Jurnal
Teknologi

Full Paper

A CRITICAL ANALYSIS OF SIMULATORS IN GRID

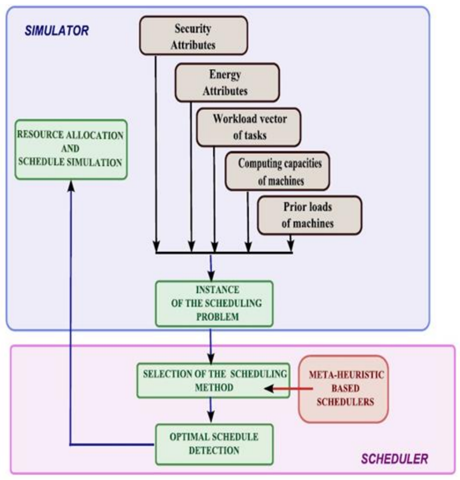
Omar Dakkak, Suki Arif*,Shahrudin Awang Nor

InterNetWorks Research Laboratory, School of Computing, Universiti Utara Malaysia, 06010 UUM Sintok, Kedah Darulaman, Malaysia

Article history
Received
02 June 2015
Received in revised form
09 August 2015
Accepted
1 September 2015

*Corresponding author
suki1207@uum.edu.my

Graphical abstract


SIMULATOR
Security Attributes
Energy Attributes
Workload vector of tasks
Computing capacities of machines
Prior loads of machines
RESOURCE ALLOCATION AND SCHEDULE SIMULATION
INSTANCE OF THE SCHEDULING PROBLEM
SELECTION OF THE SCHEDULING METHOD
META-HEURISTIC BASED SCHEDULERS
OPTIMAL SCHEDULE DETECTION
SCHEDULER

Abstract

In parallel and distributed computing environment such as "The Grid", anticipating the behavior of the resources and tasks based on certain scheduling algorithm is a great challenging. Thus, studying and improving these types of environments becomes very difficult. Out of this, the developers have spent remarkable efforts to come up with simulators which facilitate the studies in this domain. In addition, these simulators have a significant role in enhancing and proposing many scheduling algorithms, and this in turn has reflected efficiently on the Grid. In this paper, we will present some of these tools, which are: GridSim for large scales distributed computing and parallel environment, Alea for tackling dynamic scheduling problems, Sim-G-Batch grid simulator for simulating the security and energy concept and Balls simulator for evaluation peer-to-peer with integrated load balancing algorithm. Furthermore, this paper aims to guide and assist the researcher to choose the proper tool that can fit the studied research area, by providing functionality analysis for the reviewed simulators..

Keywords: Grid, simulation, simulator, evaluation

1.0 INTRODUCTION

Simulation approach is widely used in science, such as: engineering fields and computer science. Simulation refers to design a virtual framework that simulates an actual one in real life. This framework could represent a network topology, a mechanism or a device. This virtual framework is built using programming codes which are executed through software (simulator) that runs on a computer. The main advantage of simulation approach that it can test the performance repeatedly without any consequences [1, 2]. In addition to simulation, analytical modeling and measurement are two other approaches that used to evaluate the performance [3, 4]. Simulation provides a medial solution regarding the cost, time required, difficulty level, trade-off evaluation and accuracy, comparing with analytical modeling and measurement.

In a large scale cyber network such as "Grid", simulation is the preferable approach in order to test and analyze the performance of the scheduling algorithms. Moreover, simulation can bypass the challenges that related to highly parameterized problems. In this type of networks, mostly the number of jobs and resources is massive and the environment is complex. Therefore, to obtain better and expressive results, the experiments must be repeated several times. In addition to the above, it's extremely hard to get
smooth access to available and ready grid platforms to evaluate the algorithms due to both resources and user's tasks keep changing profusely [5]. Thus, the simulation approach could be the most widespread method in order to conduct these types of studies.

Previously, the Grid suffered from the lack of simulators as well as the limitation of features provided and that was a crucial point. Nowadays, varieties of the simulators are available freely. Many of Grid simulators are built based on each other; for instance: Alea simulator based on GridSim [6] , MaGate Simulator is based on the GridSim as well [7] and Sim-G-Batch Grid Simulator is based on HyperSim-G [8]. There are three types of simulation which are: emulation, mathematical simulation and discrete time event based simulation (most common). All the presented simulators belong to the third type [9].

This paper aims to contribute in assisting and guiding the researcher by highlighting the advantages and drawbacks of the reviewed simulators. In addition, it provides the researcher with described architecture; consequently, it guides the researcher to select the proper simulator for the aimed study seamlessly.

The rest of the paper is organized as follows. We provide an overview of some of the existing tools in Section 2. In Section 3, we present and analyze the features and limitations for the reviewed simulators regarding specific traits in Grid. Finally, we conclude the paper in Section 4.

2.0 SIMULATORS OVERVIEW

Commonly, simulators should cover three main elements, which are: network topology, compute resources and background conditions. Network topology refers to graph, bandwidth and latency, compute resources implies computational resources or any other type of resources, whereas the background condition refers to the load and inaccessibility [10].

In this paper, we provide an overview for four simulators which are; GridSim, Alea, Sim-G-Batch grid simulator and Ball simulator. GridSim deals with large scales distributed computing and parallel environment, Alea is based on GridSim in order to tackle dynamic scheduling problems. Sim-G-Batch grid simulator simulates the security and energy concept, whereas Balls simulator, which specialized in the evaluation of peer-to-peer with integrated load balancing algorithm. The selection of these simulators to review is based on the functionalities that offer, whereas the four selected simulators provide the most possible scenarios and tasks in grid.

2.1 GridSim Toolkit

GridSim [11] is an open tool that enables the user to simulate distributed and parallel computing systems. In addition, GridSim allows the researcher to include users, resources application and schedulers in order to evaluate or create a scheduling algorithm. GridSim provides simulation for a simple mapping between the users and the tasks, as well as between the tasks and the users with ability to manage this mapping. GridSim supports the heterogeneity in resources, where the processor could be single or multi, the memory could be shared or distributed.

GridSim include visual modeler that enables managing resources and users. GridSim can simulate up to 10,000 users with a 200 job list for one user. Visual modeler provides saving and loading the model file through XML format [11].

In GridSim version 5, additional functionalities were included. For instance: a reservation on a grid system that supported by a new framework, additional network topologies and a model that supports auction and economic scheduling algorithms was developed. Although GridSim focuses on computational resources, but a model that supports data grid is included as well.

The architecture of GridSim consists of multi-layered design. the first layer main implementation is for single/multi processor systems. This is achieved by involving with flexible java interface and runtime machinery, which namely Java Virtual Machine (JVM). An event-driven discrete event package on the top of the first layer (JVM) that takes control in the simulation for GridSim, is tackled by the second layer [11].

Third layers is in charge of modeling and simulating the main grid components such as resources, job management, resource allocation, applications and information services, whereas simulating the resource broker is handled by the fourth layer. The final layer considers about configurations for applications, resources and grid scenarios. This layer enables the variety of simulation scenarios through the services that provided by previous layers. This layer assists in providing a comparison and evaluation between the scheduling algorithms by allowing of configuring many different parameters such as: simulation time, applied strategies, the type of the job (task) and that produces an enormous number of possible scenarios that could be conducted. To receive and send the data among the components, input and output classes are determining the ports. Figure 1 shows the GridSim architecture [11].

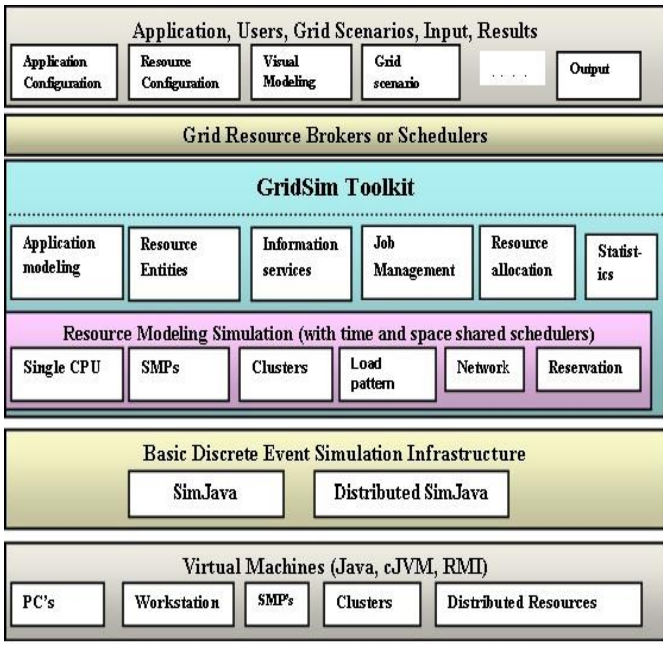


Figure 1 GridSim architecture [11].

2.2 Alea Simulator

Alea simulator [6] is based on GridSim 5 Toolkit simulator. Alea is centralized grid scheduler that implements advanced scheduling technique. One of the main features of Alea, it can deal with dynamic situation if additional jobs are suddenly involved in the system during the simulation. In such case, the generated schedule is updating the job list through the time, and thus, it can handle the new arrival jobs while the execution of old jobs is already done.

Alea simulator consists of eight main entities which are: the Scheduler, Grid Resources, Machine Loader, Failure Loader, Complex Gridlet, Result Collector, Job Loader and Visualizator. The scheduler itself consists of two parts; the first part is the communication, while the second part is the scheduling. The communication part is responsible for linking the Grid Resources with Complex Gridlet (the representative tasks) and submitting the results which received from the scheduling part (section) to the Result Collector. The scheduling part is responsible for scheduling the jobs and that depends on the strategy that used in the scheduler. The Grid Resource entity is in charge of simulating the Grid Resources, whereas Machine Loader creates the resources in the grid. Failure Loader simulates the case that if a failure occurred in the machine. Job Loader is generating dynamic jobs from the workload trace which in turn Complex Gridlet represents these jobs to the Scheduler. Once the simulation is done, the Result Collector saves the data for Visualizator and generates the final simulation results. Finally, Visualizator draws the graphs based on the obtained results [6].

By means of object oriented paradigm that utilized in the development stage, Alea can be smoothly extended. This is because the functions are splitted in different classes and thus, any modification can be conducted smoothly without any conflict with any other function that implemented in different class. Moreover, the scheduler itself is divided into two main parts as mentioned above (the communication part and the scheduling part). Therefore, extending or modifying the scheduling algorithm can be achieved smoothly and efficiently. In addition, if a new job type or a certain property is demanded, only the Gridlet class should be updated [6].

Alea simulator was evaluated and received a positive feedback from many researchers around the world [6]. The evaluation was conducted through two stages. The first stage involves complex real-life data, whereas the second stage tests the speed and the scalability of the simulator comparing to GridSim based GSSIM and the GridSim toolkit. Figure 2 shows the components of Alea simulator [6]

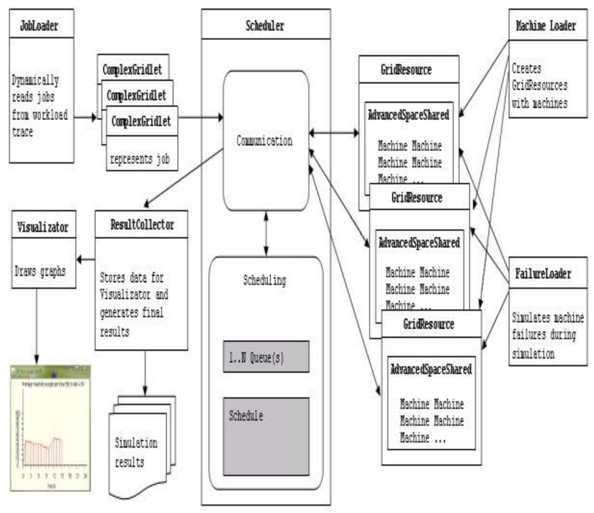


Figure 2 Main components of Alea simulator [6]

2.3 Sim-G-Batch Grid Simulator

G-Batch grid simulator [8] is based on HyperSim-G [12] which concerns about the evaluation of heuristic scheduling algorithm with a diversity of grid scenarios. The configurations in which this simulation is concerned about are: grid resource accessibility, energy consumption, size of the grid topology and the dynamicity of the system. Several scheduling criteria and models are enabled to modify by this simulator. Furthermore, it runs with a melange of meta-heuristic schedulers. The idea of designing this simulator is based on the values that will be generated from the tasks and machines' parameters. These values can be categorized in three different categories which are: the workload and computing capacity parameters, ready time vector and vectors from tasks and machines security demand and trust level. In order to simulate all possible energy scenarios, Sim-G-Batch grid simulator enables configuring many parameters regarding the energy, such as; machine specification parameters which includes (the number of classes, the maximum computing capacity value, the speed of the machine), and Dynamic Voltage and Frequency Scaling (DVFS) levels for different categories of machines [8].

The simulator was evaluated for both security and energy consumption. For security, a simple analysis was conducted by implementing two risk-resilient meta-heuristic-based schedulers integrated with large, dynamic, and heterogeneous scale. The results showed that users have to accept some additional scheduling cost in order to get verification on the utilized resources, which is better than linking their machine with unsecure resources that can cause security hazard to the user's machine. For the energy evaluation, the obtained results showed that maintaining the machine busy for a longer time with modifying their power supply will not make the task constrains regarding the deadline more flexible. But it will enhance the load balancing for the machine and make the power utilization for the whole system less [8].

G-Batch simulator can be extended easily by plugging additional classes; G-Batch simulator has the ability to work in the cloud environment as well. Figure 3 shows the flowchart for security and energy [8].

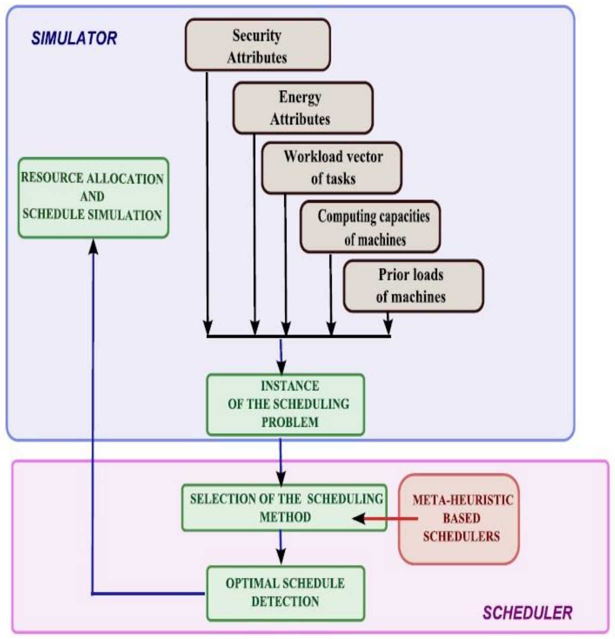

SIMULATOR
Security Attributes
Energy Attributes
Workload vector of tasks
Computing capacities of machines
Prior loads of machines
RESOURCE ALLOCATION AND SCHEDULE SIMULATION
INSTANCE OF THE SCHEDULING PROBLEM
SELECTION OF THE SCHEDULING METHOD
META-HEURISTIC BASED SCHEDULERS
OPTIMAL SCHEDULE DETECTION
SCHEDULER

Figure 3 Flowchart for security and energy in Sim-G-Batch grid simulator [8]

2.4 Balls Simulator

Balls simulator [13] concerns about peer-to-peer structure, this simulator is designed based on partitioning *de Bruijn* graph with its load balancing algorithm. The aim of this simulator is to evaluate the load balance in structured peer-to-peer topology. Balls simulator is not dedicated to grid only. But since many grid networks are using peer-to-peer topology, and with the significance of load balancing in grid, Balls simulator can be helpful and efficient tool in grid.

The architecture of Balls simulator consists of three main layers, which are: Network layer, Peer layer and Evaluation layer. Network layer concerns about the system configuration and peer communication, whereas the functionality is provided by Peer layer. Finally, the Evaluation layer, that in charge of executing simulations and observing output data.

The Network layer is splitted into two parts, the centralized part and the decentralized part. The centralized part represents the entire system on single machine and sense the communication through direct access. The centralized part offers the following components (classes): Balls, Parameters, PeerSet, and Address. P2P system is represented by Balls class. Balls class keeps a backup of the configured parameters and PeerSet (in order to access the peers). Parameters class concerns about modifying the setting from/to the configuration file. Parameters could be like: key length, default number of peers and bandwidth rate. In addition, this class automatically adjusts the values to keep the operation of the system running. Adding peers or removing them, can be done through PeerSet class.

This option is enabled by arranging the peers via an array, the peers are ordered in progressive addresses order. Moreover, accessing the peers is enabled by using binary search. When a simulation cycle is fully executed, this means that the execution of all peers is done. The order of the execution will follow the peers order in the array. In centralized mode, a dedicated simulation cycle time was designed rather than following the machine cycle clock. This is due to the affect of concurrent processes (which are conducted on the machine) on the operation speed. Addressing the peers is the task of Address class. In the centralized mode, the address is and "integer" number.

This integer number is used to locate the peers in the array and in the binary search as well. Thus, the Address class allows sending messages to the peers or its message queue through the integer number. In the decentralized network layer mode, the supported classes are: Balls, Parameters, and Address. On the contrary of centralized network layers, the Balls class in this layer represents only the aspect of one peer. Similar to Parameters class tasks in centralization mode, this layer here provides the same functions. But, since this layer has one aspect peer, the TCP/IP address has to be determined.

Therefore, the address of the EvaluationServer will be set. The Address class determines the TCP/IP address of the corresponding peer. Thus, the peers will exchange the messages through this TCP/IP address and forward the messages to the received peers based on this address [13].

All the classes' functionalities are implemented by the Peer layer. The Peer layer classes are the same for centralized mode and decentralized mode. The Peer layers classes are: Peer, PeerData, KeyInterval, IndexList, StorageList, Message, and WorkSession. Peer class is responsible for performing the actions that peers do. These actions like: exchanging messages, sending protocols, roaming randomly. All these actions are set in advance by required scenario. Class PeerData stand for the triple (p.a, p.b, p.e) for Peer p. the triple (p.a, p.b, p.e) is used by Peer p to determine its own (p.a, p.b, p.e) and the set for the neighbor list. The core calculations in the circular de Bruijn node space are provided by key interval, which is supplied by Class KeyInterval. The core calculation could be: intersection, neighbourhood verification and distance. Managing the objects is the responsibility of IndexList and StorageList classes.

The IndexList provides the list that to pointers which related to the objects that belong to current peer's tasks. This class orders the items in a progressive way of (object key, object id). The StorageList represents the objects, which are listed in the existing peer. This list is ordered relying on the object id. Thus, the binary search is enabled supported. For peer communication, the Message class is needed. Message class offers messages that can be implemented on numerous protocols.

Examples of these messages: routing, leaving, joining, transferring messages, etc. in order to implement the protocols on the peers, Class WorkSession class is required. The basic supported tasks are currently provided by Class WorkSession such as: routing, joining, leaving, index management load balancing, storage load monitoring, and storage load transfer. All the previous classes are held from the peers, while the simulation cycle is running, the continueSession method is called. Once the simulation cycle is over, the classes will be released for another cycle [13].

All the measurement and the evaluation performance functionalities are supported by evaluation layer. Due to Balls simulator supports centralized and decentralized mode, the evaluation layer is splitted into two parts as well (centralized and decentralized) in order to provide evaluation for both cases. The following classes are offered by the centralized evaluation: Peer Degree Evaluation, Routing Cost Evaluation, Arrival-Cost Evaluation, Departure Cost Evaluation, Index Management- Load Evaluation, and Storage Load Evaluation.

These classes providing the performance evaluation for the topology and load balance. Once the simulation runs, these classes start calculating the requested metrics in specific and certain time and store their values to be analyzed later. Peer Degree Evaluation calculates the average and degree of peer distribution attributed to system size. The number of hops that the peer is passing by is registered by Routing Cost Evaluation, thus Routing Cost Evaluation provides the average of routing cost.

Arrival Cost Evaluation indicates the number of messages that sent in arrival attributed to system size, whereas Departure Cost Evaluation indicates the number of messages that sent in departure attributed to system size. Both Load Evaluation, and Storage Load Evaluation are in charge to evaluate the load balancing. This is achieved by indicating the global loads, local load and the capacity of the peer during the whole simulation experiment time. In order to improve the load balancing method evaluation, Balls simulator has the ability to launch and end the functionality of load balancing and the utilization ratio. Figure 4 below shows the Balls simulator architecture [13].

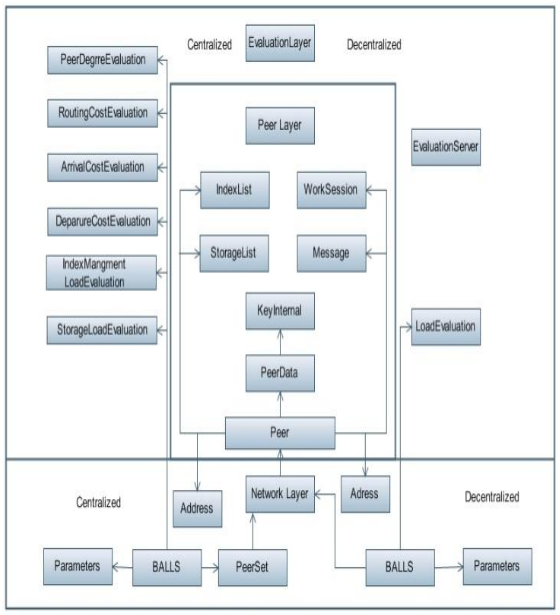


Figure 4 Balls simulator architecture [13]

Several types of load balancing are also supported, thus Balls simulator can execute the scenarios in a very similar way like real grid. In decentralized mode, the index management load balancing and storage load balancing evaluations are enabled as well. In decentralized mode, the evaluation is achieved by conceding different measurements. To calculate the aggregation of these measurements, Balls simulator uses Evaluation Server. Evaluation Server represents a server that indicates the aggregation of the measurements obtained from the peers.

Table 1 Simulations summarized specification comparison

Simulator Name	Objective	Operating System	Programming Language	Limitation and Features					
				X1	X2	X3	X4	X5	X6
GridSim	Scheduling in computing grid	Windows/Linux	Java	Yes	Yes	No	No	No	No
Alea simulator	Scheduling in computing grid	Windows/Linux	Java	Yes	No	Yes	No	No	No
Sim-G-Batch grid simulator	Simulating energy and security	Linux (Ubuntu 10.10)	C++	Yes	Yes	Yes	Yes	Yes	No
Balls simulator	Peer-to-peer structure	Windows/Linux (Red Hat Fedora)	Java	Yes	Yes	Yes	No	No	Yes

X1 Centralization scheduling supporting, X2 Decentralization scheduling supporting, X3 Dynamic scheduling supporting, X4 Security supporting, X5 Energy awareness supporting and X6 Load Balance supporting.

EvaluationServer calculates the requested load balancing measurements such as: (overload, the capacity and the load itself) periodically. These results are saved in a file. For both centralized and decentralized mode, the class LoadEvaluation is the same. This class sends frequently the load notification measurement file that included the existing load measurement. This method avoids the possibility of any computational calculation problem in the client/server, due to the small size of the of load notification file.

The simulator was verified for both centralized model and decentralized model. The verification took a place in two aspects: simulation size and simulation time.

3.0 SIMULATORS FEATURES ANALYSIS

In this section, we will provide the features for the reviewed simulators. We will highlight and analyze the features and limitations for these simulators. The traits are: aim (objective) for the simulator, the operating system that simulator operates on, the programming language used in designing stages and running codes and the limitation. Table 1 shows the properties of these simulators. From Table 1 above, we can observe that GridSim supports both centralized and decentralized mode. Alea simulator supports centralized and dynamic scheduling problems, but it doesn't support decentralized mode, whereas Sim-G-Batch grid simulator supports all the mentioned traits, except the load balance. Finally, Balls simulator supports both centralized and decentralized modes with load balance evaluation method.

4.0 CONCLUSION

This paper reviews some "of many" tools which are currently active and widely used in Grid. In parallel and distributed computing, anticipating the behavior of the participating components is very difficult. Therefore, simulators are playing very important role in evaluating the performance and proposing new algorithms as well. This paper presents four simulators which are: GridSim for simulating scheduling algorithms in computational grid, Alea simulator, which tackles dynamic problem in scheduling, Sim-G-Batch grid simulator, which concerns about the security and energy awareness in the grid and Balls simulator which is dedicated for P2P structured with load balancing evaluation ability.

We have selected these simulators in order to cover the most functions that the researcher is looking for in the grid (scheduling problem, data replication, dynamic scheduling problem, security, energy, and load balance). Finally but not least, the paper has presented their aim, architecture and limitation.

The aim of this paper is guiding the researcher in selecting the proper simulator in the grid; this is achieved by presenting the essential concepts of the architecture for the reviewed simulators, and presenting their abilities in performing grid tasks.

References

[1] L. F. Perrone, "Modeling And Simulation Best Practices For Wireless Ad Hoc Networks," in *Simulation Conference, 2003. Proceedings of the 2003 Winter*, 2003. 685-693.

[2] E. Weingärtner, H. Vom Lehn, and K. Wehrle, "A Performance Comparison Of Recent Network Simulators," in *Communications, 2009. ICC'09. IEEE International Conference on*, 2009. 1-5.

[3] J.-Y. Le Boudec. 2010. Performance Evaluation Of Computer And Communication Systems: *EPFL Press*

[4] J. Mo, 2010. "Performance Modeling Of Communication Networks With Markov Chains," *Synthesis Lectures on Data Management.* 3: 1-90

[5] M. R. K. Grace, S. S. Priya, and S. Surya, "A Survey on Grid Simulators."

[6] D. Klusáček and H. Rudová. 2010. "Alea 2: Job Scheduling Simulator," in *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*. 61.

[7] Y. Huang, A. Brocco, M. Courant, B. Hirsbrunner, and P. Kuonen. 2009. "Magate Simulator: A Simulation Environment For A Decentralized Grid Scheduler," in *Advanced Parallel Processing Technologies*, ed: Springer. 273-287.

[8] J. Kołodziej, S. U. Khan, L. Wang, M. Kisiel-Dorohinicki, S. A. Madani, E. 2014. Niewiadomska-Szynkiewicz, *et al.*, "Security, Energy, And Performance-Aware Resource Allocation Mechanisms For Computational Grids," *Future Generation Computer Systems.* 31: 77-92.

[9] R. S. T. Fahringer, Robert Walter. 2006."Simulating The Grid,"

[10] H. Casanova. 2005."Grid Performance Workshop,"

[11] R. Buyya and M. Murshed. 2002."Gridsim: A Toolkit For The Modeling And Simulation Of Distributed Resource Management And Scheduling For Grid Computing," *Concurrency And Computation: Practice And Experience.*14: 1175-1220

[12] F. Xhafa, J. Carretero, L. Barolli, and A. Durresi. 2007. "Requirements For An Event-Based Simulation Package For Grid Systems," *Journal of Interconnection Networks.* 8: 163-178

[13] V.-D. Le, G. Babin, and P. Kropf. 2006. "Balls Simulator: Evaluator Of A Structured Peer-To-Peer System With Integrated Load Balancing,"" *Research, Innovation and Vision for the Future (RIVF'06),* Ho Chi Minh City, Vietnam,