Web-based Dynamic Scheduling Platform for Grid Computing

Oh-han Kang, and Sang-seong Kang ,

Dept. of Computer Education, Andong National University, South Korea

Summary
In this paper, we designed and presented a web-based grid scheduling platform, which can model a system and simulate a scheduling method in grid computing. The presented web-based scheduling platform utilized GridSim, a grid scheduling toolkit in Java environment, as a tool for stimulation and is able to perform resource modeling, operation modeling(task modeling), algorithm compiling, simulation, and performance evaluation rapidly in a web environment. A constructed scheduling platform can be applied as a foundation for grid research and can be used to analyze the efficiency of the scheduling algorithm.

Key words:
Grid computing, Scheduling algorithm, Communication cost, Simulation, Consistency, Cluster.

1. Introduction

Grid computing has recently caught attention as a newly efficient paradigm for future parallel and distributed computing to aggregate diverse and geographically distributed high quality systems [1]. In a grid environment, various resources with disparate efficiency are distributed geographically and connected through the internet. Also they are managed by application programs, which require a huge operational ability. To maximize the capacity of systems in a grid environment, the scheduler, which readily understands the users' demand and the abilities of feasible resources so as to manage application programs efficiently, is needed. In a grid environment, characteristics such as the variety of application programs and heterogeneity of resources make the amount of required time unpredictable.

To enhance the quality of grid system with such characteristics, the scheduling method should reflect these unique qualities of the grid system.

To analyze the detailed motion and capacity of the task scheduling algorithm for a grid system, these algorithms

must be applied to an actually constructed grid system to analyze the log and evaluate performance time. However, resources composing the grid are located at relatively distant sites, retain disparate features, and might not be managed according to time and place. Moreover, because results for various grid environments are required to verify the efficiency of scheduling algorithm for typical grid environment, analysis of the algorithm using the real grid environment is currently difficult.

To solve this kind of problem, the grid system, the user's demand, and the characteristics of the application program should be designed. Moreover, the grid platform, which can simulate the scheduling algorithm should go through several researches and developments. Application programs which are processed in the grid environment possess various kinds of characteristics with regards to the operational amount, user's demand, type of communication, and ratio of input-output. These application programs with disparate characteristics require different applications of the scheduling method. Henceforth, various scheduling algorithms which reflect features of grid-system and application program should be developed, and a grid platform based scheduling tool should be invented to analyze and compare the efficiency of scheduling algorithms.

Several approaches have been proposed to schedule applications on grid computing environments [2-4]. For the analysis of the functional condition and performance of operational scheduling algorithm, the method of

Manuscript received May 5, 2006.
Manuscript revised May 25 , 2006.

modeling resources and operational grid group, and sequentially being practiced by software as if the real system operates is used as an alternative. Although simulation can easily reflect diverse features of the grid environment and simply confirm results and analysis, the exclusion of a variable which influences the grid-environment can result in a distorted outcome. Generally, the grid environment, operation, and user must be taken into consideration in a grid scheduling simulation.

In a grid environment, the number of resources, the strategies to distribute a processor of each resource, the transmission speed of the communication web of each resource, the number of node(system) composing each resource, the number of processors in each node, and the ability to manage a processor for each node should be taken into consideration. Also, regarding matters related to the operation, the number of tasks, the amount of processing time, the amount of input data for each task, the amount of output data for each task, the number of demanded processors for each task, and the cause and effect for each task should be regarded. Moreover, for contents related to the users, the number of users and the speed of the user's communication web should be reflected.

The contents of this paper will follow in this order. In chapter 2, the characteristics of a scheduling toolkit which can be used in a grid computing environment are introduced as related researches, and our unique embodiment of a scheduling platform is suggested in chapter 3. And for chapter 4, we will show applied examples of a practicing scheduling platform, and will sum up with a conclusion in chapter 5.

2. Related Works

There are several tools that are support applications for a scheduling simulation in grid computing environments [5-9]. While in the United States, researches concerning grid toolkit are conducted in SDSC(San Diego Supercomputer Center) and GDC(Grid Computing Group) of Virginia University, GRIDS research center of Melbourne promotes active researches in Australia. SDSC developed a toolkit such as MicroGrid[5], SimGrid[6], and TeraGrid in a joint effort with UC San Diego's research center, namely CSAG(Concurrent Systems Architecture Group). Australia's GRIDS research center developed GridSim[7] and fosters universal usage of its program in other organizations by continuously expanding its function.

To simulate the scheduling algorithm with the application of current grid scheduling toolkit, several procedures such as the establishment of an appropriate development environment, the analysis of the source code, the programming for resource modeling and operational modeling, and the practice of the scheduling algorithm are required. These procedures incite repetitive operations for researchers and diminish the efficiency of the scheduling algorithm. Also, Australia's GRIDS research center developed Visual Modeler(VM)[10], a GUI-based modeling tool, and also enabled simple drawing up of source codes in resource modeling and application modeling, which the research center developed for GridSim[7]. However, VM merely facilitated the modeling procedures, and showed no improvements in the functions such as simulation, analysis of capacity, the storage and management of resource and application data.

In this paper, we suggested and actualized Web-based Grid Scheduling Platform (WGridSP), which designed a system, and simulated a scheduling method in a grid computing environment. The actualization of WGridSP enables, with the help of a grid computing environment, the modeling of scheduling algorithm and simulation. WGridSP is composed of ①testbed manager, ②application manager, ③algorithm compiler, ④scheduling simulation, and ⑤efficiency-analyzing

module. WGridSP can carry out resource modeling, task modeling, algorithm compiling, simulation, and rapid evaluation in a web environment. Therefore, WGridSP furnishes researchers with the environment where they can concentrate on the actualization of algorithm and the analysis of results.

The comparison of GridSim[7], a typical Java-based scheduling toolkit, and WGridSP is illustrated in Table 1.

Table 1: Comparison of GridSim and WGridSP

		GridSim	WGridSP
Compile/Execution environment		Mandatory	Optional
Toolkit structure analysis		Mandatory	Optional
Resource/ Application modeling	tool	Text editor	Web Browser
	style	Java source	Database
	compile	Compiling at Shell	Optional
Implementation of scheduling algorithm	tool	Text editor	Web Browser
	style	Java source	Core algorithm
	compile	Shell	Server
Simulation	execution	Shell	Web
	result	Console and log file	Web page
Performance evaluation	tool	Text editor	Web Browser
	exection	Shell	Web
	parameter	Source code	Web input
	result	Console and log file	Web page, Graph

3. The Structure of Web-based Grid Scheduling Platform

WGridSP has utilized GridSim, a Java-based grid scheduling toolkit as a simulation tool. GridSim, through its choice of various variables in global grid environment and the selection of SimJava[10] package as a foundation which is frequently preferred in the simulation of distributed environment, succeeds in becoming the grid scheduling toolkit with high practicality and stability. WGridSP not only functions as an interface between user and GridSim by the means of the web, but also makes recycling possible with the management of resource modeling, application modeling materials and the scheduling algorithm. Fig. 1 shows the relationship between WGridSP and GridSim toolkit. WGridSP is designed as a separate additional component residing in a layer above the GridSim toolkit. WGridSP is composed of four functional units, as demonstrated in Fig. 2.

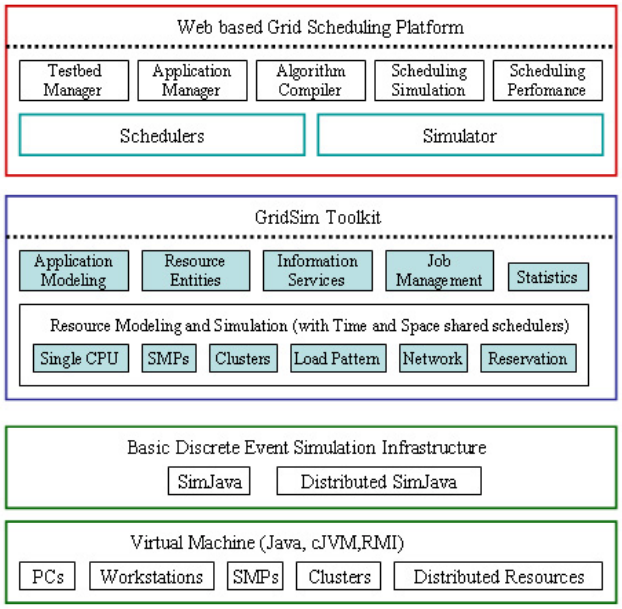

Fig. 1 Relationship between WGridSP and GridSim toolkit.

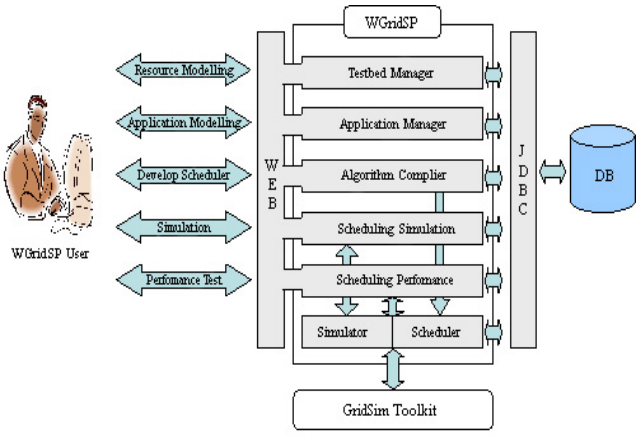

Fig. 2 WGridSP architecture.

3.1 Testbed manager

Testbed manager allows users to conduct resource modeling easily and rapidly. Especially, resource information planned by users is stored in the database through the management of each testbed, and used for algorithm simulation and performance evaluation. Testbed is composed of one or more resources, which, in turn, is composed of machines. These machines are composed of more than one processor. Generally, resources composed of more than one machine are regarded as the "Cluster system". Properties assigned to WGridSP's resources are illustrated below.

① Strategy for assigning processors(policies): It is the property determining how to assign processors when the number of processors occupied by the resource is less than the number of operations waiting to be processed. In WGridSP, one can choose between the time sharing method, which allocates and conducts more than two works to the specific processor concurrently and the space sharing method, which processes later work after earlier assigned work is finished.

② Transmission speed(baud rate): It is a transmission speed of which a specific resource is connected to the network. The size of the input data necessary to process the operation influences the size of the outcome data after processing and the amount of time required for the process

③ Cost: Defining the cost of a specific resource can be utilized in the scheduling algorithm which considers cost.

④ Processor conducting ability: It indicates the ability to process an operation and is expressed in MIPS units. It influences the length of the operation and the amount of time required for processing.

⑤ The number of processors(CPUs): It indicates the number of processors possessed by the machine. More processors enable a greater number of operations that can be managed.

Fig. 3 is an E-R(Entity-Relation) diagram, indicating the structure of the database for a testbed manager.

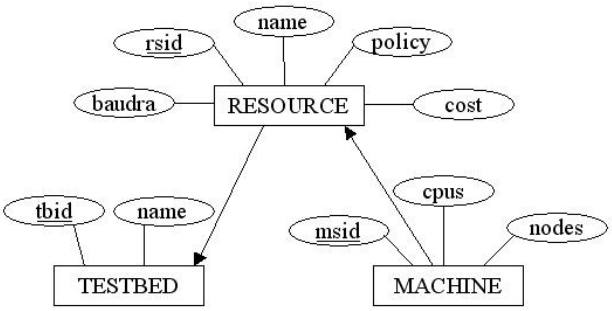

Fig. 3 Database architecture of testbed manager.

3.2 Application manager

The application manager supports modeling of application operations. The application information designed by a user is stored in a database and, thereafter, is used for simulation.

Each application is composed of one or more tasks and these tasks retain the following properties.

① Length: The task's required amount of time of CPU. Bigger values indicate a longer amount of processing time

② Input size: It indicates the size of the data needed to process the task. Because the data need to be transmitted to an assigned resource, input size affects the amount of time needed to transmit the task to resources.

③ Output size: It indicates the size of the data produced by resources after the processing task. Also, it influences the required time needed to transmit results from resources to scheduler.

3.3 Management of scheduling algorithm

WGridSP demands the core part of scheduling algorithm. The source code which garners necessary grid information is produced automatically, immediately before compiling, and these compiling results are transmitted to users to fix the possible errors. Fig. 4 is the flowchart showing the scheduling algorithm's process of compiling and storing.

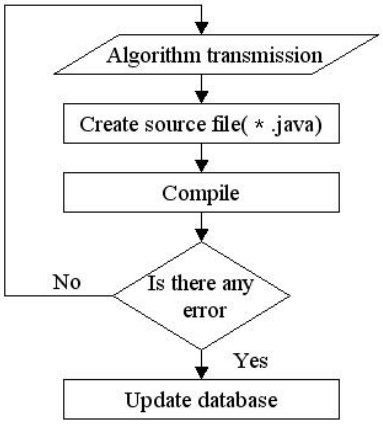

Fig. 4 Flowchart for database update.

3.4 Scheduling simulation

The selection of the testbed, application, algorithm required for scheduling simulation can be confirmed through the web-browser. By validating which resources of Gridlet are assigned to which period of time, the analysis of a detailed operational situation is possible. Fig. 5 is the pseudo-code of the detailed action of the simulator, which is the main class in simulation.

3.5 Evaluation of scheduling algorithm

In the evaluation of the scheduling algorithm, the testbed for evaluation, the property limits of the task, and the choice over more than one scheduling algorithm can verify the change in the time needed to finish the work according to the amount of work. The performance time and the size of input and output data produce a random number in uniform distribution or normal distribution. As a result, to obtain the exact measured value, the average value related to the designated number of repetitions should be acquired, to eventually get the final performance time. Fig. 6 is pseudo-code indicating the management process of the evaluation module.

```
Load resource and task informations from database;
GridSim initialize;
for i := 1 to ResourceList.size do
    Create a new resource;
    for j := 1 to MachineList[i].size do
        Create a new machine;
        for k := 1 to MachineList[i].cpus do
            Crate a new processor;
            Add the processor to created machine;
        endfor
        Add the machine to created resource;
    endfor
endfor
for i := 1 to TaskList.size do
    Create a new task;
endfor
Create the scheduler object requested by user;
Start GridSim simulation;
Send the simulation result to user;
```

Fig. 5 Pseudo-code for scheduling simulation.

4. Experiments and Results

As illustrated in this paper, the usages of WGridSP to simulate grid scheduling are shown in the following examples.

4.1 Resource modeling

Resource modeling, through the assumption of a grid system in reality, defines the environment for scheduling simulation. Resource modeling enables storage, modification, and deletion in the unit of individual testbeds. The fig. 7 is the definition of the testbed, which is composed of four resources, from R0 to R3. The resource R0 is a system, equipped with four processors; their communication speed is 100, the cost per second is 8, and the capacity is 515MIPS. R3 is the hypothetic model of a cluster system with six nodes.

```
Create Resources with loaded resource information from database;
makespan_sum := 0;
for i := TaskNum_Min to TaskNum_Max step TaskNum_Step do
    for loop :=1 to NoOfSimulation do
        Create a new resource;
        for j :=1 to i do
            length := random(from Length_Min to Length_Max);
            in := random(from Input_Min to Input_Max);
            out := random(from Output_Min to Output_Max);
            Crate a new task with length, in, out;
        endfor
        Start GridSim Simulation;
        time := result of simulation;
        makespan_sum := makespan_sum + time
    endfor
    makespan := makespan / NoOfSimulation;
    Draw the chart with makespan;
endfor
```

Fig. 6 Pseudo-code for evaluation of scheduling algorithms.

4.2 Application modeling

Application modeling is merely designed for the function of simulation. This defines not only a minority of users with distinct characteristics, but also the tasks belonging to the users. The fig. 8 indicates the application, in which three, four, and one tasks are affiliated with three users, from User0 to User2, respectively. Due to the connection to the communication speed of 100 and its zero delay time, User0 can execute a task, just after the simulation begins. User1 and User2 are connected to the communication speed of 150 and 50 respectively, and after 30 and 50 each, resources are assigned to user-owned tasks based on a scheduling algorithm.

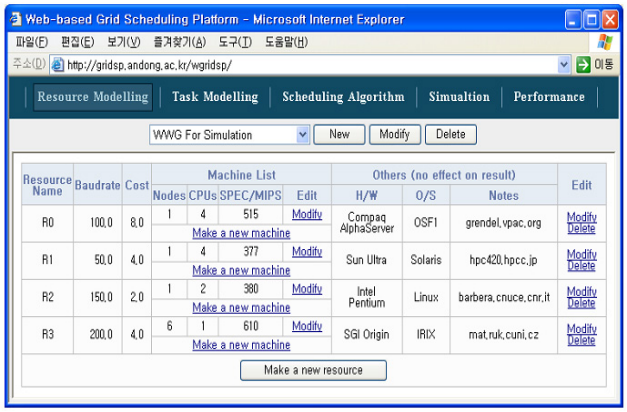

Fig. 7 The example of testbed modeling.

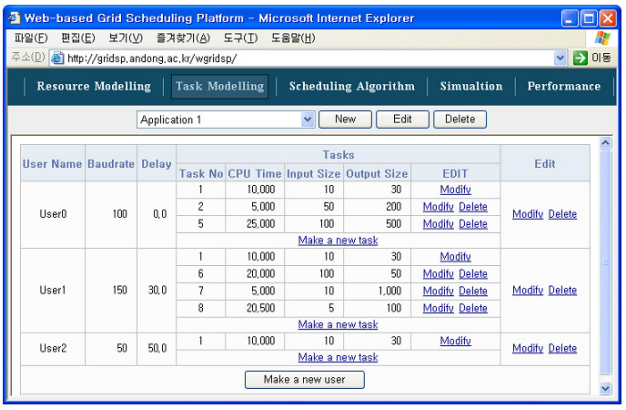

Fig. 8 The example of application modeling.

4.3 Algorithm compiling

Algorithm compiler takes a role in compiling java source-code written by a developer. All scheduling algorithms

altogether inherit Broker class. Within Broker class, a scheduleAdviser() method should be assigned as abstract, and must be defined, and major algorithms can be composed in this method. The fig. 9 is a simple algorithm, in which tasks in glUnfinishedList are allocated to randomly selected resources. Because of random selection, dynamic information of each resource is unnecessary. As a result, the value for required Dynamic Info_ is set to false.

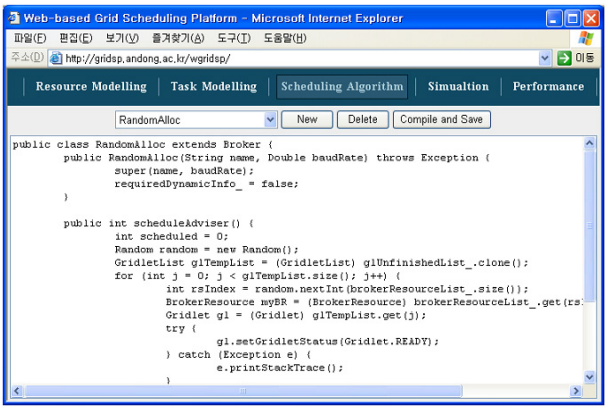

Fig. 9 Compiling of scheduling algorithm.

4.4 Scheduling simulation

The fig. 10 is the result for simulation, by the application of an algorithm, which arbitrarily assigns the application named, "Application I" to the testbed called, "WWG For Simulation." The simulation result lets the observation of tasks by not only all the users at a single glance, but the tasks assigned to specific users as well. While the part shaded in gray in the figure indicates the case, which users request for resource information in the grid and receives it, the part shaded in colors(darker ones) shows the process, in which particular resources are transmitted for execution and, thereafter, results are returned. Placing a mouse pointer on each chart enables the in-depth confirmation of execution time in each task.

4.5 Performance evaluation

The fig. 11 is the comparison between capacities of two scheduling algorithms. While 'RandomAlloc' is a method in which tasks are randomly assigned to resources, 'TimeOpt_In_EconomicModel' algorithm is a modification of time-optimal algorithms in EconomicModel[2] to adjust to WGridSP environment. The comparison of two algorithms shows that when the length of tasks is less than approximately 25,000, algorithms of random selection shows superior ability. If it is more than 25,000, it shows opposite results. This tendency is largely associated with the overheads in 'TimeOpt_In_EconomicModel,' which requires the dynamic information to be sent when the length of tasks is short.

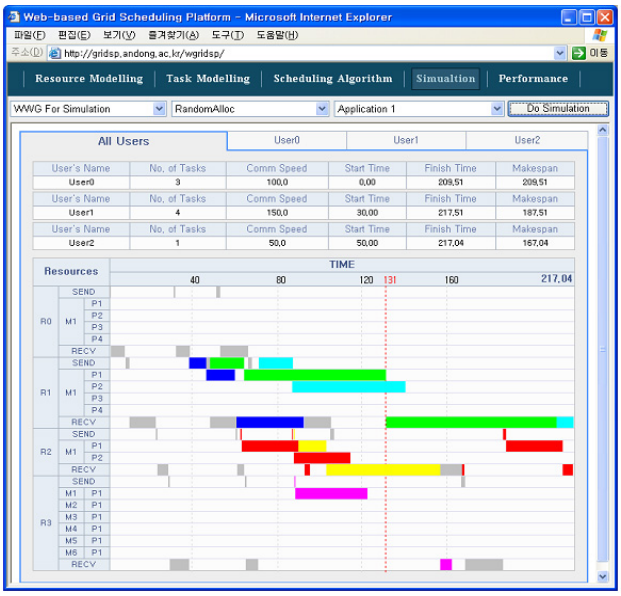

Fig. 10 Simulation results.

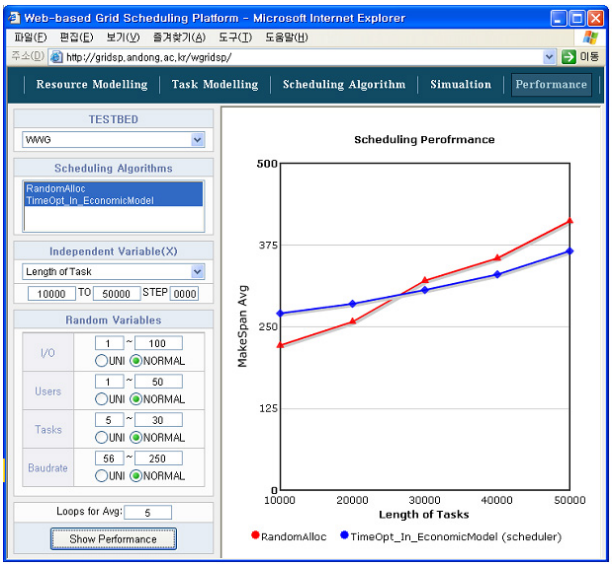

Fig. 11 Performance evaluation for different length of tasks.

The fig. 12 and fig. 13 show the change in completion time in accordance with the length of tasks by dividing the task numbers to three parts, 20, 40, and 60.

'TimeOpt_In_EconomicModel' is less influenced by the number of tasks than 'RandomAlloc,' and also the overall performance time is shorter for 'TimeOpt_In_EconomicModel.

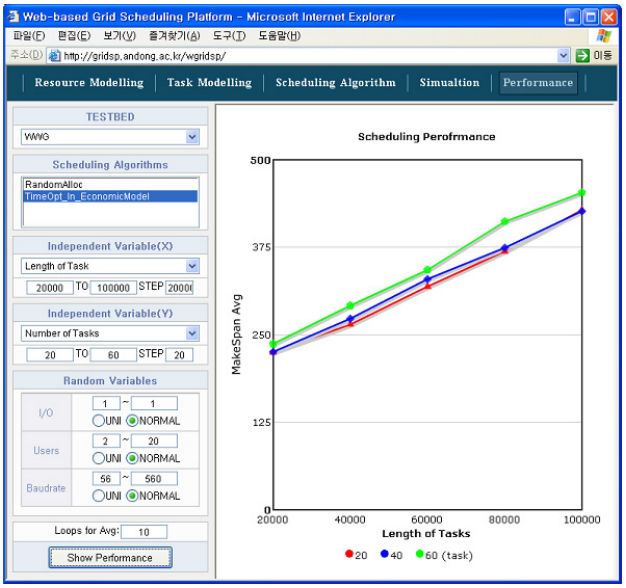

Fig. 12 Performance evaluation for different task numbers.(TimeOpt_In_EconomicModel)

5. Conclusion

Because WGridSP can simulate dynamic grid scheduling in web environment, lots of time and effort can be saved. However, the realization of functions to support existing algorithms can create unexpected technical difficulties in the process for developing new algorithms. Henceforth, diverse algorithms should be created through utilizing WGridSP, and new functions, in turn, should be added to accommodate various algorithms.

Because various kinds of application programs can be processed in a grid system, the grid platform proposed in this paper can be employed in a number of experiments, in order to maximize the capacity of grid system. WGridSP, from now, can be utilized as the basic structure for grid researches. Especially WGridSP is expected to take its significant role as a standard tool, which is needed in researches for the scheduling methods of application programs.

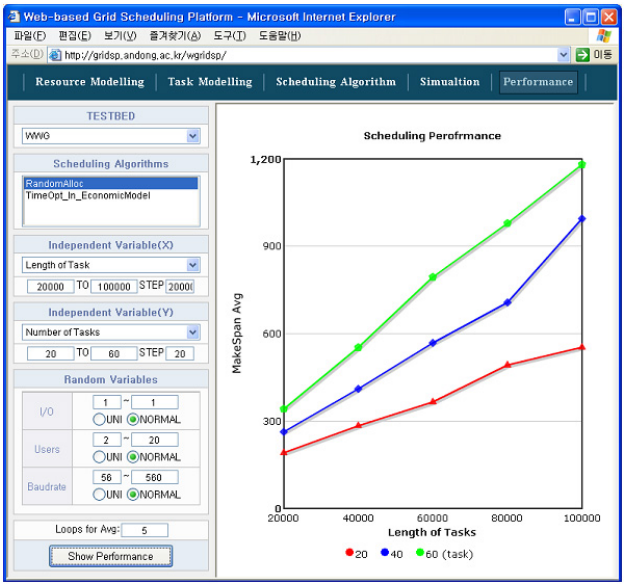

Fig. 13 Performance evaluation for different task numbers.(RandomAlloc)

Acknowledgments

This work was supported by grant No. B1220-0501-0048 from the University Fundamental Research Program of the Ministry of Information & Communication in Republic of Korea.

Reference

[1] Foster and C. Kesselman, "The Grid: Blueprint for a Future Computing Infrastructure," Morgan Kaufmann Publishers, USA, 1999.

[2] Rajkumar Buyya, "Economic-based Distributed Resource Management and Scheduling for Grid Computing," Ph. D. Thesis, Monash University, Melbourne, Australia, 2002.

[3] Srikumar Venugopal and Rajkumar Buyya, A Deadline and Budget Constrained Scheduling Algorithm for eScience Applications on Data Grids, 6th International Conference on Algorithms and Architectures for Parallel Processing, pp. 60-72, 2005.

[4] L. Lie, J. Zhan, and L. Li, A Runtime Scheduling Approach with Respect to Job Parallelism for Computational Grid, Pro. of 3rd International Conference on Grid and Cooperative Computing, pp. 261-268, 2004.

[5] H. Song, et. al., "The MicroGrid: A Scientific Tool for Modeling Computational Grids," Proc. of IEEE Supercomputing(SC 2000), Nov. 2000.

[6] H. Casanova, "Simgrid: A Toolkit for the Simulation of Application Scheduling," Proc. of the 1st IEEE/ACM International Symposium on Cluster Computing and the Grid(CCGrid 2001), May 2001.

[7] R. Buyya, and M. Murshed, "GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing", The Journal of Concurrency and Computation, Vol. 14, pp. 1175-1220, 2002.

[8] K. Aida, et. al., "Performance Evaluation Model for Scheduling in a Global Computing System". The International Journal of High Performance Computing Applications, Vol. 14, No. 3, Saga Publications, 2000.

[9] Michael Walker. "A Framework for Scheduling Data-Parallel Applications in Grid Systems". MS Thesis, University of Virginia, 2001.

[10] Anthony Sulistio, Chee Shin Yeo, and Rajkumar Buyya, "Visual Modeler for Grid Modeling and Simulation (GridSim) Toolkit", ICCS 2003, LNCS 2659, pp 1123-1132, 2003.

[11] F. Howell and R. McNab, "SimJava: A Discrete Event Simulation Package for Java with Applications in Computer Systems Modelling", Proc. of 1st Int. Conference on Web-based Modelling and Simulation, Society for Computer Simulation, 1998.

[12] Nithiapidary Muthuvelu, Junyang Liu, Nay Lin Soe, Srikumar Venugopal, Anthony Sulistio and Rajkumar Buyya, "A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids", Australasian Workshop on Grid Computing and e-Research (AusGrid2005), Newcastle, Australia. Conferences and Practice in Information Technology, Vol 44, 2005.

Oh-han Kang received the B.S. degree in Electrical Engineering from Kyungbook National University, Korea, in 1982 and M.S. and Ph.D. degrees in Computer Science from the Korea Advanced Institute of Science and Technology in 1982 and 1992, respectively. He was a visiting scholar at the Center for Distributed and Mobile Computing at the University of Cincinnati in 1999. He worked as a senior engineer for Qnix Computer Corp., from 1984 to 1993. He has been a professor in the Department. of Computer Education, Andong National University, Korea.

Sang-seong Kang received the B.S. degree in Computer in 1998 and M.S. degree in Educational Engineering from Andong National University in 2001. He has been a lecturer at Andong National University, Korea.