International Journal of Networks and Communications 2013, 3(2): 45-52
DOI: 10.5923/j.ijnc.20130302.02

# Grid and Cloud Computing Simulation Tools

**Mahdi Mollamotalebi[1,*], Raheleh Maghami[1], Abdul Samad Ismail[2]**

[1]Department of Computer Engineering, Buinzahra Branch, Islamic Azad University, Buinzahra, Iran
[2]Faculty of Computing, Universiti Teknologi Malaysia, 81310, Skudai, Malaysia

**Abstract** Grid computing aims to handle complex applications by a set of computer resources from several distributed nodes. It is different from conventional computing systems such as cluster computing in its loosely coupled, large-scale, high dynamic and heterogeneous nature. Cloud computing is providing hardware and software computing resources as a service over the Internet. Such services are classified as Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). Considering the limitations to implement a real grid or cloud computing environment in terms of needed heterogeneous equipments and large-scale networks for research aims, the simulation tools are applied widely by researchers to model and evaluate grid/cloud behaviour. This paper reviews the current most applied grid and cloud computing simulation tools as well their capabilities in different aspects of applications.

**Keywords** Grid Computing, Cloud Computing, Distributed Systems, Simulation Tools

## 1. Introduction

Nowadays data intensive applications require large computation processing and storage with high speed network connections. The rapid increase of demands is more than available equipments. CERN[1], SDSS[2], and protein archives[3] are some examples of research projects that produce more than peta bytes of data to store, analyze and transfer. These data and users are in different places geographically and a very high communication rate of gigabits per second is required.

Grid computing is defined as a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed resources depending on their capability[4]. The grid system denotes aggregating different machines' capabilities to do a data and resource intensive job efficiently. It is formed by collaborating several heterogeneous resources to achieve higher computational capacities[5, 6]. The previous systems such as parallel systems and clusters are different from grids by some special characteristics of the grid such as heterogeneity, dynamicity and scalability of the participants and resources in grid environments. A grid interconnects clusters, storage systems, scientific instruments, operating system services, and any computing infrastructure to share resources such as CPU, memory and software applications to handle highly data and resource intensive applications[7]. The grid participants manage shared resources and their authorities.

* Corresponding author:
motalebi@gmail.com (Mahdi Mollamotalebi)
Published online at http://journal.sapub.org/ijnc


Grid environment simulators are useful to do high precision experiments independent to the execution platform and in a very large scale modelled environment. The real grid platforms have some limitations to do various experiments including software reconfiguration, hardware / network infrastructure change, and real environments dependencies. They use mostly a small subset of links and routers and it is difficult to access ready-built grid platforms. It is almost impossible to evaluate real grids performance under different scenarios, given the varying number of resources whose availability changes with time and users with different requirements in a grid computing environment by individual researchers. The main limits of real life platforms are their low scalability, low software reconfiguration possibility, and difficulty to having different hardware infrastructure components and topology. Simulation tools support the creation of a repeatable and controllable environment for grid performance evaluation.

Cloud computing is defined as "a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned, and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers"[8]. It provides applications with complex provisioning, composition, configuration, and deployment requirements as a service to the end users under the payment models[9].

Some new issues of cloud computing in terms of the hardware supplying and pricing are as almost unlimited resources accessible on-demand and without needs of provisioning, free from upfront commitments which allows requesting small hardware resources and increase them whenever they are needed, and paying the resources usage

costs only for necessary short terms[10]. Internal data centres of an organization which are unavailable to the general public can also make a private cloud. Developers with creative ideas about an Internet service do not require expensive hardware and software equipments to provide their services. Instead they focus on innovative business values[11]. Some applications of cloud computing are web hosting and real-time data processing with their own requirements.

Some beneficial and exclusive characteristics of cloud computing are as the following. (a) Elastic: since it is able to extend and decrease the computing capabilities on demand (b) Scalable: it handles many requests for processing, storing and network requirements (c) Shared/multi-tenancy: a server/resource in the clouds is utilized by multiple consumers however it is possible to use it as dedicated if desired. Multi-tenancy makes cloud computing economical for public cloud providers (d) Metered: the users pay just for the resources they consumed. It is related to elasticity of clouds. Actually in traditional models a fixed cost is paid based on the assigned resources[12].

There is a challenge to evaluate the performance of real Cloud computing environments for various application models because of having heterogeneous resources, dynamic requirements of users, different workloads and scales of applications, etc. For example, using a real cloud environment such as Amazon EC2 to do the experiments is limited by the scale, because the conditions in the Internet cannot be controlled by the resource allocating or application scheduling providers[9]. So measuring the performance of applications under a real infrastructure and with varying conditions is limited and inflexible. Moreover, repeating the configuration of experimental parameters for very large scale Cloud infrastructures is burdensome. Therefore a simulation tool can be so useful to perform repeatable, controllable and scalable experiments for Cloud computing. It also allows the developers to tune the potential bottlenecks of their applications before using in a real cloud[13].

In this paper, we introduce the common real and simulated environments provided for grid and cloud computing systems and their features concisely. We hope it being useful for the grid and cloud computing researchers to choose the best solution for their hypothesis evaluations.

The rest of the paper is organized as follows. Section II introduces the grid simulators. Cloud computing service providers and simulators are presented in section III and section IV respectively, and the section V concludes the paper.

# 2. Grid Computing Simulation Tools

In the following paragraphs, the most common grid computing simulation tools along with their capabilities and limitations in different applications are presented.

### 2.1. Micro Grid

The aim of MicroGrid[14] was to provide a set of simulation tools which are able to establish and evaluate the grid computing middleware, application, and network services. It enables performing scientific experiments as repeatable and controllable. Grid applications can be run on virtual resources and the researchers can study behaviour of complex dynamic conditions. Virtual resources are modelled as a part of overall simulation and they can be of computing or networking types. It explores a set of virtual resources and executes the applications on a set of physical resources. It has flexibility to handle accurate experiments with heterogeneous physical resources[15].

The MicroGrid enables using Globus[16] applications by execution virtualization which provides a virtual grid. So the current grid applications can be experimented easily. It virtualizes host identities by mapping each virtual/logical host to one real/physical node using mapping tables. The mapping tables contain virtual IP-address to physical IP-address mappings and are used by libraries. By this way, the programs can be performed as transparent on virtual hosts and they are able to communicate with other virtual hosts' running programs. The users log in on the physical hosts and submit their job to a virtual host. It also supports large-scale network which is critical in grid environments simulation[14].

The programming framework of MicroGrid is structured and implemented by C and a limited number of statements are provided to simplify the Globus based simulations. Users modify some files of existing Globus applications to perform the simulations. Finally the main goals of MicroGrid are as supporting scalable applications by utilizing scalable clustered resources, supporting practical grids software environments to perform the simulated applications accurately, and configuring resources characteristics easily.

### 2.2. Bricks

Bricks is developed to evaluate the performance of scheduling schemes in computing environments. It is designed as object oriented and implemented by Java. It simulates the behaviour of computing systems such as job scheduling, nodes' network topologies and processing plans. It can be also used for monitoring, scheduling and predicting the resources in the computing environments and its components are able to be replaced as needed. Its architecture is based on components with replacement capabilities which allow it to be used as simulation tool for different systems and incorporation of current components through foreign interfaces[15].

The architecture of Bricks includes the parts global computing and scheduling to coordinate the simulation functions. The operation of Bricks is like a queuing system's discrete event simulator. The computing simulation environment includes client which is user's node that initializes the computing task, network which interconnects all the grid nodes with some characteristics such as

bandwidth and congestion rates, and server which are the computing resources with some characteristics such as performance and load. The servers are modelled as a queuing system. The jobs are specified by their parameters and operational requirements. Also its scheduling module includes network/bandwidth monitoring, performance and load measuring, resources database, resource prediction, and a job scheduler.

A scripting language is provided in Bricks enabling the users to configure and set the parameters of the computing environment. The users can determine the network topology, architecture of servers, and scheduling components through the Bricks script[17]. Bricks is more applicable for deadline scheduling and performance analyzing in grid environments. It is also able to be used for replicating analysis in data grids.

### 2.3. SimGrid

SimGrid toolkit aims to simulate the computing application's scheduling in a dynamic and heterogeneous distributed environment. It provides the model and abstractions and makes them able to evaluate scheduling approaches. It attempts to be realistic and generate the simulation results accurately. Its first version known as SG was low level and suitable to build general simulations. The newer version, MSG, is more application oriented and able to handle multiple scheduling agents[18].

Considering the complexity of grid computing environments in terms of network aspects and topologies especially in large scales, it may act weakly in handling all the phenomena in such computing systems. But it provides core functionalities of scheduling techniques for distributed applications in heterogeneous grid computing environments. The simulation is performed as event-driven and the most prominent part of the simulation process is the resource modelling. The resources have two performance characteristics as latency to access the resources and the number of tasks handled in a specific duration. These characteristics are modelled by the time-stamped vectors for each resource[19].

The resource interconnection topologies typically are assumed as fully connected in scheduling algorithms. The user can specify the active connections and it can be implemented by connection matrices. The weakness is failing of such fully connected structure to model a realistic grid computing environment because in real environments, the connections are typically in common by communications. The SimGrid does not impose interconnection topologies, rather it assumes the processing and network resources as unrelated and the topology constraints are handled by the grid users. So a user may create some links between the nodes in a specific manner to simulate a router. By this way, it is possible to utilize the SimGrid for flexible large scale grid computing simulations.

Moreover, the SimGrid does not distinguish between a data transfer and computation such that it considers them as tasks and the user is responsible to control the scheduling of computations on CPUs and data transfer on the network connections. It also evaluates the scheduling algorithms under different prediction error scenarios. To handle the data structures manipulation, the SimGrid provided an API for resource (SGResourctz) and for task (SG-Task). The resources are specified by their name, performance metrics, and values. The tasks are specified by their name, cost, and state. The cost may be data size or processing time and state can be the life cycle such as not scheduled, scheduled, ready, running, and completed. Some projects which are built on top of SimGrid are PSTSim[20] to simulate the scheduling of parameter sweep applications over the computational grid, and DAGSim[21] to evaluate scheduling techniques of a DAG form application.

### 2.4. GangSim

GangSim is a scheduling simulator for the grid computing that derives from an enhancement of the Ganglia monitoring toolkit for virtual organizations. It combines simulated components with instances of a virtual organization.

It simulates a policy-driven management infrastructure where the allocation of individual resource is determined from the interactions between allocation policies across virtual organizations. It models elements of real grids such as job submissions, monitoring, and usage policies infrastructures.

The main components of GangSim are ES (external scheduler), LS (local scheduler), DS (date scheduler), MDP (monitoring distribution points), S-PEP (site policy enforcement points), and V-PEP (VO policy enforcement points). Also, the sites are identified by their processors, storage capacity, and network bandwidth. The characteristics come from a configuration file at starting up stage. VOs include a number of users which submit the job(s). Moreover the jobs' data files are published between the sites. ES, LS, and DS specify where different scheduling operations are executed. The ES put the jobs in the queue and chooses the most proper site for them. Then the job moves to the selected site's LS[22].

MDPs handle the metrics of components usage by gathering their information from the schedulers. PEPs enforce the usage policies by gathering the monitoring and other required information. S-PEPs are existed at all sites to control the jobs priorities and leaves. V-PEPs decide about enforcing policies based on the virtual organizations specification for resources allocation. They communicate with S-PEPs and schedulers to apply the virtual organization policies specifications[22].

The Components of GangSim are implemented through simulation modules, job allocation rules, and environmental states containers. It also performs a periodic evaluation (e.g. less than one minute) on the components status[23]. It is able to capture realistic grids' behaviour but not for detailed behaviours of local scheduler and local users' job.

### 2.5. GridSim

GridSim toolkit is an open source platform to simulate the grid computing environments and able to model heterogeneous resources. It includes an extensible information system which is able to store and query information about resource properties to design a resource discovery system. It specifies an arbitrary network topology in the simulated grid environment. Some of the main features of the GridSim toolkit are as following: (a) It allows modelling of heterogeneous types of resources. They also have advanced reservation ability. (b) Applications with different parallel models can be simulated. There is no limit on the number of application jobs that can be submitted to a resource. (c) Multiple user entities can submit tasks for execution simultaneously in the same resource and both static and dynamic schedulers are supported. It allows modelling of several regional grid information services (GIS) components for resource discovery[24]. The characteristics of users in GridSim include job properties (e.g. run time), scheduling constraints, time-zone, budget, etc.

The GridSim has a multi-layered and modular architecture. The first layer is concerned with the scalable Java interface and the runtime machinery, called JVM (Java Virtual Machine), whose implementation is available for single and multiprocessor systems. In the second layer, there is a basic discrete-event infrastructure built using the interfaces provided by the first layer.

Modelling and simulation of core grid entities such as resources, information services, and so on are handled by the third layer and the fourth layer is concerned with the simulation of resource aggregators called grid resource brokers. A broker receives a task issued by a user and applies the scheduling on it. With regard to competitions between users for limited resources, the broker is responsible to handle the related tradeoffs[23]. The final layer is focused on application and resource modelling with different scenarios using the services provided by the two lower-level layers for evaluating scheduling and resource management policies, heuristics, and algorithms.

Each instance of the user entity differs from others by some characteristics such as types of job created, job execution time, number of parametric replications, and scheduling strategy. Each user is connected to an instance of the broker entity. Brokers face to extreme competition while gaining access to resources. Each instance of the resource entity represents a grid resource. Grid information service provides resource registration service and keeping track of a list of resources available in the grid. The interaction between entities is able by using events. The resource entities register themselves with the Grid Information Service (GIS) entity, by sending events. The classes Input and Output define ports to receive and send data in between the entities[24].

There are some other simulators for grid environments such as OptorSim, ChicSim, NSGrid, SimBOINC, GridNet, GrenchMark, etc. with their specific applications, but we just introduced the most used grid simulators in the above. Table 1 summarizes the comparison of above introduced grid simulators.

**Table 1.** Summarized comparison of grid simulators

| Grid Simulator | Coding | Designing environment | Application |
|---|---|---|---|
| MicroGrid | C | Language | To handle virtualization and global coordination |
| Bricks | Java | Language | To reproduce the results of various structures/workloads scheduling |
| SimGrid | C | Library | To schedule multiple servers in dynamic and heterogeneous grids |
| GangSim | Java | Library | To handle usage constraints for sites and virtual organizations |
| GridSim | Java | Library | To handle scheduling, resource discovery and allocation |

# 3. Cloud Computing Service Providers

Cloud computing providers offer the service in the following fundamental models: Infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS), see Figure 1. IaaS is the most basic model and includes physical/virtual computers. IaaS clouds can offer resources such as images, storage, firewall, load balancer, IP-address, etc. An IaaS cloud provider supplies them on-demand by its massive data centers[25]. The users patch and maintain their operating system and applications and the IaaS services are billed based on the utility computing reflecting the resources consumes. Some IaaS providers are Amazon EC2, Windows Azure VM, Google Compute Engine, Aneka and HP Cloud.

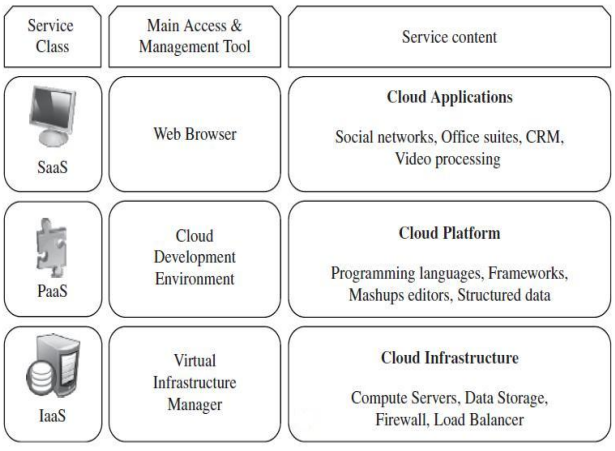


**Figure 1.** The layered organization of the cloud stack[27]

A provider of PaaS services delivers computing platforms such as operating systems, programming environments, databases, and web-servers. So an application developer implements and runs its programs on such platform without needs of payments to buy and manage the underlying hardware/software. The allocated resources are expanded automatically based on the user's application demands. Some PaaS providers include Amazon Elastic Beanstalk,

Engine Yard, Google App Engine, and Windows Azure Compute.

The application is installed and operated by the SaaS providers and the users use them from the cloud. Because the cloud's infrastructure or platforms are not managed by the users there are no needs to install/run the applications on the user's machines which it decreases the maintenance costs. Applications can be distributed to multiple virtual nodes as balanced to keep the scalability, and it is transparent from the user[26]. Some SaaS providers are Google Apps, Salesforce, and Microsoft Office 365.

Several other providers are existing such as Rackspace, Joyent, Apple iCloud, etc. Also HP and IBM provide private clouds to consumers. In the following paragraphs the most important cloud computing service providers are introduced.

### 3.1. Amazon Web Services

Amazon Web Services (AWS) provides online services to other websites or client side applications. Other developers utilize the AWS services in their applications. The services are accessible through HTTP protocol and the costs are paid depending on the usage duration. Today it is a common IaaS provider and some of its main building blocks include S3, EC2, and SQS. The Simple Storage Service (S3) utilizes the (Key,Value) pair to specify the data stored in AWS. It handles data objects of size one byte to five terabytes and the number of objects is not limited. Also the objects are authenticated to apply right assignments.

Amazon Elastic Cloud (Amazon EC2) is a web service to deliver processing capacities over the web. Its interface enables users to get and set the capacities easily by providing full controls for computing resources. It decreases the boot time of new server instances which allows the users scaling the capacity rapidly once changes on the computing requirement occurred. The user pays just for the used capacities. It also makes the users application able to be failure flexible.

Amazon Simple Queue Service (Amazon SQS) is a reliable and scalable queue messaging service. A developer moves its data among distributed components of its application easily without needs the components be always available. It allows having automated workflows joined together the Amazon EC2 and other web services.

All the stations on the network are able to send and receive the messages without needs to install additional software. The components of an application are performed as independent and on its own network. The messages are stored concurrently on different servers to prevent message loses. Authentication preparations are offered ensuring about the messages transferred in the queues. Moreover, the cost of message transferring is low and it is paid per request handles[28].

### 3.2. Microsoft Windows Azure

It is a platform for the public clouds which is usable to create web applications storing data in Microsoft datacenters, to build virtual machines, to create large-scale applications with many users, etc. A cloud application with concurrent users needs a vendor to be established for Software as a Service (SaaS). The Microsoft Windows Azure supports scalable and reliable applications. The user chooses the technology to build its applications arbitrary (such as C, C# or Java) and the codes are executed in instances of virtual machines. Windows Azure handles and monitors the virtual machines, e.g. the patches[29]. It offers both IaaS and PaaS services and primarily is designed to be used as a Windows-based application (.NET) however other platforms such as Java, PHP, etc. are supported. The services provided by Microsoft Windows Azure are Azure Compute, Azure Service Bus, Azure SQL, and Azure Caching.

The Azure Compute offers the computing powers as separated containers (named role) developed for different computing goals. The Azure Service Bus offers a messaging service to handle the asynchronous applications and supports different transport protocols. Azure SQL is provided under SQL-Server databases technology. It uses horizontal partitioning/sharing to control the database's size limitations. The Azure Caching is useful for application's performance improvements by fast accessing the repeatedly used data. The caching service is distributed and 4GB of memory is usable as the cache.

### 3.3. Salesforce

Salesforce provides customer relationship management software solution as a SaaS under the cloud. This software is provided in different categories such as Sales-Cloud, Service-Cloud, and Data-Cloud. The Sales-Cloud is accessible by any user through a device connected to Internet. It provides sale representative along with the customers' profile and accounts records enabling the users to direct the business campaigns. The Service-Cloud allows companies to build and follow cases of all business channels. It also developed social networking websites for the companies to share the analytical capabilities and know the customers' requirements. The Data-Cloud gets the contact and companies profiles from the Sales-Cloud and caused to make the decisions faster and easier through increasing the marketing productivities with fresh and accurate data[30].

The Salesforce also provides a PaaS service with the feature of subscription-based payment mechanism which is different from other resource consumption-based ones. The PaaS is provided through the website FORCE.COM and the users can create their business applications. It uses a programming language similar to Java named APEX. It also offered a component based framework to design the users' interfaces for the cloud applications named Visualforce and utilizes a tag based language[12].

Many other companies are now are providing cloud services. For instance, Google Docs and Google App Engine are provided by the Google company. Google Docs is a SaaS to delivers real-time documents collaborations and Google App Engine is a PaaS platform allows creating applications executable on Google's infrastructure. The user develops its

application locally on his computer using App Engine SDK and the development machine is emulated. Then the user uploads his/her application to Google App Engine to be accessible in the Internet. Apple iCloud is another cloud service which acts in collaboration with Apple's devices such as iPhone, iPad, etc. to data synchronization. Table 1 summarizes the comparison of above cloud computing providers.

**Table 2.** Cloud computing providers comparison

| Cloud Provider | Cloud Type | Provided Services | Costs |
|---|---|---|---|
| Amazon | Public | IaaS, PaaS,SaaS | per hour, Storage usage, Data transfer |
| Microsoft | Public | IaaS, PaaS,SaaS | Subscription,Storage usage, Data transfer, CPU usage |
| Salesforce | Private &Public | PaaS,SaaS | Registration, Unlimited resource usage |

# 4. Cloud Computing Simulators

With regard to specific characteristics of Cloud computing such as isolation of the multi-layered service abstractions (SaaS, PaaS, and IaaS), it is not possible to use grid computing simulators in Cloud modelling. Among the grid simulators, just GridSim is able to support the economic-driven resources management and application provisioning simulation, but it also cannot simulate the Cloud virtualized infrastructures. Yahoo and HP provided a Cloud computing testbed, Open Cirrus that supports datacenters federation for ten organizations[31]. But due to the shared mechanism used for its resource conditions, it is difficult to create its experimental environments repeatedly. In the next paragraphs we introduce some provided simulation tools for Cloud computing and their applications.

### 4.1. MDCSim

The MDCSim is a discrete event simulator developed at the the Pennsylvania state university in 2009 with hardware specification for different servers, communications, connections and switches to estimate their power consumptions. It is a flexible and scalable simulation platform for analyzing multi-tier datacenters. Also, it is able to do the experiments with various designs in three layers and analyze the performance with realistic workload. It estimates the throughput, response times and power consumptions accurately[32, 33].

### 4.2. GreenCloud

GreenCloud is an extension of packet-level simulator NS2 with the capability of evaluating the energy-aware Cloud datacenters. It is able to extract, aggregate, and make the information about computing/communication elements' consumed energy. It can model the network communication aspects of datacenters in detail. Because it is built on top of the NS2, so it can implement TCP/IP reference models. The weakness of GreenCloud is its limitation on scalability. It is usable only in small datacenters because of high time and memory requirements during the simulation[34].

The GreenCloud is coded in C++ and it acts as packet level such that for transferring data messages among entities a packet should be allocated in the memory and then its protocol processing can be executed. But CloudSim and MDCSim are event based and they do not build and process the packets individually. They capture the effects of objects interactions which can reduce the simulation time and improve the scalability significantly. On the other hand the GreenCloud is more accurate simulator[32].

### 4.3. CloudSim

With regard to complex composition and deployment requirements of Cloud computing services, it is difficult to evaluate the performance of Cloud's application and resource models under different systems and users' configurations. The CloudSim attempts to tackle with this problem by modelling the Cloud computing systems and their components such as datacenters, virtual machines etc. It is an extensible and easy to setup simulating framework for Cloud computing infrastructure and application services. Less efforts and times are needed to deploy Cloud based application experiments in CloudSim. It is able to model large scale Cloud environments on single node. It also supports the network topologies simulations and federated Cloud environments. It can create and manage multiple independent virtual services on one datacenter with the capability of working as space or time shared allocations[35].

In the CloudSim software architecture, the layer called CloudSim simulates the virtualized datacenters including management interface and storages. Assigning the nodes to virtual machines, handling the applications execution and system states are some other roles of this layer. So the efficiency of hosts allocation can be implemented in this layer. Cloud hosts are able to be allocated to a number of virtual machines as concurrent. The user code layer indicates the fundamental entities of nodes such as its machines specifications and applications, number of users, and scheduling constraints. By using this layer, the developers can simulate different Cloud scenarios with customised settings[35]. The bottommost layer manages the interactions between CloudSim components which is possible through message passing. The packet model and flow model which are widely utilized in distributed systems simulations are supported in CloudSim to design the networks. The flow model has lower computational overheads and approximates properly the real network models such as TCP/IP[34].

Because CloudSim provides the basic components (nodes, virtual machines, and applications), so it is able to simulate the scenarios of all IaaS, PaaS, and SaaS services. The researchers using CloudSim concentrate only on their evaluating system designs regardless to the low level details of Cloud infrastructure[9]. It is also scalable with low

simulation overheads. Table 3 summarizes the comparison of introduces cloud computing simulators.

**Table 3.** Summarized comparison of cloud simulators

| Cloud Simulator | Language of coding | Open Source availability | Simulation speed | Graphical environment |
|---|---|---|---|---|
| MDCSim | C++ and Java | × | Fast | × |
| GreenCloud | C++ | ✓ | Low | NetworkAnimator |
| CloudSim | Java | ✓ | Fast | CloudAnalyst |

# 5. Conclusions

Grid and cloud computing environments have some specific characteristics such as heterogeneity, dynamicity and scalability which necessitate particular research tools. Performing the real grid/cloud platforms experiments deal with some limitations on software/hardware reconfiguration, enlarging the scale, etc. which impede performance evaluations. The researchers need to experiment and tune their proposed grid/cloud computing method under different scenarios and varying number of resources/users to realize its strengths and weaknesses (e.g. potential bottlenecks) before using in a real environment.

This paper has introduced the most important and common simulators for grid and cloud computing as well their characteristics. Also the main cloud computing providers and their services are introduced concisely. We also presented a summarized comparison of introduced tools and providers. To sum up, none of the introduced tools is the best of all. Each simulator is proper for a specific application and conditions of grid or cloud environment and the researchers choose the best adapted tool according to their requirements and applications.

# REFERENCES


[1] Large Hadron Collider (LHC). Available: http://lcg.web.cern.ch/LCG/

[2] Sloan Digital Sky Survey. Available: http://www.sdss.org

[3] 2012). The protein data bank (pdb). Available: http://www.rcsb.org/pdb/

[4] R. Buyya, "Economic Paradigm for Service-Oriented Grid Computing," presented at the The INT Media's Grid Computing Planet Conference and Expro, San Jose, California, USA, 2002.

[5] R. Buyya, et al., "Nimrod/G: an architecture for a resource management and scheduling system in a global computational grid," in High Performance Computing in the Asia-Pacific Region, 2000. Proceedings. The Fourth International Conference/Exhibition on, 2000, pp. 283-289 vol.1.

[6] K. Krauter, et al., "A taxonomy and survey of grid resource management systems for distributed computing," Software: Practice and Experience, vol. 32, pp. 135-164, 2002.

[7] P. Trunfio, et al., "Peer-to-Peer resource discovery in Grids: Models and systems," Future Gener. Comput. Syst., vol. 23, pp. 864-878, 2007.

[8] R. Buyya, et al., "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Gener. Comput. Syst., vol. 25, pp. 599-616, 2009.

[9] R. R. a. R. N. C. Rajkumar Buyya, "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities," presented at the 7th High Performance Computing and Simulation Conference (HPCS), Leipzig, Germany.

[10] M. Armbrust, et al., "A view of cloud computing," Commun. ACM, vol. 53, pp. 50-58, 2010.

[11] M. Armbrust, et al., "Above the Clouds: A Berkeley View of Cloud Computing," 2009.

[12] E. J. Qaisar, "Introduction to cloud computing for developers: Key concepts, the players and their offerings," in Information Technology Professional Conference (TCF Pro IT), 2012 IEEE TCF, 2012, pp. 1-6.

[13] A. Quiroz, et al., "Towards autonomic workload provisioning for enterprise Grids and clouds," in Grid Computing, 2009 10th IEEE/ACM International Conference on, 2009, pp. 50-57.

[14] H. J. Song, et al., "The MicroGrid: A scientific tool for modeling Computational Grids," Sci. Program., vol. 8, pp. 127-141, 2000.

[15] C. S. Y. Anthony Sulistio, Rajkumar Buyya, "Simulation of Parallel and Distributed Systems: A Taxonomy and Survey of Tools ".

[16] K. C. Foster I, "Globus: A metacomputing infrastructure toolkit," The International Journal of Supercomputer Applications and High Performance Computing, vol. 11, pp. 115-128, 1997.

[17] A. Takefusa, et al., "Overview of a performance evaluation system for global computing scheduling algorithms," in High Performance Distributed Computing, 1999. Proceedings. The Eighth International Symposium on, 1999, pp. 97-104.

[18] A. Sulistio, et al., "A taxonomy of computer-based simulations and its mapping to parallel and distributed systems simulation tools," Softw. Pract. Exper., vol. 34, pp. 653-673, 2004.

[19] H. Casanova, "Simgrid: a toolkit for the simulation of application scheduling," in Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on, 2001, pp. 430-437.

[20] H. Casanova, et al., "The AppLeS Parameter Sweep Template: User-level middleware for the Grid\m{1}," Sci. Program., vol. 8, pp. 111-126, 2000.

[21] A. Jarry, et al. (2000, DAGSim: A Simulator for DAG SchedulingAlgorithms.Available:http://lara.inist.fr/bitstream/handle/2332/509/LIP-RR2000-46.pdf?sequence=1

[22] C. L. Dumitrescu and I. Foster, "GangSim: a simulator for grid scheduling studies," presented at the Proceedings of the Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05) - Volume 2 - Volume 02, 2005.

[23] F. C. Benjamin Quetier, "A survey of Grid research tools: simulators, emulators and real life platforms," in Proceedings of 17th IMACS World Congress (IMACS 2005), Paris, France, 2005.

[24] R. BUYYA, MURSHED, M.M. Year., "GridSim: a toolkit for the modeling and simulation of distributed resource management and scheduling for Grid computing," in Concurrency and Computation: Practice and Experience, 2002.

[25] H. S. Alex Amies, Qiang Guo Tong, Guo Ning Liu., Developing and Hosting Applications on the Cloud: IBM Press, 2012.

[26] L. T. Hamdaqa Mohammad, Tahvildari Ladan, "A Reference Model for Developing Cloud Applications," presented at the CLOSER, 2011.

[27] W. Voorsluys, et al., "Introduction to Cloud Computing," in Cloud Computing, ed: John Wiley & Sons, Inc., 2011, pp. 1-41.

[28] (2012, "Amazon elastic compute cloud". Available: http://aws.amazon.com/ec2/

[29] (2012, windowsazure official website. Available: http://www.windowsazure.com

[30] (2012, Official Salesforce website. Available: http://www.salesforce.com

[31] A. I. Avetisyan, et al., "Open Cirrus: A Global Cloud Computing Testbed," Computer, vol. 43, pp. 35-43, 2010.

[32] D. Kliazovich, et al., "GreenCloud: a packet-level simulator of energy-aware cloud computing data centers," The Journal of Supercomputing, vol. 62, pp. 1263-1283, 2012/12/01 2012.

[33] L. Seung-Hwan, et al., "MDCSim: A multi-tier data center simulation, platform," in Cluster Computing and Workshops, 2009. CLUSTER '09. IEEE International Conference on, 2009, pp. 1-9.

[34] S. K. Garg and R. Buyya, "NetworkCloudSim: Modelling Parallel Applications in Cloud Simulations," in Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on, 2011, pp. 105-113.

[35] R. N. Calheiros, et al., "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Softw. Pract. Exper., vol. 41, pp. 23-50, 2011.