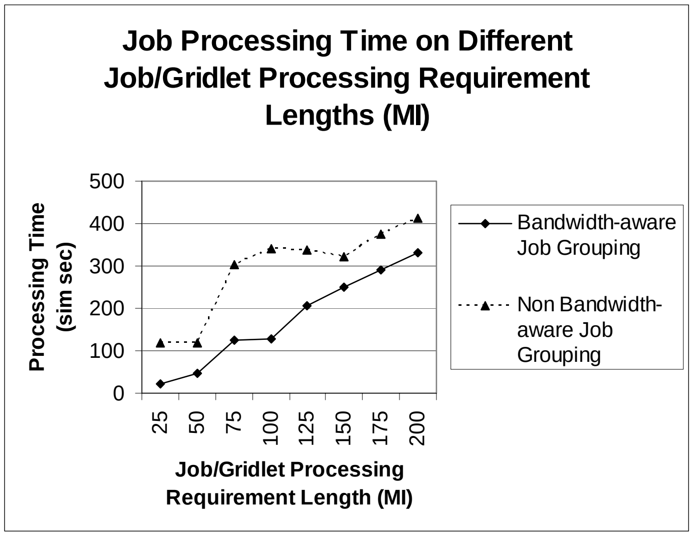

Fig. 5: Processing Time on Different Processing Requirement Lengths

8.0 CONCLUSION

In this paper, a framework for bandwidth-aware grouping-based scheduling 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.

Possible future work of this framework includes integrating the grouping criteria to undertake the bandwidth and processing capability together which may produce as a ratio of computation and communication without using different stages of selection.

REFERENCES

[1] M. Baker, R. Buyya, D. Laforenza, "Grids and Grid Technologies for Wide-area Distributed Computing". *Software—Practice & Experience*, Vol 32, No. 15, 2002, pp. 1437 – 1466.

[2] S. You, H. Kim, D. Hwang, S. Kim, "Task Scheduling Algorithm in GRID Considering Heterogeneous Environment", in *The 2004 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'04)*, Monte Carlo Resort, Las Vegas, Nevada, USA, June 21 - 24, 2004, pp. 240-245.

[3] W. K. Ng, *A Bandwidth-aware Job Grouping-based Scheduling on Grid Environment*, M.Comp.Sc. Dissertation, University of Malaya, 2005.

[4] R. Buyya, Manzur Murshed, "GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing". *Concurrency Computation: Practice and Experience*, Vol. 14, No. 13-15, 2002, pp. 1507-1542.

[5] Manzur Murshed, R. Buyya, "Using the GridSim Toolkit for Enabling Grid Computing Education", in International Conference on Communication Networks and Distributed Systems Modeling and Simulation *(CNDS 2002)*, San Antonio, Texas, USA, January 27-31, 2002.

[6] N. Muthuvelu, J. Liu, N. L. Soe, S. Venugopal, A. Sulistio, R. Buyya, "A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids", in *Proceedings of the 3rd*

Australasian Workshop on Grid Computing and e-Research (AusGrid 2005), Newcastle, Australia, Jan 30 - Feb 4, 2005, pp. 41-48.

[7] C.D. Murta, T. P. Corlassoli, "Fastest Connection First: A New Scheduling Policy for Web Servers", in *Proceedings International WWW Conference(11),* Honolulu, Hawaii, USA, *May 7-11,* 2002.

[8] A. M. Law, W. D. Kelton*, Simulation Modeling and Analysis*. 3rd ed., US, McGraw-Hill, 2000.

[9] D. R. Smith, Simulation: The Engine Behind the Virtual World, eMatter, 1999.

[10] A. Sulistio, C. S. Yeo, R. Buyya, "A Taxonomy of Computer-Based Simulations and Its Mapping to Parallel and Distributed Systems Simulation Tools". *Software-Practice & Experience*, Vol 34, No. 7, 2004, pp. 653-673.

[11] C. M. Overstreet, Model Specification and Analysis for Discrete Event Simulation. PhD Dissertation, Dept. of Comp. Sc., Virginia Tech, 1982.

[12] R. Buyya, Economic-based Distributed Resource Management and Scheduling for Grid Computing. PhD Dissertation, Monash University, 2002.

BIOGRAPHY

Ng Wai Keat obtained his B. Comp. Sc. and M. Comp. Sc. (by research) in 2003 and 2005 from University of Malaya. Currently, he is a researcher at MIMOS. His research areas include grid computing, parallel computing and web services.

Ang Tan Fong obtained his B. IT and M. Comp. Sc. in 2000 and 2001 from University of Malaya. He is a lecturer at the Faculty of Computer Science & Information Technology, University of Malaya, Malaysia. His research areas include web services, Voice over IP (VoIP) and grid computing.

Ling Teck Chaw obtained his M. Comp. Sc. and PhD in 1996 and 2005 from University of Malaya. He is a lecturer at the Faculty of Computer Science & Information Technology, University of Malaya, Malaysia. His research areas include Core network research, Inter-domain QoS, VoIP, Grid computing and Network security.

Liew Chee Sun is a lecturer at the Faculty of Computer Science & Information Technology, University of Malaya, Malaysia. He obtained his M. Comp. Sc. in 2002 from Universiti Sains Malaysia. Prior to working at University of Malaya, he was the core researcher of the Grid Computing Research Lab in USM. His current research interests include Grid computing, Peer-to-Peer computing and VoIP.