Simulation Modelling Practice and Theory 104 (2020) 102144

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Cloud computing simulators: A comprehensive review

N. Mansouri[a,b,*], R. Ghafari[a], B. Mohammad Hasani Zade[a]

[a] Department of Computer Science, Shahid Bahonar University of Kerman, Iran
[b] Mahani Mathematical Research Center, Shahid Bahonar University of Kerman, Kerman, Iran

ARTICLE INFO

Keywords:
Cloud computing
Architecture
Energy model
Simulator features

ABSTRACT

Nowadays, cloud computing is an emerging technology due to virtualization and providing low price services on pay-as per-use basis. One of the main challenges in this regard is how to evaluate different models of the cloud resources usage and the ability of cloud systems based on QoS constraints. Experimentation in a real environment is very difficult and expensive. Therefore, many works pay much attention on designing cloud simulation frameworks that only cover a subset of the different components. Consequently, choosing the right tools to use needs a deep comparative analysis of available simulators based on different features. In this paper, we present a comparative analysis of 33 tools for cloud environments. This review will enable the readers to compare the prominent simulators in terms of the supported model, architecture, and high-level features. Subsequently, it provides recommendations regarding the choice of the most suitable tool for researchers, providers, and managers of cloud environment. Eight common simulators are appraised in a practical way based on various scenarios in order to evaluate their performance. Finally, it addresses the open issues and future challenges for further research.

1. Introduction

Over the past few years, cloud computing has gained tremendous popularity since it presents a flexible and efficient solution for many services by Internet [1,2]. Cloud is a large and complex system since it is composed of several users, service providers, physical machines, service broker, task scheduling algorithms, bandwidth, internet latency, and storage technology etc. [3]. On the other hands, all cloud-based implementations need various configurations. Therefore, it is very difficult for researchers to evaluate the performance of their policies in a real cloud environment. Cloud simulation tool is a viable and attractive solution for this challenge since it provides the analyzing of system behavior by considering a specific component under different scenarios.

Obviously, the traditional simulators of distributed systems [4,5] can't model the cloud computing community and so a lot of studies focus on this issue and try to design simulation toolkits for cloud computing. Nevertheless, due to the availability of numerous cloud simulators it is necessary that a critical evaluation of such tools be investigated in order to select a suitable one for specific research. Because every simulator does not support all requirements of the researcher for evaluation. Many surveys of cloud tools have been presented during the last years and most of them study the architecture and the high-level features of each simulator [6-11].

Table 1 shows various review works on cloud simulator in the literature. The novelty of the work we present here, in relation to other surveys, is to focus on a study of 33 tools in terms of architectures, and main positive and negative characteristics. Moreover, we present list of problems that each tool has solved. Eight common simulators are appraised in a practical way based on various

* Corresponding author at: Department of Computer Science, Shahid Bahonar University of Kerman, Iran.
E-mail address: najme.mansouri@gmail.com (N. Mansouri).

https://doi.org/10.1016/j.simpat.2020.102144
Received 15 February 2020; Received in revised form 9 June 2020; Accepted 25 June 2020
Available online 12 July 2020
1569-190X/

Table 1
Review articles for cloud simulators.

Reference	Year	Main goal	Weakness
Zhao et al. [6]	2012	Present a comparison among 11 tools based on developing language that simulators use, software or hardware that designed simulator is.	It does not discuss about advantages, disadvantages of simulators and their applications.
Singh et al. [7]	2015	Study 10 simulators and present their motivation.	It does not compare the main characteristics of simulators.
Sharkh et al. [8]	2016	Present a comparison among 12 various simulators in terms of motivations and main features.	It does not explain the architectures of simulators and does not cover many other important simulators.
Bhatia and Sharma [9]	2016	Investigate the platform, graphical support, license, and language of 12 simulators.	It does not discuss other important features (e.g., communication model, energy model, and etc.) and their applications
Byrne et al. [10]	2017	Provide a review of 33 cloud tools based on very high-level feature such as platform and documentation available.	It does not analysis the strong points and limitations of simulators. Also, their applications and architectures aren't specified.
Fakhfakh et al. [11]	2017	Give an overview of 22 tools based on different criteria such as energy and communication models.	It does not specify the application of simulators and their architectures.

scenarios in order to evaluate their performance regarding resource consumption and execution time for solving scheduling problem. We discuss the performance of simulators with focusing on the scalability factor that indicates how fast a toolkit can execute as the number of requests is increasing.

1.1. The importance of cloud simulators

Each cloud system should have some characteristics such as the awareness of geographic location, high access time, rapid elasticity, management of heterogeneity data, and application scalability to achieve its primary goals [12]. These features inherently lead to the development of complex application modules and resource management schema. New strategies that are designed for the cloud to take advantage of the characteristics of the cloud must be investigated in various aspects like low latency [13,14]. Most of the cloud researchers have been interested to study the cloud challenges like power saving, cost modeling, load balancing, and security concepts. Recently, it becomes more and more significant to discuss around some areas of cloud computing such as:

- How do cloud computing and cloud-based applications perform?
- Are services of cloud computing safe and privacy protected?
- Which cloud services are more energy-efficient and affordable?

Different systems collect large amounts of data independently of the human business processes and then deploy this information in IT and business purposes. When there are several parameters relevant to technical performance, then proposing an efficient cloud solution is becoming more and more challenging. Therefore, developers require an appropriate way to model their strategies and extend data management policies. Testing in a real environment (e.g., Amazon EC2 [15], Microsoft Azure [16], Google App Engine [17]) is expensive, time consuming, and unrepeatable especially for performance (e.g., throughput, cost benefits) and security issue analysis of cloud. In other words, quantifying the performance of scheduling strategies in a real cloud under transient conditions will lead to many problems for the following reasons:

- Cloud consists of different requests, access patterns, and resources (i.e., hardware, software, and network).
- Users show dynamic behavior based on QoS requirements.
- Applications may have varying workloads and characteristics.
- Cloud-based environment control is not in the hands of developers and so it is not feasible to repeat benchmarking experiments under the same conditions.

Furthermore, experimentations are constrained by the rigidity of the infrastructure since if a failure occurs during testing, then it will affect the results of tests as well as other applications execution. Performance analysts are struggling with performance challenges in the large-scale cloud due to massive scalability, high complexity, and dynamic configuration. While one of the critical factors in the maintenance of performance is to guarantee that system performance is SLA-driven. Scheduling of requests based on different components generates many various types of execution paths and so performance analysts face with a challenge in determining system behavior [18]. For example, specifying the main source of the problem in performance degradation or finding the critical execution paths could be challenging. The best solution for this challenge is a cloud simulator that models different types of cloud applications, data centers, virtual machines, and many analysis utilities.

From another point of view, the energy consumption of cloud data centers is an important challenge. Energy consumption is increasing day by day due to a non-energy aware strategy in resource management. About 0.5% of energy consumption in worldwide is related to cloud data centers, while this is envisaged to fourfold in 2020 [19]. To this end, it makes sense to evaluate various polices with simulators based on energy model to optimize the use of cloud resources.

In terms of cost factors, using of real resources for the evaluation of new strategies from the beginning of the solution development process is not always feasible. On the other hand, using of benchmark solution (i.e., considering a set of servers) does not ensure a suitable view of scalability issues. While scalability plays an important role in cloud scenarios and evaluation by analytical methods is impossible due to the increasing complexity.

As a result, the simulator can play an important role in reducing cost, efficiency, infrastructure complexity, and security risks before a solution can be deployed on real infrastructure. By focusing on issues related to the quality of a particular component under various scenarios, cloud simulators enable performance analysts to monitor the behaviors of the system. In summary, cloud simulators have the following general advantages compared to cloud service [20]:

- Cloud simulators do not need installation and the costs of maintenance (i.e., no investment). In addition, risk assessment in simulation tools during the early stages does not involve the capital cost while it is implemented. Thus, a developer can recognize a risk that is related to the design or any parameter.
- In cloud simulators, user can easily change inputs, scenarios, and load. Therefore, the results as output can be analyzed in different conditions.
- Simulator learning is easy for developers. If they know the programming language well, the evolution with simulators will not be a problem.

Furthermore, potential users who benefit from cloud simulators include:

- Cloud providers and architecture: They are the main users and can design and evaluate their strategies with a simulator.
- Cloud clients: Simulators can be useful for large companies to compare various providers of private and public clouds, evaluate cloud deployment solutions, and study customer-related workloads.
- Scientific community: Simulators are primary tools for researchers to analyze their proposed methods before testing on a real setup.
- External players: The simulators can be helpful for everyone who is worried about cloud applications. For example, government employees should be concerned about energy efficiency or carbon printing from a cloud. Therefore, their researchers can study energy performance configuration with a suitable cloud simulator.

Without cloud simulators, researchers and cloud providers only have to use theoretical and inaccurate assessment or trial and error that lead to a waste of resources [21]. In the literature, there are traditional simulators for distribution systems [22] that do not have the ability to model the cloud environment. In the last few years, different cloud simulators are designed for evaluating resource utilization.

1.2. The relationship between cloud simulators and the cloud-based software development

We explain a use case to show the main roles of a simulator in software development. Consider a scenario that developer aims to design a multi-layered resource allocation strategy for data centers of a cloud. In cloud environment, a resource allocation strategy tries to reduce access latency and satisfy SLAs. The cloud simulator will initially help the developer in defining the problem by answering to the basic questions such as the following:

- What elements does this development (environment) include?
- Which resources should be affected?
- When the resource allocation policy should be triggered?
- What factors seem to be crucial?

Then, a simulator can help to the developer for finding the weaknesses that make a solution ineffective such as:

- Is it based on suitable topology?
- Is it utilizing the network resource effectively?
- Is the workload distributed well between VMs?

Then, developers can maneuver based on several experiments (i.e., configurations) and “What if?” scenarios to analyze the impact of each parameter during resource allocation procedure. For example, the proposed strategy may perform better for data-intensive requests than CPU intensive applications. Then, developer decides which parameters have more significance on the results of scheduling algorithm in different conditions. For example, developer changes the number of data centers, load of communication, and number of requests to monitor the results of system. Finally, the obtained results are used for final tuning before the proposed strategy is implemented in a real environment for scheduling the real requests.

Table 2
Basic entities of cloud simulator.

Elements	Examples
Basic components	Servers, data center, switches, links, users, virtual machine.
Attributes of component	Capacity, power capability.
Simulation scheme	Stochastic, deterministic.
Events	Entry of task or user, task completion.
Frequency of events	Distribution of task arrival or access pattern (e.g., uniformly random and exponentially distributed).
Main functions	Task scheduling, resource management, virtual machine migration, data management.
Process sequence	Definition of relations among events, entities, and results.

1.3. Cloud simulator challenges

Several grid simulators such as MONARC [23], SimGrid [24], GridSim [25], OptorSim [26], and MicroGrid [27] have been developed to model the behaviors of grid applications (i.e., scheduling, data replication, allocation, and monitoring). In addition, for modeling on-demand virtualization using resources and management of applications, there is little support or not at all support in the grid simulators that exist [10].

In grid toolkits, tasks are considered deterministic and non-interactive, while cloud toolkits should model the behaviors of data centers that include several virtual machines in multi-tenancy conditions with dynamic user load at non-deterministic intervals [10]. In addition, the cloud aims to provide services based on subscription way and a pay-as-you-go model for cloud users. Because of this, cloud simulators should define some economic components like brokers and cloud exchanges for supporting the real-time business services between provider and customer.

The simulator designing procedure involves making decisions about: the entities that are created, served, or operated during the simulation procedure. Some entities of cloud are very common like switches, servers, tasks, links, and users. Moreover, simulator design must cover the flow of the simulation procedure, the resources and events related to it or the processing phases, and pay particular consideration to the frequency of the event and its duration. It is also important to select the suitable probability distribution for describing the uncertainty and process changes. Table 2 lists the basic components that are necessary for designing the first prototype of a cloud simulator [8].

There are several challenges in the cloud computing simulators that are explained as follows.

1.3.1. Challenges of physical components and structure

The selection of simulator elements can be an effect on the structure of simulator and its organization. The designer should develop an adaptive resource scheduling and allocation policy based on simulator architecture to improve resource utilization. Most of the typical simulators define a constant number of virtual machines that have fixed capabilities. While it is better to assign a chunk of resources to virtual machines and then these capabilities are scaled up and down according to the real-time usage in the cloud [8]. Furthermore, the probability of requests and the locations of clients is variation. Cloud environment has dynamic nature and so resources must be automatically allocated and de-allocated to increase agility. Therefore, defining an appropriate distribution in a cloud simulator is a point to discuss.

Defining a reasonable relationship among important parameters such as processors, memory, and network is a critical challenge. We can achieve to the lower execution time with the higher computing speed but the total cost is increased. For the same workloads, various configurations setting lead to different prices. Tsai et al. [28] introduced an intelligent configuration model based on heuristic rules for the cloud environment. They try to tune the configuration by a heuristic model along with the cost constraints. Moreover, their proposed model extracts the trend patterns from the historical data. Then, it uses the obtained patterns for estimating future usage. The size of historical data is a point to be discussed. For example, if we have large size data then linear regression is suitable for determining the relationship of the usage and time. Furthermore, the size of time interval is an important factor during the decision. For example, more number of intervals can be used for the sensitive workloads since the obtained results are more accurate. Finally, the existence of an outlier in history should be analyzed. For example, we can remove outliers when its reason is random variability. Otherwise, the accurate analysis is required for the future prediction. Most of the designer only consider Infrastructure as a Service (IaaS) aspects of the cloud, while Platform as a Service (PaaS) or Software as a Service (SaaS) should be offered by a simulator of cloud [8].

1.3.2. Challenges of network model

Many of the challenges encountered when developing a cloud computing system is yet faced by software architects when trying to introduce a cloud simulator. One of these challenges is determining approaches for routing network requests and rerouting to reduce the arrival delay. Therefore, cloud simulators must propose alternatives for solving traffic issues like that existing in real cloud data centers. Using relay switches, traffic handling with a central controller (such as techniques used in Software Definition Network (SDN) controllers) [29,30], Routing as a Service [31], or allowing the user to make decisions can involve these alternatives. Other challenges of the network model are related to selecting a fixed and flexible bandwidth allocation mechanism. Moreover, the designer of simulator should consider VM interconnection topology. In a real cloud environment, a fully connected model is not usual and hence simulator should model a realistic network with adding root, aggregate, and access level switches [32].

1.3.3. Challenges of resource allocation and load balancing

There are many external and internal challenges in designing a resource allocation method for a cloud environment. A summary of these challenges is presented in the article by Sharkh et al. [33]. For example, external challenges involve geographical challenges, dynamic manner of users in resource demands, and optimizing the cost model to maximize revenue. As a result, these challenges can lead to limitations on the location of VMs. In addition, a dynamic load balancing technique is vital since the large variation in the workloads of the cloud happens [34]. Data locality is one of the internal challenges, particularly for data-intensive applications. Therefore, each cloud simulator should be predefined several scheduling and data management policies. Another strengths point is letting the user to add their scheduling algorithms and precisely evaluate them. Since resource assignment is one of the factors that has a significant impact on the effectiveness of power, economic profit, availability, and overall performance of the cloud data center [35,36].

1.3.4. Challenges of application model

Cloud simulators are different in how user requests and executive entities are presented. Some typical simulators present user demands or tasks by a list of resource characteristics such as processing capability in the millions of instructions per second (MIPs), storage space, and deadlines. While advanced simulator considers tasks as the abstracted into applications and tries to assume the interdependence of the tasks. Since tasks have to communicate in real cloud environments [37]. This scenario is reality more accurate and effects on task scheduling. There are more limitations for running a cloud task than running a grid task and so cloud simulation is more difficult. One of the key features of the cloud is scalability (i.e., scale in and out) in the application model. In the simulation, it is difficult to increase the number of components that are provided by a particular application and the capabilities of resource at runtime. Nowadays, cloud data centers are usually used to store large-scale data files and so one of the most challenging issues is reducing energy consumption of cloud data centers despite the large volume of data and the dynamic nature of the cloud [38]. Therefore, cloud simulator should provide basic statistics for power consumption of data center components. Furthermore, the main goal of cloud service provider is to maximize the profit from cloud infrastructure, so cloud simulator should provide a cost model based on different factors such as payment budgets and task preferences.

1.3.5. Challenges of scalability

One of the most significant factors in introducing a novel cloud simulator from the beginning or expanding one of the available simulators is the capability of scaling up to real use cases. A cloud simulator must be able to model large-scale topologies with numerous requests, which is similar to the real cloud system [39]. Therefore, the features of simulators (e.g., programming languages) may lead to scalability issues. If cloud simulator applies simulation engines to manage events, then it may have some limitations on the size of the applications and the number of data centers. A typical simulator may be able to solve a simple problem at a reasonable time, but it will not be able to solve the problem in an appropriate time when faced with a high number of requirements. Therefore, the designer of cloud simulator tries to provide parallel data processing techniques to overcome this challenge. Therefore, we can see that designing a comprehensive simulator for cloud requires a wide variety of knowledge such as security issues, real network components, cost policies, energy modeling, architecture, load balancing techniques, cloud applications, and data management.

The rest of the paper is organized as follows. Section 2 presents the background information about cloud computing environment and simulations. Section 3 reviews different cloud tools. Section 4 compares all simulators. Section 5 presents the simulation based study analysis. Section 6 provides conclusion and future scope.

2. Background

2.1. Cloud computing

Cloud computing is the most popular computing system in the academic research and industry, and it provides cloud services on-demand anywhere and anytime [40,41]. The five essential characteristics of cloud computing (i.e., on-demand self-service, high-performance network access, rapid elasticity, resource pooling and measured service) are represented in Fig. 1 [42].

The cloud can be effectively deployed in many ways and four main deployment models are: (1) Public cloud that provides cloud services for public use (e.g., Google Cloud Platform), (2) Private cloud that provides cloud services only for the organization itself and the main usage is managing of sensitive data, (3) Hybrid cloud combines the functionalities of both public cloud and private cloud, and (4) Community cloud shares the cloud infrastructure among several organizations or individuals to address the common concerns. In Fig. 1, there are three service models of cloud: (1) SAAS (Software as a Service) is application that's available via a third-party over the internet, (2) IAAS (Infrastructure as a Service) is cloud-based service such as storage, networking, and virtualization, and (3) PAAS (Platform as a Service) provides a platform on which software can be developed [13].

Therefore, cloud is a complex environment with a large number of data centers and tasks and it must schedule tasks frequently among the resources and flexibly manage to meet the requirements of the users. The all advantages mentioned above have to be properly evaluated before they can be applied by the users. The service providers have to evaluate the different algorithms and management models so that they can obtain maximum profit and user satisfaction. It is essential to test all approaches in a repeatable and controlled environment with varying composition and scenarios. However, evaluating the behavior of cloud in a real environment is not practical since it involves huge cost, scalability problem, and other issues. The solution to this challenge is using simulators that can model cloud functionalities before being implemented in a real setup.

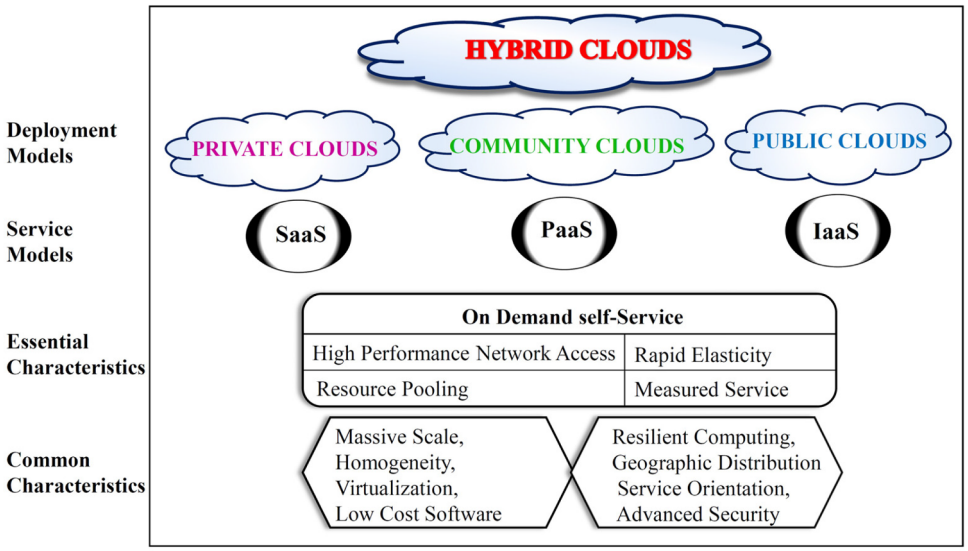


Fig. 1. NIST cloud definition structure.

2.2. Cloud simulation

The deployment of real cloud incurs heavy cost and great effort. The cloud simulators can be used to evaluate the behavior of cloud in a repeated manner. Fig. 2 illustrates the common architecture for cloud simulators that has four layers: (1) Resources layer that provides hardware elements such as CPU, memory, storage and network bandwidth, (2) Cloud Services layer that virtualizes the available resources of cloud for provisioning to users' requests, (3) Application layer that allows users to submit their applications and consume the resources, and (4) Simulator Kernel layer that consists of the libraries for managing the simulation and its parameters.(1) Application layer that allows users to submit their applications and consume the resources, (2) Cloud Services layer that virtualizes the available resources of cloud for provisioning to users' requests, (3) Resources layer that provides hardware elements such as CPU, memory, storage and network bandwidth, and (4) Simulator Kernel layer that consists of the libraries for managing the simulation and its parameters). Moreover, this layer provides the configuration of the cloud research experiments that necessary for executing on the virtual machines. Table 3 lists the potential cloud activities that a simulator tool has a role to play.

Several cloud tools having different features and are developed for specific purposes but all of them consist of main components that are shown in Table 4. In summary, a good cloud simulator presents the following characteristics: ease of use, graphical user interface support, configuration, flexibility, repeatability, network model support, economic-driven resource management, and energy modeling.

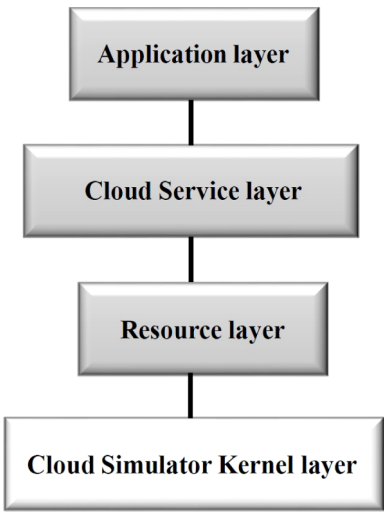


Fig. 2. Architecture for cloud simulators.

Table 3
The main activities of simulator.

Define	Process details such as resource management are specified more precisely and easily.
Pinpoint	Factors affecting performance, time, execution costs etc. are well investigated.
Maneuver	Different scenarios are tested to identify the best conditions and results.
Analyze	Changing parameters such as the number of requests and the ability of the system are checked to keep the conditions manageable.
Decide	By comparing different algorithms, their strengths and weaknesses are revealed.
Scale	Running multiple real scenarios will improve the validation and verification process.
Confirm	The quality of solutions are reviewed and the risk of error and loss of money are reduced.

Table 4
The main components for designing a cloud simulator.

Simulator objective	The main objective (e.g., energy consumption, security, performance, etc.) must be determined.
Application model	User requests with resource specifications (required MIPs, memory, storage, and deadline), request probability distributions, and dynamic scalability must be considered.
Network model	Topology representation, network request representation, routing of a request, and bandwidth allocation must be defined.
Resource scheduling	Scheduling strategy for assigning tasks to the VM for execution based on different factors (e.g., resource utilization, execution time, energy efficiency and revenue generation must be designed.
Customization capability	Changing capabilities to meet user' function must be provided.
Presentation issues	Well-designed appearance should be presented for input setting and results analyzing.

3. The review of cloud simulation tools

Today, several cloud simulators are developed by mathematical formulas to model real environment. We present a concise description of 33 simulators in two parts: (1) Description and main features, and (2) Architecture. In the first section, we review simulators that are based on CloudSim tool. The second section describes the analysis of other cloud simulators.

3.1. CloudSim based tools

In this section, CloudSim is introduced and then all extensions of CloudSim framework are explained.

3.1.1. CloudSim

Description and main features: Calheiros et al. [43] presented CloudSim as a toolkit for modeling cloud environments at CLOUD Laboratory, University of Melborn, Australia. CloudSim models data centers, virtual machines, service brokers, and resource provisioning methods. It provides a flexible switching between space-shared and time-shared allocation of processing elements to services. Researchers can implement cloud-based application by very less effort and time and test the performance in heterogeneous environments.

Architecture: Fig. 3 indicates the CloudSim structure that is composed of three main layers: (1) User Code that provides the configuration parameters for hosts, cloudlets, Virtual Machines (VMs), number of users, and broker scheduling algorithms, (2) Cloudsim that manages the execution of core elements such as cloudlets and data centers during simulation, and (3) CloudSim core simulation engine that models queuing and communication between components.

3.1.2. NetworkCloudSim

Description and main features: Garg and Buyya [32] presented a new framework as NetworkCloudSim that implements the network layer. As the use of cloud computing for different applications is increasing day by day, it is important to understand how these applications and systems work when deployed on the cloud. Because of the complexity of shared resources, it is often difficult to parse the performance of planning algorithms and the simulation of parallel and distributed applications in the cloud. Therefore, the importance of simulation tools to evaluate the cloud computing model is increasing. Nevertheless, CloudSim presents a very simplistic application models without any communicating tasks and provides the limited network model for the data center. Therefore, NetworkCloudSim is introduced, which is an extension of the famous cloud simulator CloudSim with a scalable network and application model. It provides accurate evaluation of scheduling. Developers with NetworkCloudSim can configure the most of parameters of different network topologies and the power efficient resource management.

Architecture: The important elements of NetworkCloudSim architecture are represented with dark boxes on Fig. 4. NetworkCloudSim supports the modeling of data center resources such as network and computing resources and a wide range of application models such as parallel application. Even a multi-layered web application can be modeled with little change. The evaluation results have shown that NetworkCloudSim can simulate cloud data networks and applications with high-precision communication tasks such as MPI.

3.1.3. CloudAnalyst

Description and main features: Wickremasinghe et al. [44] introduced CloudAnalyst simulator based on CloudSim to evaluate the large-scale distributed applications in cloud environment. The first goal of CloudAnalyst is that it separates the simulation

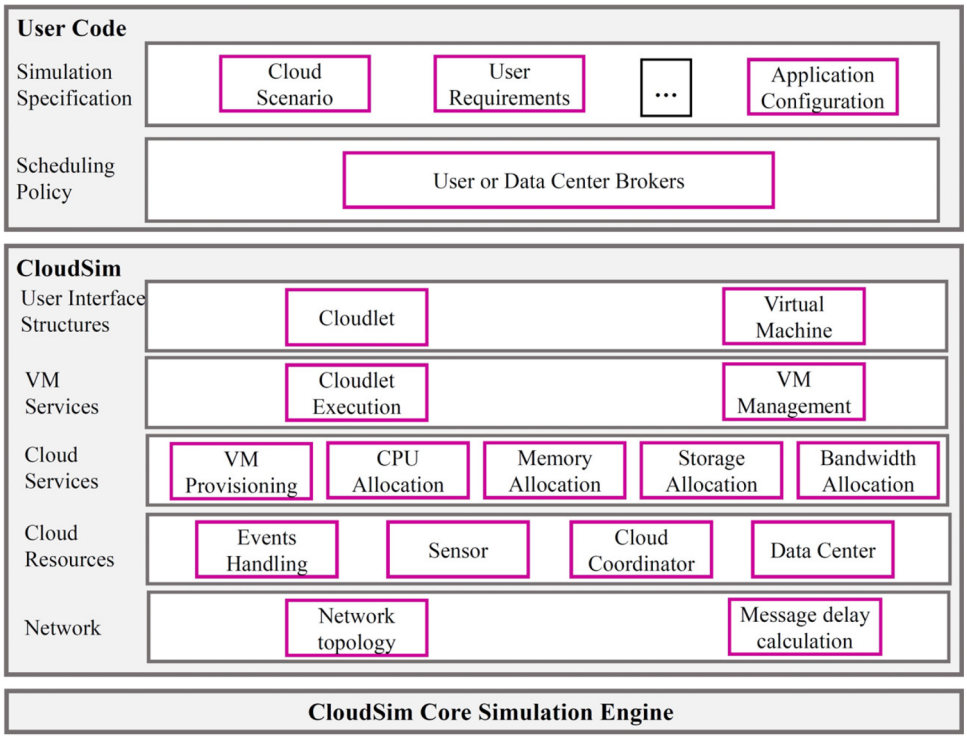


Fig. 3. CloudSim structure.

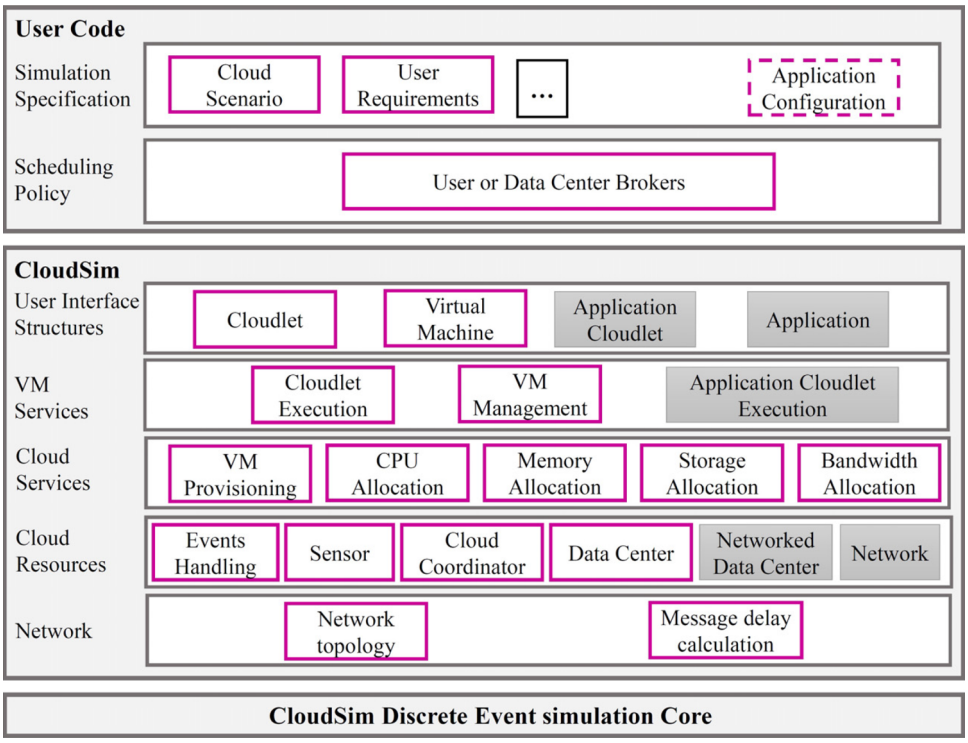


Fig. 4. NetworkCloudSim structure.

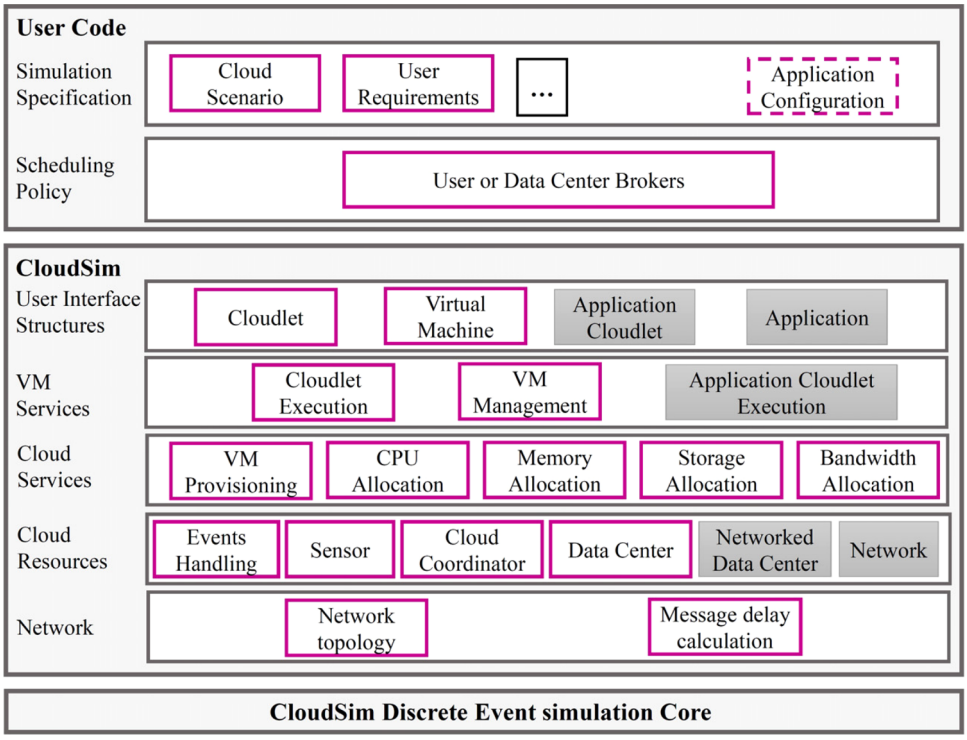


Fig. 5. CloudAnalyst structure.

experimentation exercise from a programming exercise. In other words, a developer can focus on the features of simulation without spending too much time on the problem of programming using a simulator. CloudAnalyst provides information such as the geographic location of users, location of data centers, number of users and data centers. Moreover, CloudAnalyst calculates the response time of requests, the processing time of requests, and other metrics based on this information. The CloudAnalyst saves the simulator information by "xml" file and output with "pdf" format.

Architecture: Fig. 5 indicates the CloudAnalyst structure that is developed on top of CloudSim simulators. Then, it adds some functionality for Internet application behavior. Finally, CloudAnalyst provides a graphical user interface to set up the parameters of experiments quickly and easily.

3.1.4. EMUSim

Description and main features: Calheiros et al. [45] proposed EMUSim that is an integrated emulation and simulation environment to evaluate the performance of cloud computing applications. EMUSIM are made on top of two software systems, namely automatic emulation framework (AEF) [46] for emulation and CloudSim for simulation. EMUSim is based on both simulation (to appraise

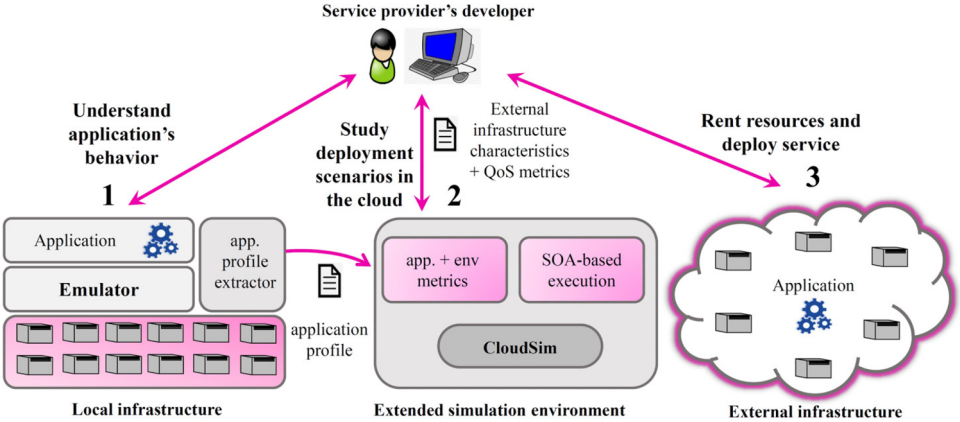


Fig. 6. EMUSim structure.

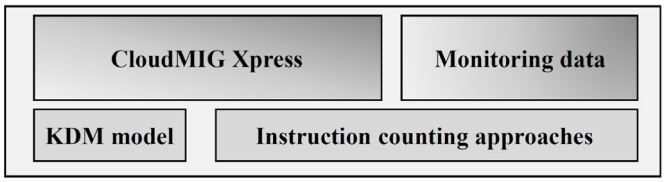


Fig. 7. CDOSim structure.

higher resource scale and changeable request workload) and emulation (to elicit professional information of the real application). EMUSIM automatically exploits the information from application behavior through emulation and then uses this information to produce a simulation model. In addition, EMUSim performs the evaluation using the information accessible to the customers of general IaaS providers. It does not need additional information like the number of VMs and their locations in a given time.

Architecture: The structure of EMUSIM is shown in Fig. 6 that has three main components: (1) Local infrastructure that investigates applications and their requirements, (2) Simulation that develops the scenario of cloud and executes the applications on the available host, and (3) External infrastructure that rends and deploys the services.

3.1.5. CDOSim

Description and main features: Fittkau et al. [47] introduced CDOSim toolkit to simulate cost and performance characteristics in cloud deployment. It integrates CloudSim simulator and CloudMIG [48] framework. In CDOSim, mega integer plus instructions per second (MIPIPS) unit is calculated as a new metric for the computing performance of data centers. It considers the mapping among services and the types of virtual machines and uses adaptation techniques such as assigning a new virtual machine if the utilization of CPU exceeds from specific threshold. Therefore, CDOSim tool accurately predicts the execution time for each service provider.

Architecture: The architecture of CDOSim is the integration of CloudSim components and CloudMIG Xpress [49] that is shown in Fig. 7. CloudMIG Xpress applies the extracted Knowledge Discovery Meta-Model (KDM) models and Monitoring data to create profiles and mapping models.

3.1.6. TeachCloud

Description and main features: Jararweh et al. [50] introduced a cloud computing educational toolkit (TeachCloud). The main challenge faced by cloud computing instructors is the lack of a training tool to test. The authors listed several key issues of teaching cloud computing in Table 5.

We can see the biggest challenge is the lack of hands-on experience. TeachCloud is a useful and easy-to-use simulation and modeling tool for cloud computing, which fills the gaps created by the lack of cloud computing education tools. TeachCloud is the generalization of CloudSim, a research-oriented simulator that is used to extend and validate cloud computing. TeachCloud also allows students to do experiment with the real cloud system at different cost condition. TeachCloud can use several cloud components such as data centers, storage, architecture of service oriented, constraints of service level agreement, networking, processing elements, business process management, virtualization, and web-based applications. TeachCloud also introduces the MapReduce [51] model and so it can process large datasets. Moreover, TeachCloud presents a graphical interface to build the customized network structures.

Architecture: Fig. 8 indicates the modules of TeachCloud. For providing network topology, there are two processes. The first one loads a previously saved structure and shows it in the Visual Designer Canvas. The second process applies Palette and Properties tool to build the network topology into Visual Designer Canvas. The list of attributes such as bandwidth and delay can be set in a properties window and finally the results of simulation are presented by different charts.

3.1.7. DartCSim

Description and main features: Li et al. [52] proposed DartCSim simulator based on CloudSim and hides the implementation details. The developers must know the source code well and write hundreds lines of code to perform a simplest strategy through CloudSim. Therefore, DartCSim defines a user-friendly interface and hence users can set the parameters of simulation such as cloudlets, network topology, and management algorithm with a visual interface. The multi-tiered architecture of DartCSim integrates the front-end and the backend. The simulation results that include results of data centers, cloudlets, and success rate are presented with auto-generated forms. DartCSim develops three types for configuration as basic type, power-aware network, and without power-aware network.

Table 5
Challenges of teaching cloud computing.

Challenge or Difficulty	Percentage agreed
Lack of hands-on experience	93%
Lack of a comprehensive textbook	63%
Lack of help material on the Internet	57%
Insufficient background	17%
Vast amounts of different topics	17%

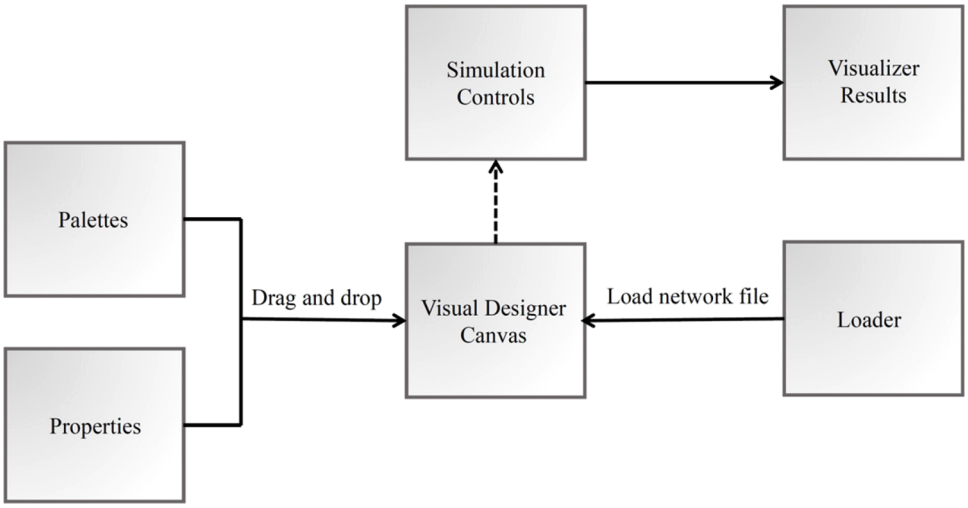


Fig. 8. TeachCloud network.

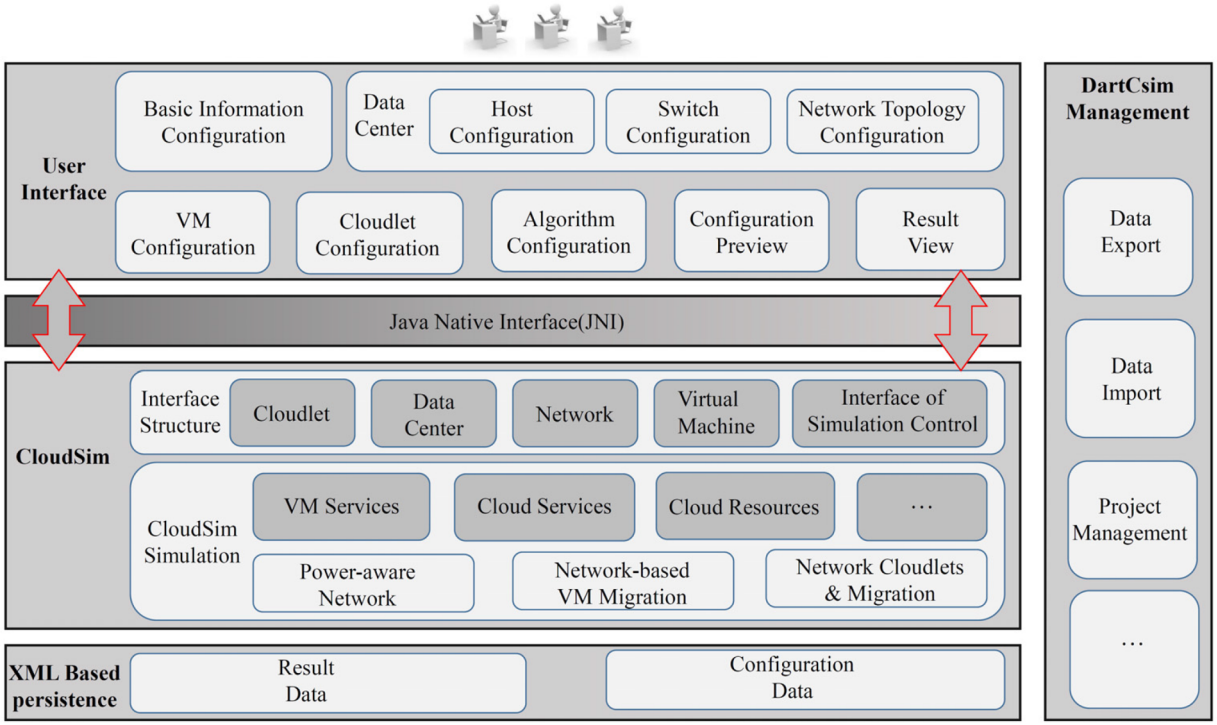


Fig. 9. DartCSim structure.

Architecture: Fig. 9 indicates the DartCSim architecture that includes three main modules: (1) User interface that shows the input of simulation and the results, (2) CloudSim module executes the simulation and contains three main parts (i.e., Power-aware network, Network-based VM migration, and Network Cloudlets & Migration), and (3) DartCSim Management that imports the specific data to the project, export, and store the results of simulation.

3.1.8. DartCSim+

Description and main features: Li et al. [53] suggested DartCSim+ that enhances CloudSim with integrating the power and network models. It tries to overcome the CloudSim constraints such as (1) CloudSim doesn't provide both the power model and the network model simultaneously, (2) the components of the CloudSim network doesn't provide the power awareness simulations, and (3) the migration simulation doesn't consider the costs of the entire network. DartCSim+ attempts to solve these problems by presenting the model of energy-aware network and the live migration of the aware network. In addition, DartCSim+ defines a resend mechanism to

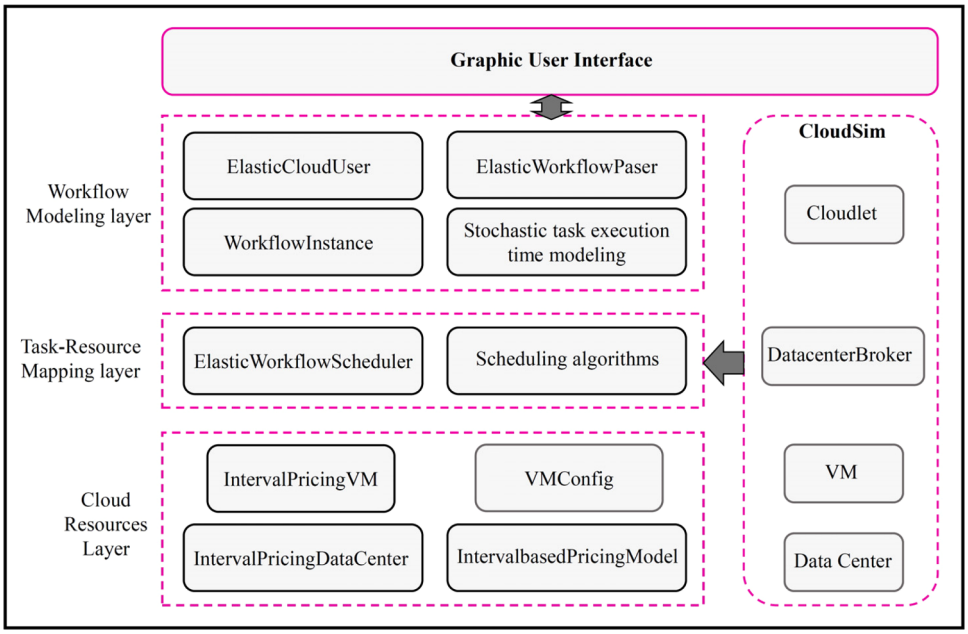


Fig. 10. ElasticSim simulator.

present a more realistic network model to resolve the failure of transmission. It contains network links and network interface cards to support a more comprehensive and practical network framework. In addition, it enhances the accuracy of simulation by modeling the network overhead.

Architecture: The structure of DartCSim+ is like CloudSim but it adds newly added network entities to support application programming interfaces (APIs). Therefore, users can set all transmission links and NICs before execution and get the power consumed by corresponding components.

3.1.9. ElasticSim

Description and main features: Cai et al. [54] presented ElasticSim simulator that extends CloudSim by considering the resource runtime auto-scaling. In real environment, the execution times of tasks with various properties show various probability distributions. The provisioning of resource according to the stochastic execution times and interval based cloud-pricing model is difficult. ElasticSim supports the impacts of the task execution time probability distribution and the tightness of workflow deadlines on the scheduling strategies. In addition, it designs a delegate method that determines the procedure of transforming the schedules of static strategies into practical VMs renting. Finally, ElasticSim has a graphical user interface to show the execution state in real time.

Architecture: Fig. 10 indicates the structure of ElasticSim that has three main layers: (1) Workflow modeling that manages the dependencies of workflows and simulates the stochastic task execution times, (2) Task-Resource mapping that provides the runtime resource auto-scaling approach and schedules workflows, and (3) Cloud Resource that defines VMs with interval based pricing models.

3.1.10. FederatedCloudSim

Description and main features: Kohne et al. [55] proposed FederatedCloudSim that is a SLA-aware federated Cloud simulator. The main goal of FederatedCloudSim is to test various types of cloud federations. Therefore, the authors extended the basic operations of CloudSim by adding some new functionality that covers SLAs, workload generation, event logging, scheduling and brokering. In the current implementation, SLA is presented based on the deadline and the warranted resources. But, the service level objectives of SLA can be easily determined through the configuration level of the cloud. The service level objectives of SLA can be monitored during execution and if necessary, the rescheduling of VMs is triggered. For example, if one resource becomes a bottleneck due to over-provisioning then VMs can be replaced or transferred.

Architecture: In comparison with the original CloudSim simulator, the top level of FederatedCloudSim is new and decides that tasks are run locally or transferred to a remote cloud within the federation.

3.1.11. FTCloudSim

Description and main features: Zhou et al. [56] presented FTCloudSim toolkit for modeling different the service reliability enhancement methods. For investigating the performance of each approach, FTCloudSim triggers failure events and provides some performance metrics. It presents four reliability enhancement methods: (1) no reliability method, (2) a checkpoint technique, (3) an incremental checkpoint technique, and (4) an incremental checkpoint and the image storage node is chosen randomly.

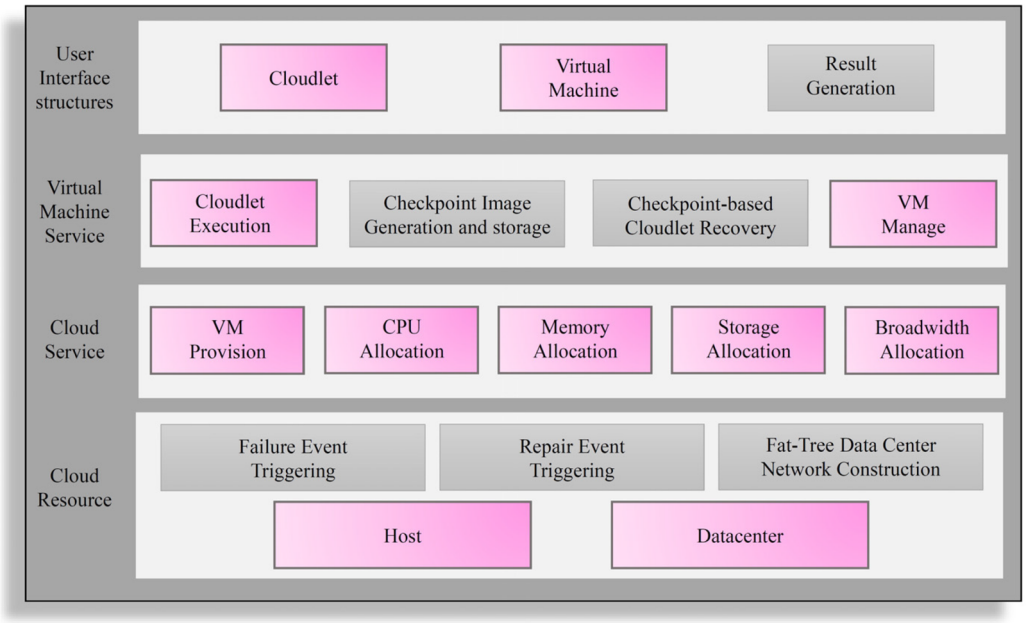


Fig. 11. FTCloudSim structure.

Architecture: Fig. 11 indicates the structure of FTCloudSim that has four additional capabilities compared with CloudSim: (1) Fat-tree data center network construction that is a common architecture in the current commodity data center, (2) Failure and repair event triggering that builds some failure events based on different distributions such as Weibull [57] distribution and exponential distribution, (3) Checkpoint image generation and storage that creates and stores checkpoint image periodically according to the different methods, (4) Checkpoint-based cloudlet recovery that resumes a task that is failed according to the latest available checkpoint image, and (5) Results generation that shows the simulation results based on some metrics such as total execution time, average lost time, network resource usage, storage usage, and total checkpoint image data that is transferred.

3.1.12. *WorkflowSim*

Description and main features: Chen and Deelman [58] introduced WorkflowSim for modeling Scientific Workflows in cloud environment. Workflows in heterogeneous distributed systems show different levels of overheads that are explained based on computational operations and miscellaneous works. Most of simulators such as CloudSim do not consider fine granularity simulations of

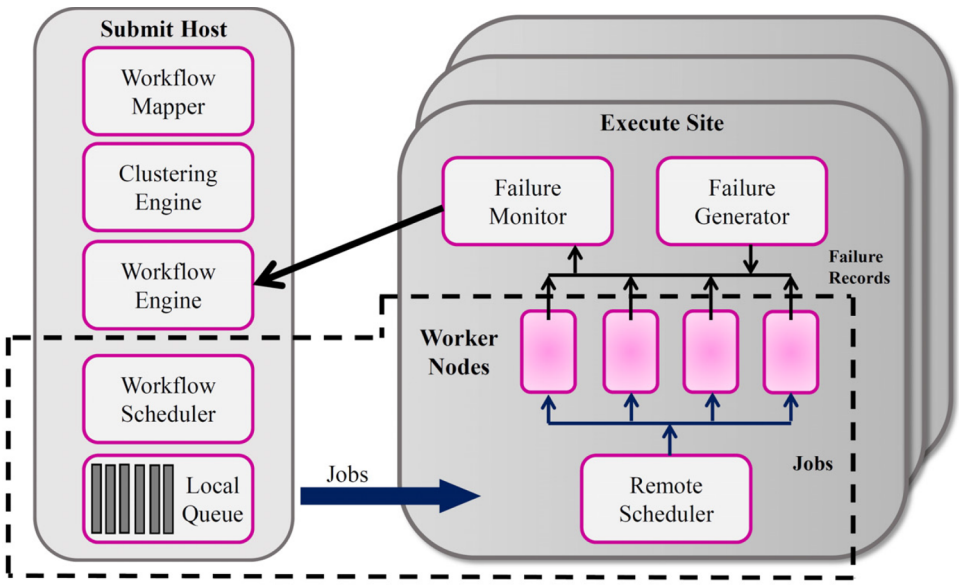


Fig. 12. WorkflowSim structure.

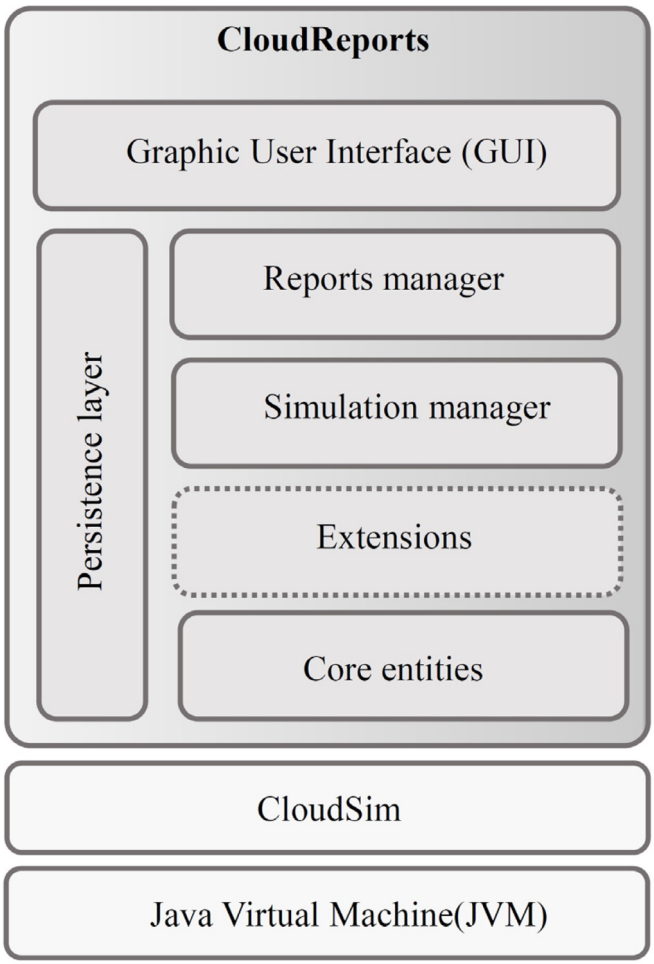


Fig. 13. CloudReports structure.

workflows and task clustering. Task clustering reduces the number of jobs to be executed and so execution overhead is decreased. Generally, a job may have high risk for suffering from failures since it consists of several tasks. Researchers with WorkflowSim can study the impact of job failures runtime performance of workflows for different clustering methods.

Architecture: Fig. 12 shows the structure of WorkflowSim that is composed of several components: (1) Workflow Mapper that maps abstract workflows to concrete workflows, (2) Workflow Engine that controls the data dependencies, (3) Workflow Scheduler assigns jobs to resources, (4) Clustering Engine that groups small tasks into a large job, (5) Failure Generator that injects task failures for each execution site, (6) Failure Monitor that stores the failure information such as resource id and task id.

3.1.13. CloudReports

Description and main features: Teixeira Sá et al. [59] suggested CloudReports as an extendable simulator for energy-aware cloud environments. CloudReports extends CloudSim by some functionalities: (1) presents a graphic user interface, (2) automatically organizes simulation results, (3) simplifies the configuration of system, and (4) customizes user behaviors by setting the VM and data center specifications. At the end of simulation, CloudReports presents a complete report that includes the log of operations. It also draws different charts with detailed information for resources usage, virtual machine allocations, execution of cloudlets, and energy consumption of data center. Moreover, the additional generated files and output data are exported to the third-party applications (e.g., MATLAB and Octave).

Architecture: Fig. 13 presents the structure of CloudReports that is composed of five components: (1) Core entities that defines the fundamental elements like users, data centers, physical machines, virtual machines, networks, and storage area networks, (2) Extension shows the user-implemented code, (3) Simulation manager that transforms the environment created through the Graphical User Interface (GUI) into CloudSim components that are used during simulation, (4) Report manager that processes all simulation data to draw multiple charts, and (5) GUI allows the manipulation of different elements (e.g., data centers, hosts, storage area networks, users, VMs, and network links) and also developers can determine the costs of operation, scheduling provisioning methods, and resource utilization models.

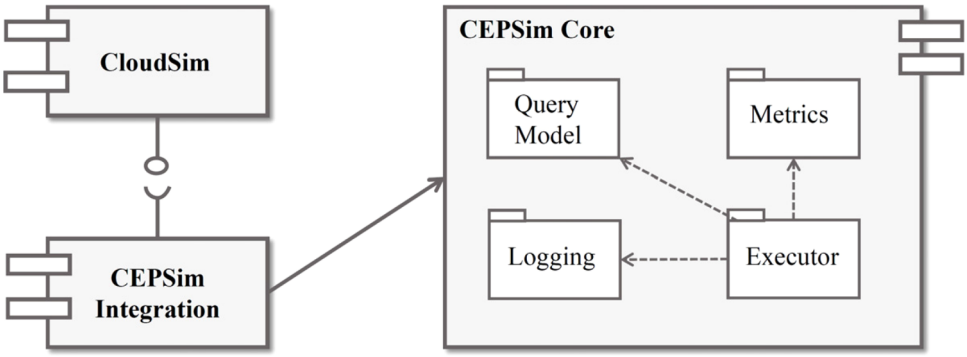


Fig. 14. CEPSim structure.

3.1.14. CEPSim

Description and main features: Higashino et al. [60] proposed CEPSim toolkit for complex event processing in cloud environment. Complex Event Processing (CEP) was introduced by the work of Luckham on Rapide [61]. In CEP systems, users define queries by means of proprietary languages like Aurora Stream Query Algebra (SQuAl) [62] and CQL [63]. CEPSim adds a new model by the directed acyclic graphs (DAGs) to CloudSim. It tries to show continuous queries processing fast streams of data and execute these queries in various systems (i.e., including private, public, and multiple). Moreover, various scheduling algorithms and operator placement policies can be customized in CEPSim.

Architecture: Fig. 14 indicates the structure of CEPSim. The three main components are: (1) CEPSim Core that handles most of the CEP-related logic, (2) CloudSim that controls the simulation infrastructure and execution, and (3) CEPSim Integration that integrates the other components.

The CEPSim Core consists of four main modules: (1) query model that describes queries, (2) query executor that performs the most of logic in the simulation, (3) logging that presents the logging facilities of simulation, and (4) metrics that calculates the parameters of simulator.

3.1.15. DynamicCloudSim

Description and main features: Bux and Leser [64] presented DynamicCloudSim that extends CloudSim by handling heterogeneity, failure, and dynamic changes at runtime. CloudSim assigns new VMs to the host based on the most available resources, but DynamicCloudSim considers a random machine within the datacenter. Therefore, there is probable that virtual machines with equal configurations are assigned to different types of physical machines and so present varying amounts of computational resources. Moreover, DynamicCloudSim models the external loads that are created due to sharing common resources with other machines and applications. Finally, DynamicCloudSim provides straggler VMs and failures to model fault-tolerant approaches.

Architecture: DynamicCloudSim has a structure like CloudSim with additional models such as inhomogeneity in computational resources, slight aberrations to a VM's performance, and defining the rate of failure during task execution.

3.1.16. CloudExp

Description and main features: Jararweh et al. [65] proposed CloudExp simulator to address virtualization and business process management in cloud system. CloudExp tries to overcome some shortcomings of CloudSim such as: (1) CloudSim is developed on top of a grid environment, which provides some limitations for the simulated infrastructures, and (2) CloudSim only simulates a simple network model with limited workload traffic structure. CloudExp provides the generator of Rain [66] workload to model real workloads and integrates MapReduce processing model to process huge data files. Developers with CloudExp can make a SLA in a simple and effective manner based on the measurable terms (e.g., number of users, service availability, service cost, service outage handling terms, business continuity and disaster recovery, network performance, and security measures). Moreover, CloudExp has suitable GUI for setting cloud configurations and showing results with charts. It provides new network models like VL2, BCube, Portland, and DCell to show real topologies for cloud. Finally, this simulator develops Mobile Cloud Computing (MCC) simulation framework.

Architecture: The architecture of CloudExp is like CloudSim but it adds six functionalities: (1) workload generator, (2) modules for MapReduce, (3) modules for mobile cloud computing, (4) extensions for SLA, (5) wide range of network topologies, (6) Graphical User Interface (GUI).

3.1.17. CM Cloud

Description and main features: Alves et al. [67] proposed a cost module for Cloudsim (called CM Cloud). CM Cloud expands the function of Cloudsim by supporting various cost models. CM Cloud can design any cost model using XML and support current cloud service providers such as Google, Microsoft Azure, and Amazon by retrieving values directly from their web pages dynamically. CM Cloud estimates the overall cost of the simulation and is able to measure the financial costs of the system and compare the results with different cloud providers. It also allows IT companies and researchers to determine the best cost-effective strategy for deploying the

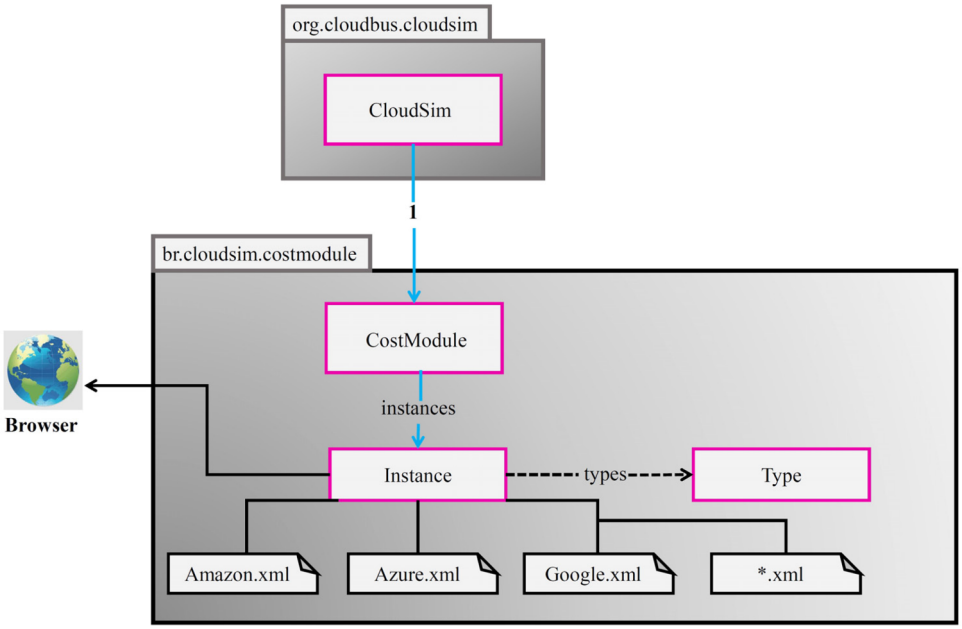


Fig. 15. CM Cloud structure.

cloud and choosing the best provider for deploying applications.

Architecture: CM Cloud structure is presented in Fig. 15 and consists of three main modules: (1) CostModule that executes the cost models, (2) Instance that is created to access cost models from the providers of services or customized cost models, and (3) Type that determines the types of instances.

3.1.18. MR-CloudSim

Description and main features: Jung and Kim [68] proposed MR-CloudSim that focuses on MapReduce computing model on CloudSim. MapReduce model is one of the most widely used computational models for using a server cluster. Since CloudSim does not provide a good view of the size and contents of files, implementing the MapReduce model needs to modify or enhance the function in CloudSim. In addition, it is very common for large data processing.

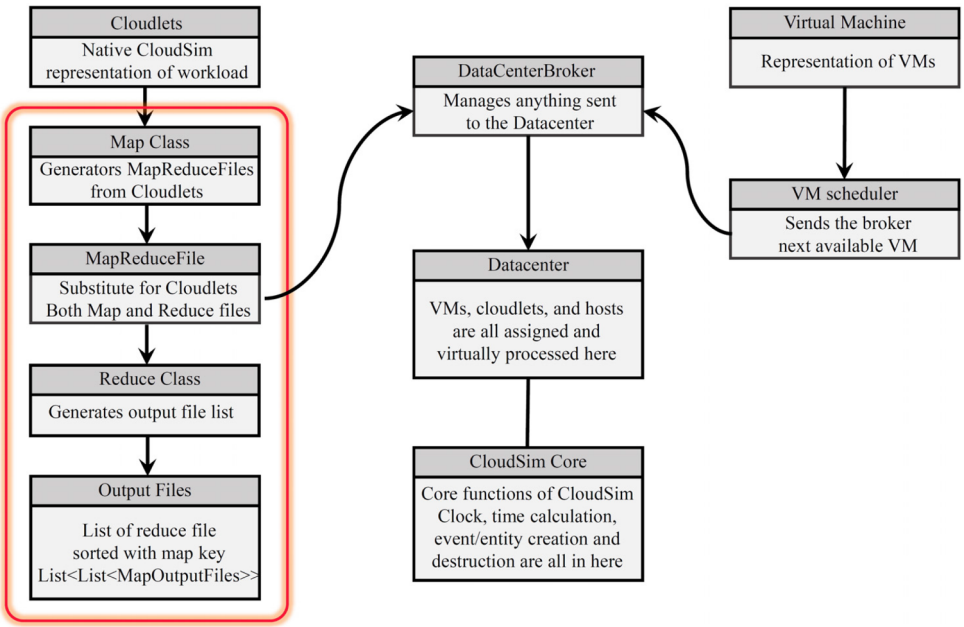


Fig. 16. MR-CloudSim structure.

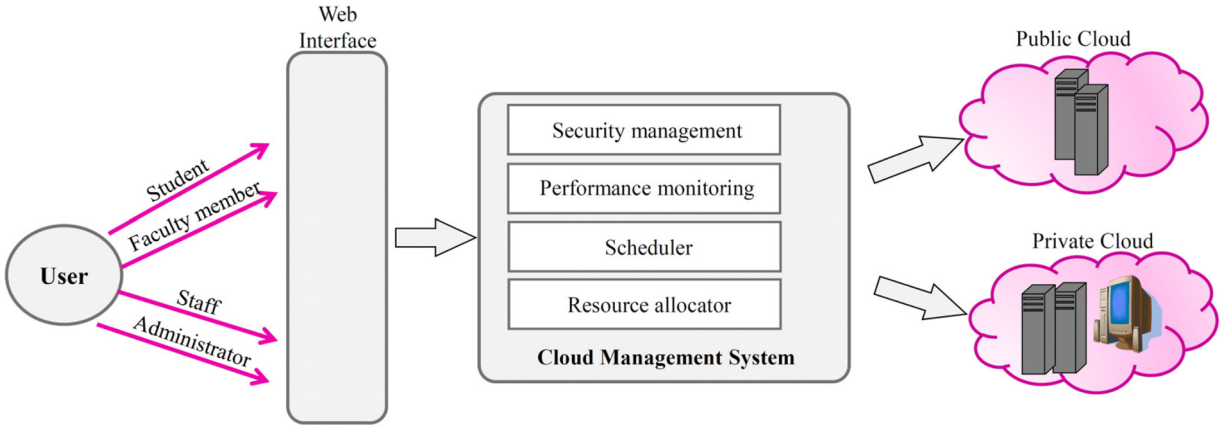


Fig. 17. UCloud structure.

Architecture: Fig. 16 indicates the overall structure of classes for MR-CloudSim. Map class gets cloudlets and divides them into a number of MapReduceFiles. MapReduceFile class defines cloudlet with the additional characteristics (i.e., key, map/reduce maker, and pair parent pointer) as input for MapReduce. Finally, Reduce class sorts the marked MapReduceFiles according to their keys.

3.1.19. UCloud

Description and main features: Sqalli et al. [69] modeled a hybrid cloud for a university environment and called it UCloud. The architecture used here is based on a hybrid cloud model that uses both public and private clouds and is developed using CloudSim.

Architecture: UCloud structure (Fig. 17) has two main elements: cloud management system and hybrid cloud.

UCloud management is in charge of the cloud management system (CMS) and includes some functions such as performance monitoring, university activities, security management, etc. This information can be used for resource allocation in the future. Because universities need private cloud for very secure services and need public cloud for performance and scalability issues, hybrid cloud provides the suitable environment for universities. It should also distribute the complexities of specifying applications and services to the private cloud, the public cloud, or both. Cloud computing can be used for universities. The hybrid cloud model is used here for universities environments. The experiments proved that the public cloud is not appropriate all the time. When the private cloud utilization reached its maximum then public cloud is applied.

3.2. Other cloud simulators

3.2.1. MDCSim

Description and main features: Lim et al. [70] suggested a Multi-tier Data Center Simulation (MDCSim) platform to design and analysis of large-scale system. It provides a flexible and scalable simulation platform for in-depth analysis of multilayer data centers. A three-layer prototype data center is also designed with Infiniband Architecture (IBA) architecture. Based on the RUBIS criterion

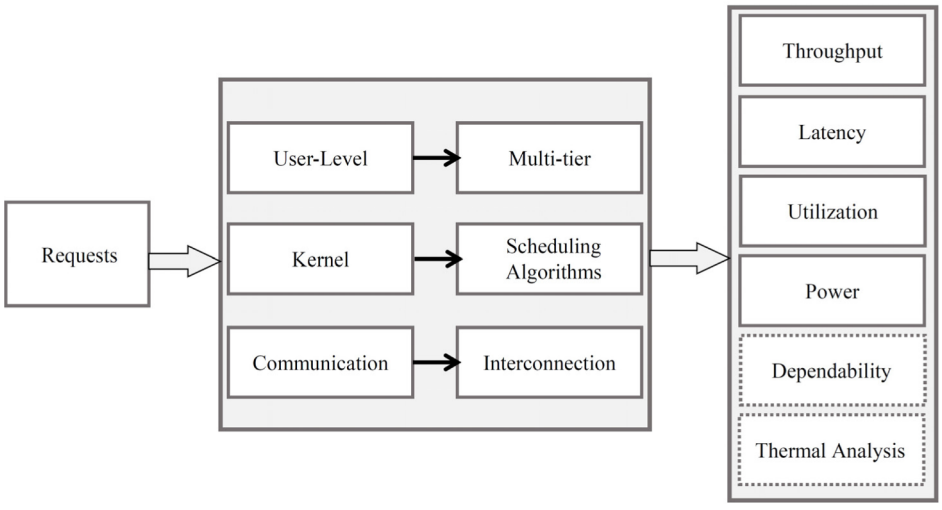


Fig. 18. MDCSim structure.

Table 6
Simulation tools and their features of a holistic data center simulator.

Simulation Tools/Methods	Automated processing	Online analysis	Iterative design	Thermal analysis	Workload management	Cyber-Physical interdependency
Online thermal aware workload management	✗	✓	✗	✓	✓	✓
Dynamic CRAC control	✗	✓	✗	✓	✗	✓
Offline thermal aware workload management	✗	✗	✓	✓	✓	✓
SLA and power analysis	✓	✓	✗	✗	✗	✗
CFD Simulation	✗	✗	✓	✓	✗	✗
GDCSim	✓	✓	✓	✓	✓	✓

[71], it is shown that the simulator is very accurate in estimating power, response time, and power consumption.

Architecture: The MDCSim simulator is designed as interchangeable three-level architecture (Fig. 18) that performs simulation in three layers (i.e., communication layer, kernel layer, user-level layer). One of the main features of MDCSim is that it captures all of the main contextual pattern design specifications, kernel-level planning artifacts, and program-level interaction between layers of a three-layer data center.

3.2.2. GDCSim

Description and main features: Gupta et al. [72] proposed a new simulator (GDCSim) to analyze green data center and resource management techniques. Recently, researchers focus on activities in data center design such as proper management of resources, thermal awareness, and cooling units to reduce energy consumption. In addition, GDCSim is validated against CFD simulators [73]. GDCSim is expanded as part of the BlueTool (BlueTool is a computer infrastructure project funded by NSF). The purpose of this project is to provide suitable research infrastructures in both hardware and software to raise the level of awareness of the environmental importance of data centers operating worldwide.

Table 6 summarizes some holistic simulation tools with important features. From Table 6, we can observe that most of researches consider the online analysis of workload management methods and the thermal effects. The main challenge in data center analysis is the lack of a comprehensive simulator that can test the impact of new computational resource management techniques with different data center designs. Therefore, to fill this gap, GDCSim is introduced to evaluate the energy efficiency of data center and simulate the physical behavior of a data center. Moreover, it captures the dependency between online resource management programs.

Architecture: The architecture of GDCSim is shown in Fig. 19. It has four basic elements: (1) BlueSim that generates various

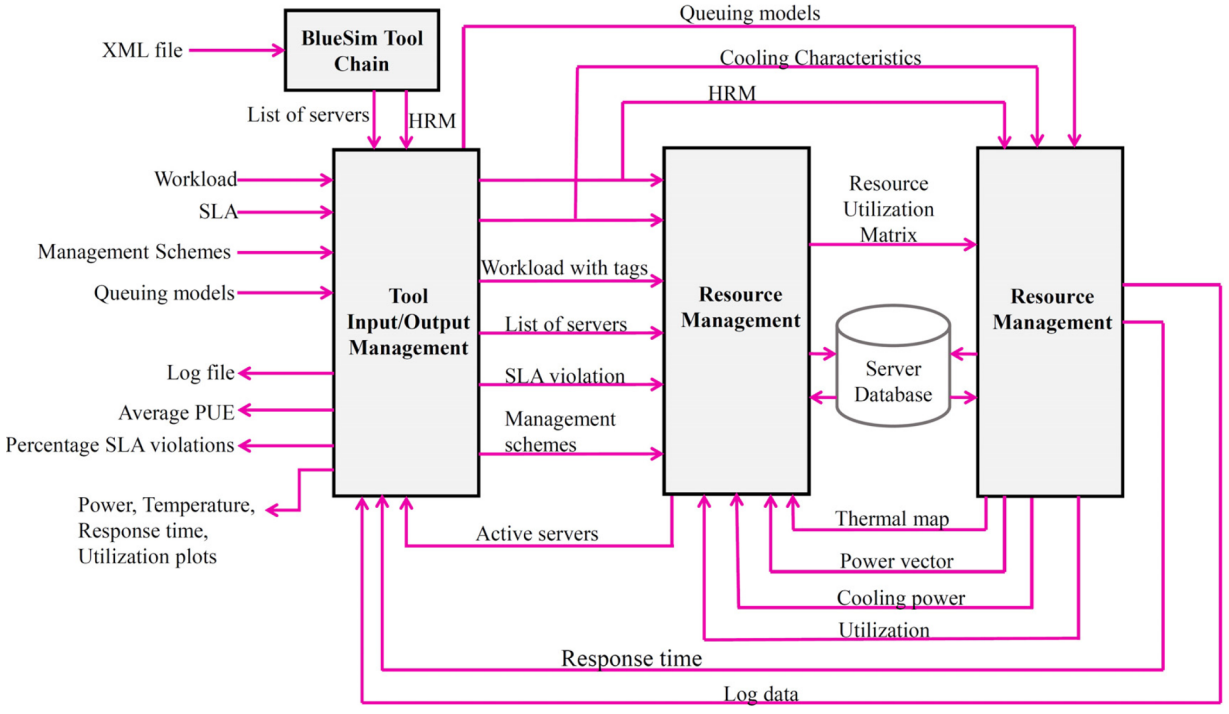


Fig. 19. GDCSim structure.

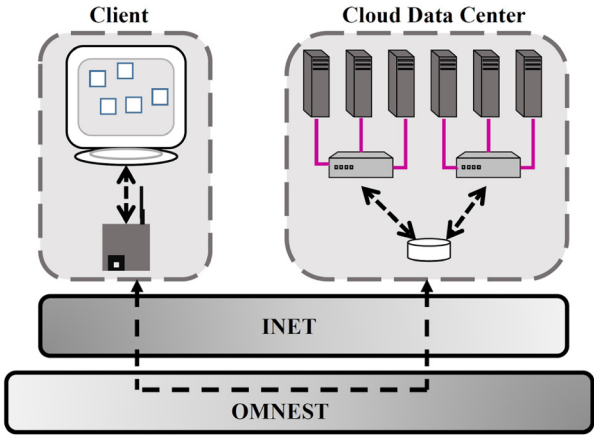


Fig. 20. CloudNetSim structure.

configurations for data centers, (2) Input Output Management (IOM) that specifies the characteristics of the workload, (3) Resource Management (RM) that consists of workload management, power management, and cooling management, and (4) Simulator that includes four modules as Queuing, Thermodynamic, Power, and Cooling.

3.2.3. CloudNetSim

Description and main features: Cucinotta and Santogidis [74] presented CloudNetSim simulator to provide the management of resource and scheduling algorithms in cloud environment. It models end-to-end network communications among clients and servers. The storage access model of CloudNetSim is simple and can be added from other modules likes SIMCAN. CloudNetSim introduces CPU scheduling for hypervisor and at the guest OS levels. Moreover, it presents VM deployment and scheduling algorithms with application models. It can model thousands of nodes with important QoS metrics.

Architecture: The overall architecture of CloudNetSim is shown in Fig. 20. The computing parts of the simulation are modeled on OMNEST. Then, the network communication modules and CPU scheduling can be comprehensive emulated.

3.2.4. CloudNetSim++

Description and main features: Malik et al. [75] presented a toolkit for data center simulations in OMNET++ and called it CloudNetSim++. A modeling tool is introduced to simplify the simulation of distributed data center architecture, energy models, and high speed data center communication network. CloudNetSim++ is the first cloud computing simulator to use actual physical properties of the network to model the distributed data centers. Researchers with CloudNetSim++ can combine their own applications and analyze them under realistic network architectures with traffic template. The salient features of this simulator includes: Providing a public framework that allows users to describe SLA policy, module, and scheduling algorithms for different data center components without worrying about the bottom level of detail, ability to simulate data centers with high speed communication links geographically, and adding data center resources such as routers, computing servers, and switches is easily done through the user interface.

Architecture: Fig. 21 indicates the CloudNetSim++ architecture that consists of five modules: (1) Pricing Policy Manager that computes the billing cost for each user request based on the agreement, (2) Cloud Usage Monitor that analyzes usage patterns, (3) Task Scheduling Selection Module that determines the scheduling policy, (4) VM Manager that determines VM assignments according to the received SLA requests, and (5) User Task Scheduler that gets all the incoming user requests and distributes them to the appropriate VMs. Moreover, CloudNetSim++ introduces a rich GUI to simplify debugging and analyzing the results.

3.2.5. GreenCloud

Description and main features: Kliazovich et al. [76] proposed a packet-level simulator called GreenCloud for energy-aware cloud computing data centers. Unlike common toolkits such as CloudSim or MDCSim [70], GreenCloud uses, aggregates, and processes information of the energy consumption in cloud data centers. GreenCloud is an extension to the network simulator NS2 [77] and presents a fine-grained model for energy consumed through various networking components such as servers, switches, and links. Additionally, GreenCloud takes into account the distribution of workload.

Architecture: The architecture of GreenCloud with three-tier data center structure is shown in Fig. 22. It consists of three layers: (1) access, (2) aggregation, and (3) core layers. The aggregation layer can facilitate increasing number of servers (e.g., 10,000 servers) while keeping inexpensive switches (layer 2) in the access network and so a loop-free structure is provided.

3.2.6. iCanCloud

Description and main features: Núñez et al. [78] introduced iCanCloud simulator for cloud infrastructures that is written in C++. It considers the different instance types of Amazon. It contains two main sections for configuration. Firstly, the cloud model definition is

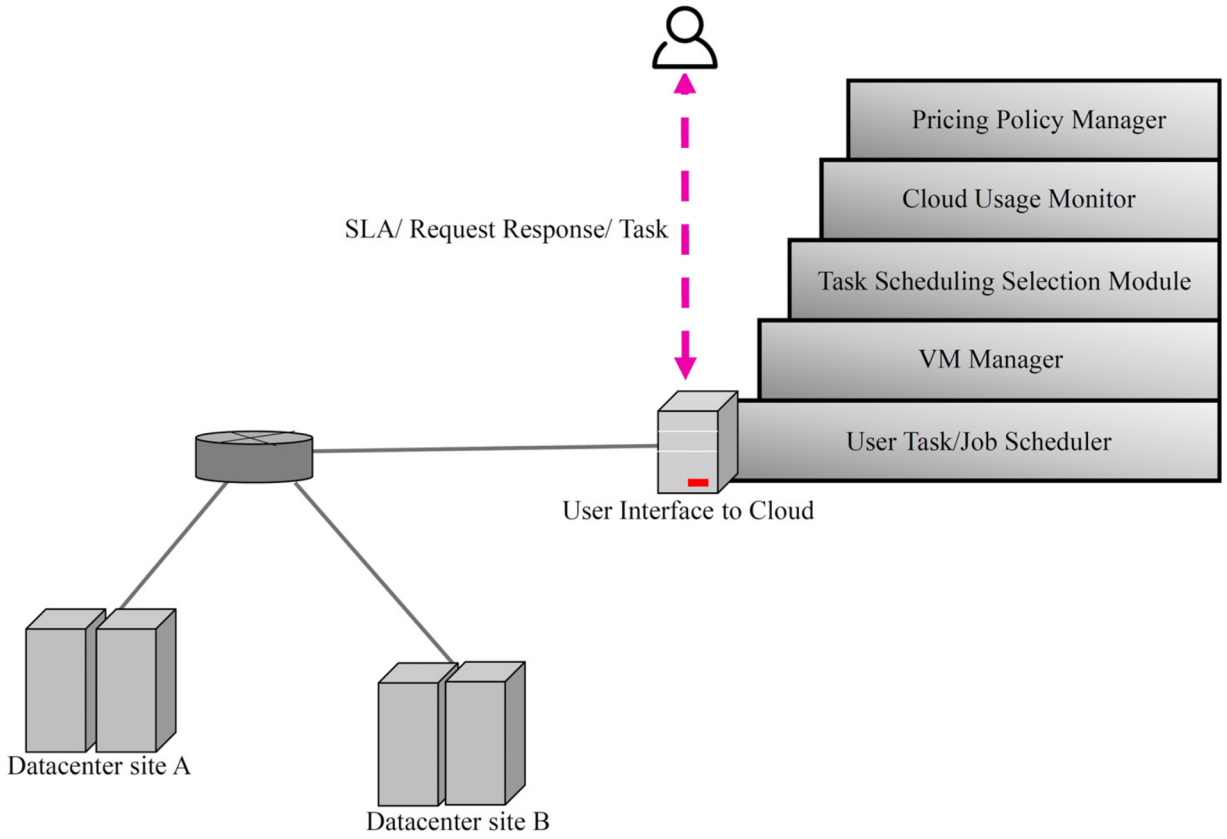


Fig. 21. CloudNetSim++ structure.

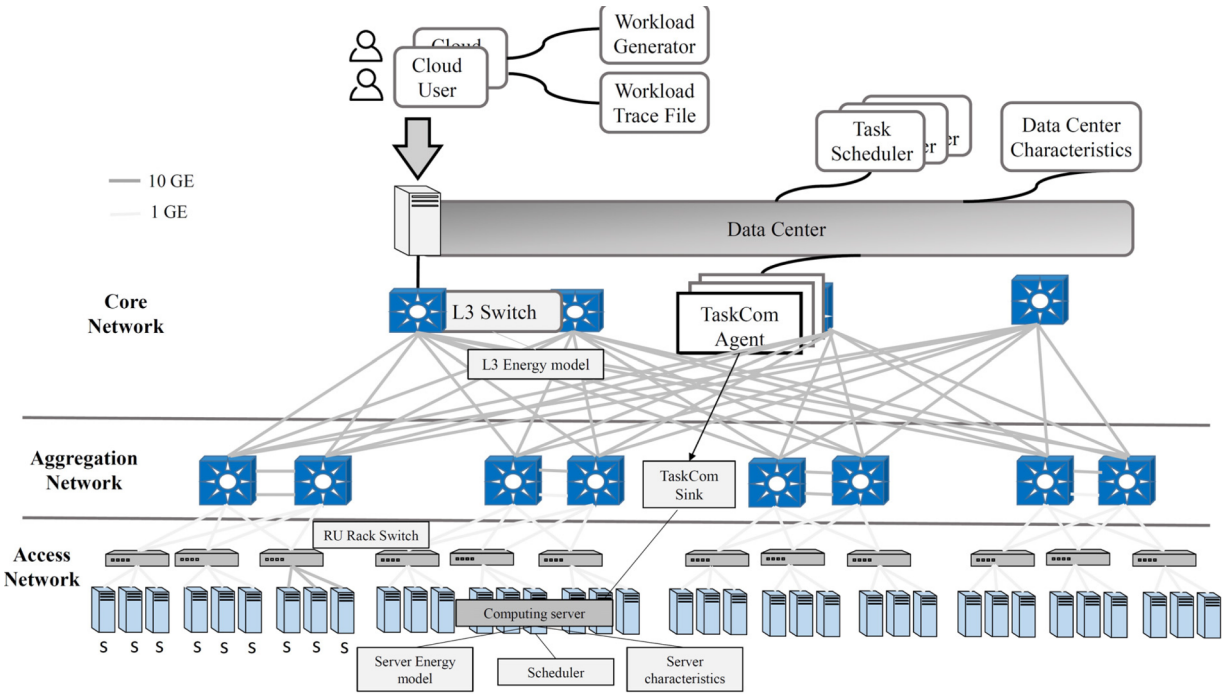


Fig. 22. GreenCloud structure.

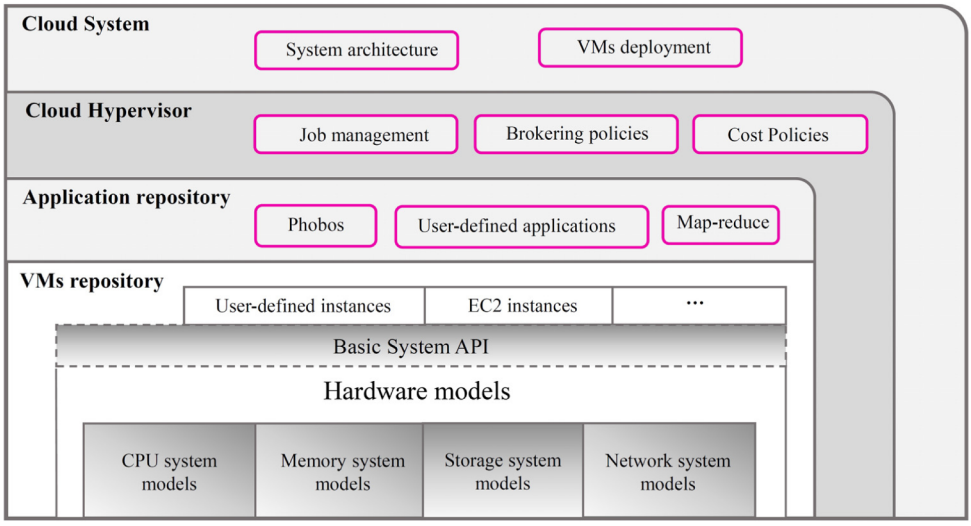


Fig. 23. iCanCloud structure.

for setting the characteristics of cloud system. Secondly, the user configuration that defines the users of cloud. The main advantage of iCanCloud is that it presents a complete hypervisor for different brokering strategies. Moreover, it can model large size cloud environments with parallel techniques and hence one experiment can be run on several machines.

Architecture: The iCanCloud structure is presented in Fig. 23. (1) Hardware layer that models disk drives, memory, and CPU, (2) Basic system's API layer that defines the interface among applications and the services of hardware models, (3) VMs repository consists of a set of VMs that are defined by user, (4) Cloud hypervisor that manages all tasks and VMs for execution, (5) Cloud system that contains VMs to build the entire cloud system.

3.2.7. secCloudSim

Description and main features: Rehman and Anwar [79] introduced secCloudSim as a secure cloud simulator. The secCloudSim extends iCanCloud toolkit to provide the basic security mechanisms and model the security related experiments. It considers the Challenge-Handshake Authentication Protocol (CHAP) [80] for user's authentication. With Access Control List (ACL), the privileges of authentic user are investigated and the user obtains the authorized for applying the corresponding services as per ACL rights. secCloudSim provides a framework that researchers can develop the security characteristics such as encryption, decryption, encapsulation, authentication and privacy assurance.

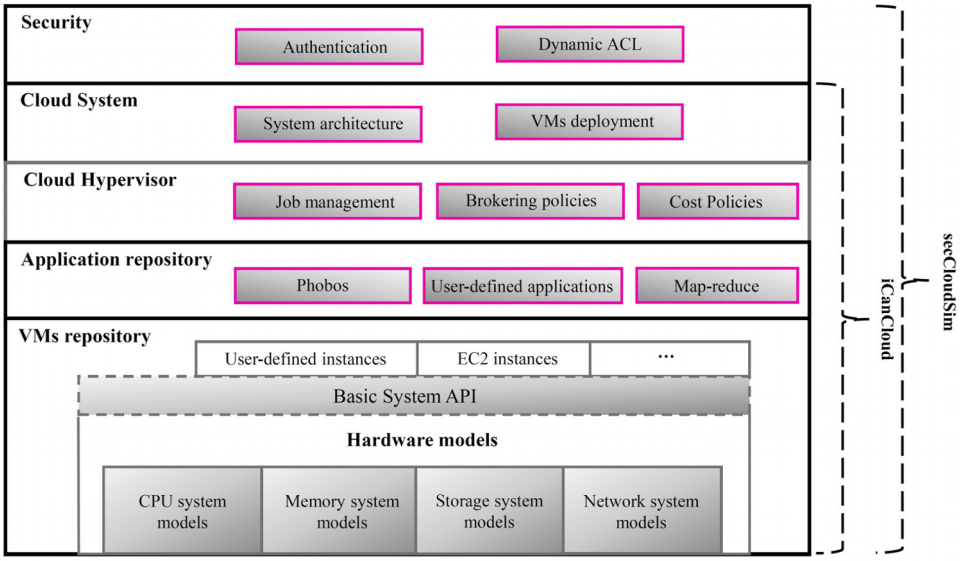


Fig. 24. secCloudSim structure.

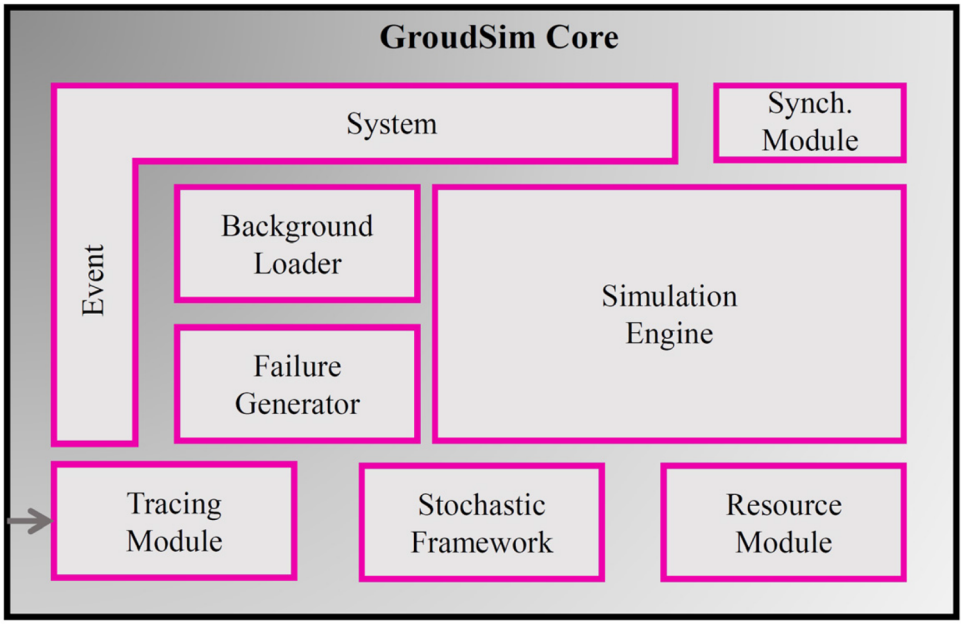


Fig. 25. GroudSim structure.

Architecture: Fig. 24 indicates the structure of secCloudSim that is based on iCanCloud-layered architecture. The newly developed Security layer of secCloudSim includes two components: (1) Authentication Module that employs CHAP to authenticate the users and allow them to apply the cloud services, and (2) Dynamic Access Control List Module contains a list of users, list of cloud services like VM, network system, storage, computational power, and the access rights of these services. The permission sets of rights can be varied for each user based on requirements.

3.2.8. GroudSim

Description and main features: Ostermann et al. [81] proposed GroudSim simulator for both Grid and Cloud environments according to the simulation-independent discrete-event core. This Java-based simulation toolkit can offer the network resources of cloud and grid, task submissions, two cost models, tracing mechanism, integration of failure, and background load. GroudSim can provide both real and simulated executions of real-world programs since it is added as a back-end in the ASKALON Grid computing environment.

Architecture: Fig. 25 shows the GroudSim structure that consists of the provisioning of VMs, the execution of jobs on VMs, file transfer, failures at different levels, cost calculation, and background load.

3.2.9. CloudSched

Description and main features: Tian et al. [82] introduced CloudSched simulator to present various resource scheduling strategies in cloud. These strategies take into account CPU, storage and the network bandwidth of physical machines and virtual machines to avoid bottlenecks. It includes four methods for energy-efficiency as Round Robin, Modified Best Fit Decreasing, Offline without delay, and Online without delay. CloudSched generates the distribution of service time, arrival process, and request distribution by random function. Moreover, external benchmark workload like LLNL data [83] can be integrated with CloudSched.

Architecture: Fig. 26 shows the architecture of CloudSched that has three main layers: (1) User Interface that provides interface for choosing resource and sending requests, (2) Core layer of scheduling that assigns user requests to suitable physical machines, (3) Cloud Resource that consists of physical machines and virtual machines with the specific amount of CPU, memory, storage, and bandwidth, etc.

3.2.10. SimIC

Description and main features: Sotiriadis et al. [84] presented a new inter-cloud simulation platform called SimIC that develops the setting of inter-cloud based on the exchange user requirements. SimIC provides optimization for different performance metrics of entities. The main characteristic of SimIC is the automation of service distribution that is varied among decentralized meta-brokers. These are defined on the top of each cloud for communicating with others in a distributed topology such as grid computing. In addition, it designs reactive orchestration according to the available workload of different user specifications. In SimIC, a request is assigned to the data center that can execute it based on the service contract. Therefore, SimIC provides service elasticity and interoperability for cloud environment.

Architecture: Fig. 27 shows the structure of SimIC that composed of three main components: (1) intra-cloud elements (e.g.

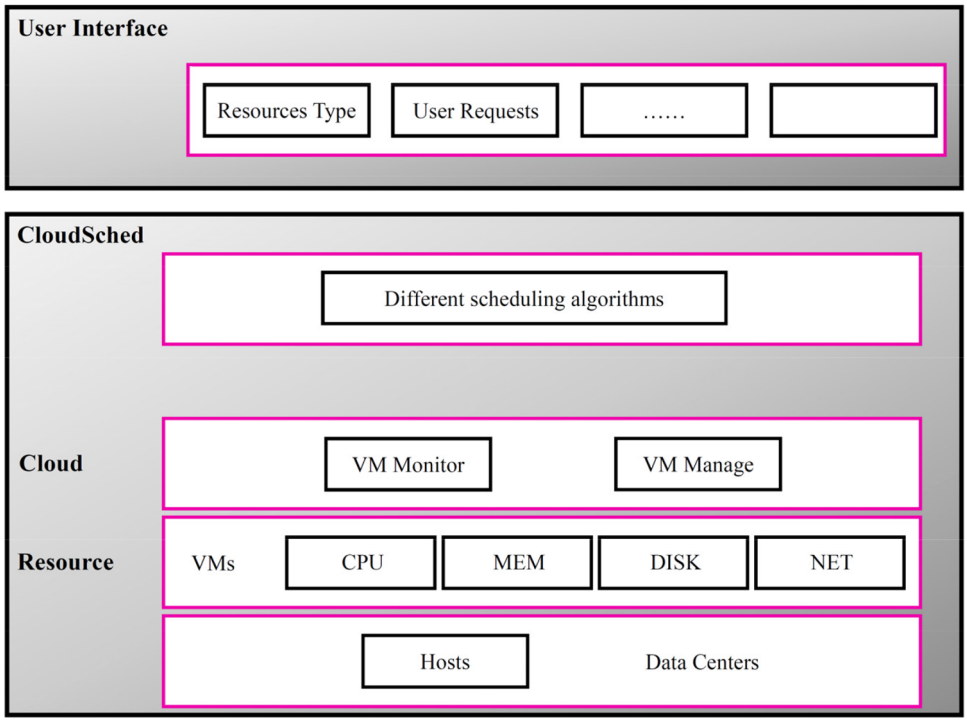


Fig. 26. CloudSched structure.

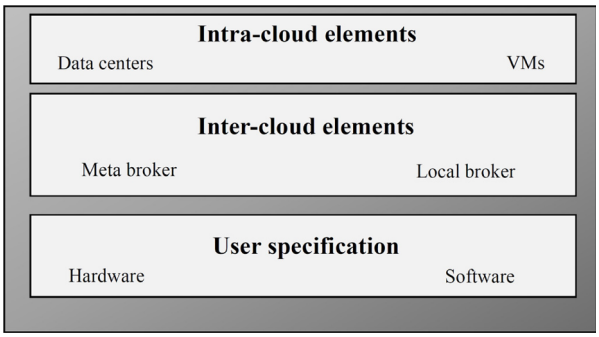


Fig. 27. SimIC structure.

datacenter), (2) inter-cloud elements (e.g. meta-brokers or decentralized resource managers, and (3) Additional functionality (e.g. service distribution, importing user specifications, exporting the results of simulation, and drawing simulation charts. Moreover, the framework of SimIC has been implemented based on the segmental format and so its developers can easily adapt components, edit their number and relationships, design different allocation algorithms by extending available schedulers or creating new policy.

3.2.11. SPECI

Description and main features: Sriram [85] introduced a Simulation Program for Elastic Cloud Infrastructures "SPECI" to explore the cloud-scale data centers. SPECI is a simulation tool that allows users to discover the scaling aspects of the big data centers as well as the performance features of future data centers. The main goal of this toolkit is to model the scalability and performance aspects of data centers according to the size and middle design policy. In addition, SPECI can study the inconsistencies that arise after the failures are appeared.

Architecture: Fig. 28 presents the structure of SPECI that has two main layers: (1) Topology layer that defines the data center layout like nodes and network links, and (2) Experiment layer that contains the components of measuring based on SimKit [86]. SimKit provides the event scheduling and random distribution drawing.

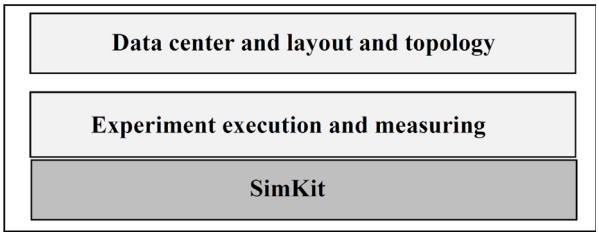


Fig. 28. SPECI structure.

3.2.12. SCORE

Description and main features: Fernández-Cerero et al. [87] presented a simulator SCORE based on lightweight Google Omega simulator [88] to optimize the resources and energy consumption of cloud. SCORE tries to simulate the parallel scheduling, energy efficient monolithic schema, and synthetic workloads. In addition, it considers security features, allocation policies, and heterogeneity of data centers. The empirical experiment proved that SCORE is an efficient and reliable framework for evaluating security, energy, and scheduling algorithm in cloud systems.

Architecture: Fig. 29 shows the structure of SCORE that contains two main components. The first component is Core Simulator (CS) that is considered as the fundamental execution engine and inherits from the Omega lightweight tool. Core Simulator composed of three layers: (1) Workload generation, (2) Core Engine Simulation that investigates the features of workload, and (3) Scheduling models that implement different scheduling strategies. The second component is Efficiency Module (EF) that focuses on the efficiency techniques and energy-efficiency metrics.

3.2.13. GAME-SCORE

Description and main features: Fernández-Cerero et al. [89] introduced GAME-SCORE simulation tool that implements the scheduling model with the Stackelberg game [90]. Stackelberg game consists of two main players as Scheduler and Energy Efficiency Agent. GAME-SCORE extends SCORE toolkit [87] and tries to model the energy efficient IaaS of the clouds. The experiments proved that Stackelberg approach performs better than static energy optimization strategies. Since, it attempts to provide a good balance between low power consumption and high throughput in cloud resource.

Architecture: The structure of GAME-SCORE is like simulator SCORE that is shown in Fig. 30. Nevertheless, GAME-SCORE adds a model for balancing the efficiency of scheduling and energy minimization based on a non-zero symmetric game.

3.2.14. DISSECT-CF

Description and main features: Kecskemeti [91] proposed DISSECT-CF (DIScrete event baSed Energy Consumption simulaTor for Clouds and Federations) tool to provide an energy-aware scheduling for infrastructure clouds. DISSECT-CF supports a resource-sharing model that can model the resource bottlenecks like CPU and network. In addition, the optimization of generic resource sharing performance enhances the entire simulation. DISSECT-CF presents a more complete IaaS stack simulation. It allows to users to derive energy consumption from several resource usage counters.

Architecture: Fig. 31 represent the structure of DISSECT-CF simulator that consists of five layers: (1) Infrastructure Management, (2) Infrastructure Simulation, (3) Energy Modeling, (4) Unified resource sharing, and (5) Event system.

4. Comparison of different cloud simulators

Choosing appropriate tools for specific work is one of the main challenges since there are different tools, which are intended for a particular aim. We provide the summary of cloud simulators discussed in this article. Tables 7a-7e present a representative overview of the major efforts of CloudSim based simulators. Tables 8a-8d provide an attempt to compare the other explained simulators by

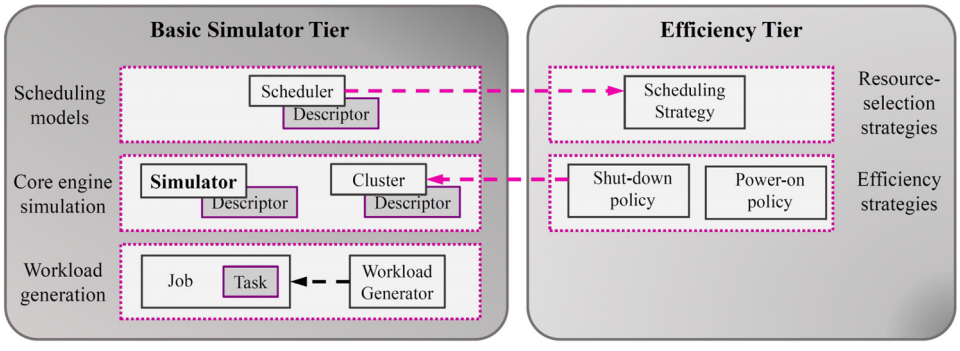


Fig. 29. SCORE structure.

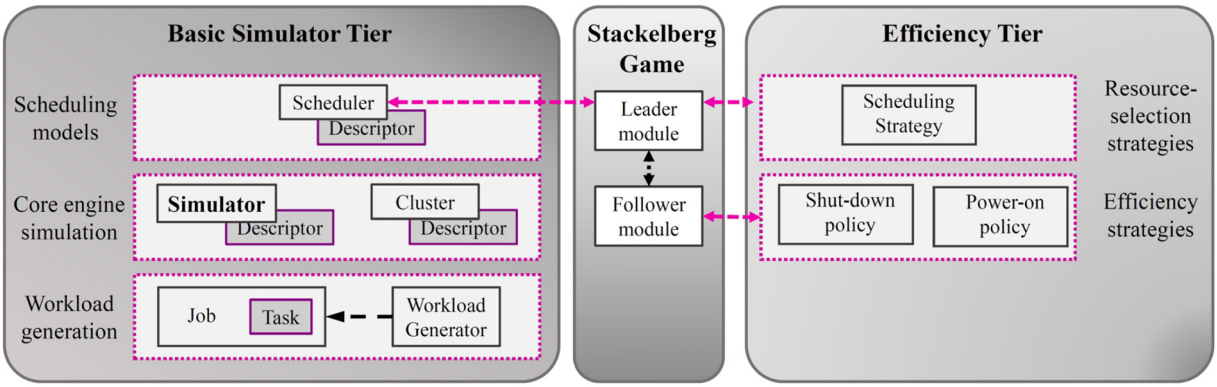


Fig. 30. GAME-SCORE structure.

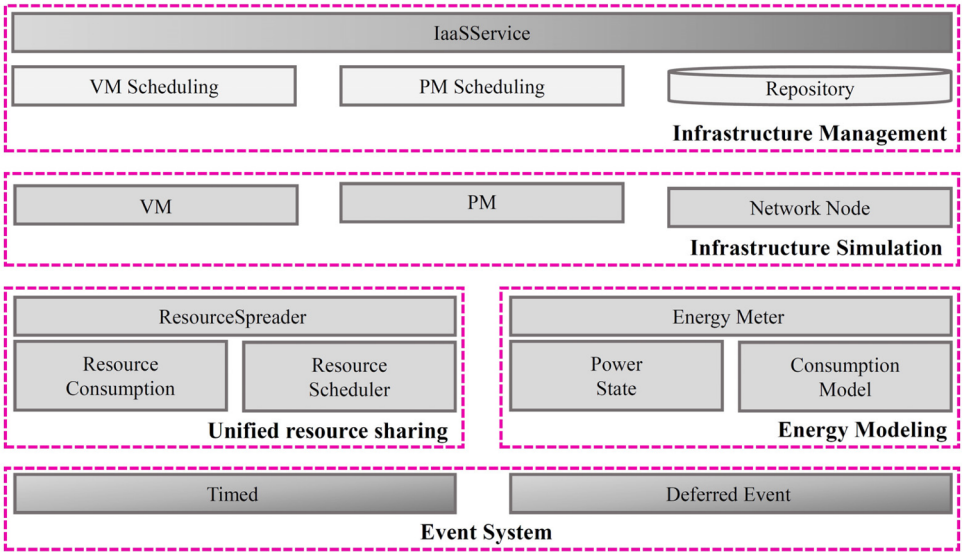


Fig. 31. DISSECT-CF structure.

Table 7a
Comparison of CloudSim extensions.

		CloudSim & Extensions CloudSim	NetworkCloudSim	CloudAnalyst	EMUSim
Parameters	Platform	SimJava	Cloudsim	Cloudsim	Cloudsim, AEF
	Language	Java	Java	Java	Java
	Availability	Open Source	Open Source	Open Source	Open Source
	GUI	✗	✗	✓	✗
	Application Model	✓	✓	✓	✓
	Communication Model	Limited	Full	Limited	Limited
	Energy Model	✓	✓	✓	✓
	SLA Support	✗	✗	✗	✗
	Cost Model	✓	✓	✓	✓
	Network Topology Model	✓	✓	✓	✓
	Congestion Control	✗	✗	✗	✗
	Traffic Patterns	✗	✗	✗	✗
	Federation Model	✓	✓	✓	✗
	Parallel Experiments	✗	✗	✗	✗
	Publication Year	2009	2011	2010	2012
	Documentation Available	✓	✓	✓	✓
	Platform Portability	✓	✓	✓	✗
	Distributed Architecture	✗	✗	✗	✗
	Software/Hardware	Software	Software	Software	Both

Table 7b
Comparison of CloudSim extensions.

		CloudSim & Extensions CDOSim	TeachCloud	DartCSim	DartCSim +
Parameters	Platform	Cloudsim	Cloudsim	ClodSim	Cloudsim
	Language	Java	Java	Java, C + +	Java
	Availability	Open Source	Open Source	Open Source	Open Source
	GUI	✓	✓	✓	✗
	Application Model	✓	✓	✓	✓
	Communication Model	Limited	Limited	Full	Full
	Energy Model	✓	✓	✓	✓
	SLA Support	✓	✓	✗	✗
	Cost Model	✓	✓	✓	✓
	Network Topology Model	✓	✓	✓	✓
	Congestion Control	✗	✗	✗	✗
	Traffic Patterns	✗	✗	✗	✗
	Federation Model	✓	✓	✓	✓
	Parallel Experiments	✗	✗	✗	✗
	Publication Year	2012	2012	2012	2013
	Documentation Available	✓	✗	✗	✓
	Platform Portability	✓	✓	✓	✓
	Distributed Architecture	✗	✗	✗	✗
	Software/Hardware	Software	Software	Software	Software

Table 7c
Comparison of CloudSim extensions.

		CloudSim & Extensions ElasticSim	FederatedCloudSim	FTCloudSim	WorkflowSim
Parameters	Platform	Cloudsim	Cloudsim	Cloudsim	Cloudsim
	Language	Java	Java	Java	Java
	Availability	Open Source	Open Source	Open Source	Open Source
	GUI	✓	✗	✗	✗
	Application Model	✓	✓	✓	✓
	Communication Model	Limited	Limited	Limited	Limited
	Energy Model	✓	✓	✓	✓
	SLA Support	✗	✓	✗	✗
	Cost Model	✓	✓	✓	✓
	Network Topology Model	✓	✗	✓	✗
	Congestion Control	✗	✗	✗	✗
	Traffic Patterns	✗	✗	✗	✗
	Federation Model	✓	✓	✓	✓
	Parallel Experiments	✗	✗	✗	✗
	Publication Year	2016	2014	2013	2012
	Documentation Available	✓	✓	✓	✓
	Platform Portability	✓	✓	✓	✓
	Distributed Architecture	✗	✗	✓	✗
	Software/Hardware	Software	Software	Software	Software

various features.

From Tables 7-8, we can see that:

- Around 54% simulators are designed based on CloudSim tool and is an indication that why Java is the popular programming language.
- Around 75% of simulators are freely available that is interesting for researchers.
- Around 30% of simulators present full to limited GUI that is key factor for easy configuration.
- All of simulators present an application model that can model the interactions and dependencies among virtual machines working together to present a single service like in the case of a multi-tiered web application.
- Around 27% of the simulators provide full communication modeling in data center that is necessary for calculating the required time for transferring messages from source equipment to destination equipment. For example, GreenCloud implements a full TCP/IP protocol.
- Around 78% of simulators allow simple energy modeling that is important in green computing.
- Around 30% of simulators consider SLA violations and so the requested resources are assigned to users based on SLA parameters.
- Around 78% of simulators support the simple cost model that is critical for service provider.
- Around 75% of simulators present limit or full network topology models.

Table 7d.
Comparison of CloudSim extensions.

		CloudSim & Extensions			
		CloudReports	CEPSim	DynamicCloudSim	CloudExp
Parameters	Platform	Cloudsim	CloudSim	Cloudsim	ClodSim
	Language	Java	Java	Java	Java
	Availability	Open Source	Still Not Available	Open Source	Still Not Available
	GUI	✓	✗	✗	✓
	Application Model	✓	✓	✓	✓
	Communication Model	Limited	Limited	Limited	Full
	Energy Model	✓	✓	✓	✓
	SLA Support	✗	✗	✗	✓
	Cost Model	✓	✓	✓	✓
	Network Topology Model	✓	✓	✓	✓
	Congestion Control	✗	✗	✗	✗
	Traffic Patterns	✗	✗	✗	✓
	Federation Model	✓	✓	✓	✓
	Parallel Experiments	✗	✗	✗	✗
	Publication Year	2011	2015	2014	2014
	Documentation Available	✓	✓	✓	✗
	Platform Portability	✓	✓	✓	✓
	Distributed Architecture	✗	✗	✓	✗
	Software/Hardware	Software	Software	Software	Software

Table 7e
Comparison of CloudSim extensions.

		CloudSim & Extensions		
		CM Cloud	MR-CloudSim	UCloud
Parameters	Platform	CloudSim	Cloudsim	Cloudsim
	Language	Java	Java	Java
	Availability	Open Source	Still Not available	Still Not Available
	GUI	✗	✗	✗
	Application Model	✓	✓	✓
	Communication Model	✗	Limited	Limited
	Energy Model	✓	✓	✓
	SLA Support	✗	✗	✗
	Cost Model	✓	✓	✓
	Network Topology Model	✗	✓	✓
	Congestion Control	✗	✗	✗
	Traffic Patterns	✗	✗	✗
	Federation Model	✓	✓	✓
	Parallel Experiments	✗	✗	✗
	Publication Year	2016	2012	2012
	Documentation Available	✓	✓	✓
	Platform Portability	✓	✓	✓
	Distributed Architecture	✗	✗	✓
	Software/Hardware	Software	Software	Software

- Around 6% of simulators allow to implement the congestion based algorithms and so developers can setup close to real network environment of cloud.
- Around 6% of simulators consider traffic pattern that is the flow of data inside a datacenter.
- Around 60% of simulators support the federation model that means whether a toolkit allows users to simulate the federated cloud applications.
- Around 12% of simulators support the parallel execution and so combine more than one machine to do together in order to process tasks.
- Around 90% of simulators have portability ability and can be used under multiple operation systems (e.g., MS Windows, Linux) without significant effort and performance difference.
- Around 15% of simulators provide the ability that a single simulation run being distributed among multiple system at the same time.
- Around 6% of simulators are designed based on software and hardware. EMUSIM is the only toolkit that allows researchers to study the performance of cloud applications on real hardware.

Based on the above discussion, there are several simulators to model cloud environments and simulate several workloads executing on them. Nevertheless, they consider only static application models and so the lack of flexibility is a main limitation of these

Table 8a
Comparative study of cloud simulators.

		Cloud Simulators MDCSim	GDCSim	CloudNetSim	CloudNetSim++
Parameters	Platform	CSim	Bluetoll	OMNeT++	OMNET++
	Language	Java,C++	C/C++, Shell	C++	C++
	Availability	Commercial	Open Source	Still Not Available	Open Source
	GUI	✗	✗	✗	✓
	Application Model	✓	✓	✓	✓
	Communication Model	Limited	✗	Limited	Limited
	Energy Model	✓	✓	✗	✓
	SLA Support	✗	✓	✗	✓
	Cost Model	✗	✗	✗	✓
	Network Topology Model	✓	✗	✓	✓
	Congestion Control	✗	✓	✗	✗
	Traffic Patterns	✗	✗	✗	✓
	Federation Model	✗	✗	✗	✓
	Parallel Experiments	✗	✗	✗	✗
	Publication Year	2009	2011	2013	2014
	Documentation Available	✗	✗	✗	✗
	Platform Portability	✓	✗	✓	✓
	Distributed Architecture	✗	✗	✗	✗
	Software/Hardware	Software	Software	Software	Software

Table 8b
Comparative study of cloud simulators.

		Cloud Simulators GreenCloud	iCanCloud	SecCloudSim
Parameters	Platform	NS-2	OMNET, MPI (SIMCAN)	iCanCloud
	Language	C++, OTcl	C++	C++
	Availability	Open Source	Open Source	Still Not Available
	GUI	Limited	✓	✓
	Application Model	✓	✓	✓
	Communication Model	Full	Full	Full
	Energy Model	✓	✗	✗
	SLA Support	✓	✗	✗
	Cost Model	✗	✓	✓
	Network Topology Model	✓	✓	✓
	Congestion Control	✗	✗	✗
	Traffic Patterns	✗	✗	✗
	Federation Model	✗	✗	✗
	Parallel Experiments	✗	✓	✓
	Publication Year	2010	2011	2014
	Documentation Available	✓	✗	✗
	Platform Portability	✗	✓	✓
	Distributed Architecture	✗	✗	✗
	Software/Hardware	Both	Software	Software

models.

The dynamic adaptation of the simulated applications is one of the main research aspects that must be considered. Since dynamic changes are very important for a long-running application. Therefore, the unforeseen events and failure during application execution can be managed. In addition, it can consider new strategies to control the rapid changes (e.g., customer behavior) that are caused by the dynamic nature of the marketplace. These changes can be at nonfunctional level (e.g., time constraint changes) or the functional level (e.g., task addition or deletion). Therefore, the requirement of dynamic change will cause new challenges at the simulation level. Consequently, more work is required towards evaluating the scheduling algorithms, which can manage the dynamic simulated applications.

Some simulators are designed based on older tools and try to solve the predecessors' limitations. On the other hand, some other simulators are developed from scratch with specific features. Each simulator has its own disadvantages and advantages and hence users must select the correct simulator according to their requirements and preferences. Consequently, we list the positive and negative aspects of the studied simulators in Tables 9a-9e and 10a-10d, respectively. Finally, we can say that selecting a simulator depends up on the types of research problems. Nevertheless, CloudSim is a good choice for a general purpose due to on its features and popularity in the research community. Tables 11a-11b present an overview of 33 cloud simulators through an analysis topics and problems that each simulator was applied to evaluate or solve. Therefore, Table 11 is useful for researchers to select the appropriate tool for the implementation of their projects.

Table 8c
Comparative study of cloud simulators.

		Cloud Simulators GroudSim	CloudSched	SimIC
Parameters	Platform	–	–	SimJava
	Language	Java	Java	Java
	Availability	Open Source	Open Source	Still Not Available
	GUI	✗	✓	✗
	Application Model	✓	✓	✓
	Communication Model	✗	Full	Limited
	Energy Model	✗	✓	Rough Estimation
	SLA Support	✗	✗	✓
	Cost Model	✓	✓	✓
	Network Topology Model	✗	✓	✗
	Congestion Control	✗	✓	✗
	Traffic Patterns	✗	✗	✗
	Federation Model	✗	✗	✓
	Parallel Experiments	✗	✗	✗
	Publication Year	2010	2013	2013
	Documentation Available	✓	✗	✗
	Platform Portability	✓	✓	✓
	Distributed Architecture	✗	✗	✗
	Software/Hardware	Software	Software	Software

Table 8d
Comparative study of cloud simulators.

		Cloud Simulators SPECI	SCORE	GAME-SCORE	DISSECT-CF
Parameters	Platform	SimKit	Google Omega	Google Omega	–
	Language	Java	Scala	Scala	Java
	Availability	Open Source	Open Source	Open Source	Open Source
	GUI	✗	✗	✗	✗
	Application Model	✓	✓	✓	✓
	Communication Model	Limited	✗	✗	Full
	Energy Model	✗	✓	✓	✓
	SLA Support	✗	✓	✓	✗
	Cost Model	✓	✗	✗	✗
	Network Topology Model	✓	✗	✗	✓
	Congestion Control	✗	✗	✗	✗
	Traffic Patterns	✗	✗	✗	✗
	Federation Model	✗	✗	✗	✗
	Parallel Experiments	✗	✓	✓	✗
	Publication Year	2009	2018	2018	2014
	Documentation Available	✗	✓	✓	✓
	Platform Portability	✓	✓	✓	✓
	Distributed Architecture	✗	✓	✓	✗
	Software/Hardware	Software	Software	Software	Software

Table 12 groups the requirements based on cloud services related requirements. We can see that most of simulators support only IaaS and model the public cloud providers. CloudExp, UCloud, secCloudSim, SCORE, and GAME-SCORE are the only simulators that consider the security factors during interaction modeling.

Due to the amount of the consumed energy is increasing, one of the main parameters in cloud data center is energy consumption and so cloud simulators try to provide the appropriate energy models. In this section, we focus on modeling energy to predict the energy consumption of different components of data center (e.g., CPU, memory, storage, and network). Table 13 summarizes the level of supporting for CloudSched, CloudSim, DCSim, GDCSim, GreenCloud, iCanCloud, SCORE, and GAME-SCORE in energy consumption models. We can see that CloudSched simulator considers the energy models only for the CPU physical servers. While, iCanCloud models the impact of energy in various aspects (i.e., CPU servers, memory, storage, and power supply unit (PSU)). CloudSim provides several power models for CPU servers (i.e., Linear, Square, Cubic, Square root, and Linear interpolation). Nevertheless, CloudSim does not study the impact of other elements in the energy consumption of the data center. GreenCloud simulator computes the total energy consumption by adding the energy consumed of the network switches, memory, hard disks, and network interface cards consumption. DCSim tool models the energy consumption of servers based on the linear and the linear interpolation models of CloudSim. Nevertheless, DCSim does not assume any other element for the energy consumption of the data center. GDCSim presents two different cooling models (i.e., dynamic and constant) to estimate the necessary power for cooling system. GDCSim also considers the energy consumption of CPU physical servers. SCORE simulator considers the shutdown and

Table 9a
A summary of strengths.

Simulator	Strengths/focus
CloudSim	(1) It models data centers, virtual machines, and resource provisioning. Therefore, developers can scale the problem up by changing the number of cloudlets and hosts for software development. (2) It presents a basic energy consumption model based on CPU utilization. Therefore, developers can develop a model for the Device Voltage and Frequency Scaling (DVFS) approach. (3) It implements space-shared and time-shared allocation policies. Therefore, a developer can flexibility switch between these policies for allocating the processing cores to virtualized services.
NetworkCloudSim	(1) It provides extra communication models like message passing. Therefore, developers can model complex applications. (2) It presents a network flow model for data centers based on bandwidth sharing and latencies. Therefore, developers can design a scalable network. (3) It defines root, aggregate, and edge (access) level switches. Therefore, developers can customize switches and their ports. (4) It provides a flow model and a packet model. Therefore, a designer can select a lower computational overhead model with less detail (i.e., flow model).
CloudAnalyst	(1) It offers a full GUI and geographical factors. Therefore, developers can easily pick up and implement cloud scenarios. (2) It presents a wide range of configurable factors such as data center distribution and user location. Therefore, it has high degree of configurability and flexibility.
EMUSim	(1) It extracts information from applications by emulation. Therefore, developers can apply this information to create an accurate model. (2) A designer can better predict the behavior of service and resource utilization on cloud platforms by automatically extracting information.
CDOSim	(1) It offers Service Level Agreement violations and measures delays and costs from a client perspective. Therefore, a developer can make a decision for selecting a client perspective and the runtime deployment strategy. (2) It presents the real-life user trace due to the cloud clients' lack of knowledge about cloud options. Therefore, developers can compare the effectiveness of a particular cloud solution with those of other solutions based on different cloud deployment parameters. (3) Developers can investigate the impact of the architecture of cloud platform on the application performance by a test benchmark that is provided by CDOSim.
TeachCloud	(1) It presents a friendly GUI and so researchers and industrial engineers can set and run their scenarios. (2) It provides various architectures such as VL2, BCube, Portland, and Dcell. Therefore, developers can monitor the components of data centers and evaluate the impact on system effectiveness through a specific module.
DartCSim	(1) It offers an intuitive interface to represent the data center characteristics and network topology. (2) It provides the ability that the configuration of data could be imported or exported at any level. Therefore, developers can set the data of a single CPU or the data of a whole data center.

Table 9b
A summary of strengths.

Simulator	Strengths/focus
DartCSim +	(1) It offers the power and network models and hence developers can design network and power-aware scheduling methods. (2) It handles the distortion problem by a module for controlling transmission of network links.
ElasticSim	(1) It provides resource runtime auto-scaling and task execution based on stochastic modeling. Therefore, developers can model efficient workflow scheduling algorithms. (2) Developers can investigate the probability distribution of task execution time and the length of cloud pricing intervals on scheduling strategies.
FederatedCloudSim	(1) It provides several federated cloud scenarios and so developers can study the interconnection of different cloud service providers (CSPs). (2) It contains packages for evaluating the SLA-aware scheduling algorithms. Therefore, developers can define and monitor the distributed services based on SLAs. (3) It offers different structures (i.e., central and distributed multi-level) for service placements and scheduling strategies.
FTCloudSim	(1) It focuses on simulating the reliability mechanism and so developers can investigate the reliability assurance of cloud service. (2) It adds new modules that can trigger failure events based on special distribution. (3) It provides a checkpoint-based cloudlet recovery approach to resume tasks from the host failure.
WorkflowSim	(1) It supports a stack of workflow parser and workflow engine delay to workflow optimization techniques with better accuracy are implemented. (2) It implements several workflow-scheduling methods such as HEFT, Min-Min, and Max-Min and so developers can compare their algorithms with them in a simple way. (3) It considers task clustering and layered overhead to the workflow simulation. Therefore, developers can test various strategies based on a real trace of overheads. (4) It models failures for two layers (i.e., task/job) by an interface and hence developers can design fault-tolerant techniques.
CloudReports	(1) It offers modular architecture, broker policies, and resource utilization models. Therefore, developers can extend their applications for evaluating with several scheduling and provisioning strategies. (2) It includes a GUI with many functionalities and so various aspects for the researcher to play the role of service providers and clients are provided.
CEPSim	(1) It uses directed acyclic graphs to introduce complex event processing queries. Therefore, developers can execute large-scale experiments in various environments such as private and public clouds. (2) It models various CEP cloud-based environments independently of query languages and platform characteristics. (3) It extends various scheduling and placement algorithms and so developers can evaluate queries in heterogeneous systems under different load conditions.

Table 9c
A summary of strengths.

Simulator	Strengths/focus
DynamicCloudSim	(1) It models instability concepts that are caused by dynamic parameters such as runtime and failures in executing tasks. Therefore, developers can evaluate their applications such as schedulers in a shared computational infrastructure (i.e., public cloud). (2) It supports the changes in the performance of VM based on exponential distribution. Therefore, developers can adjust the number of performance changes based on their respective environments. (3) It defines a failure generation method to determine the task is bound to succeed or fail. Therefore, developers can determine the rate of failure during experiments that is common in distributed computing.
CloudExp	(1) It adds a MapReduce into CloudSim and so developers can handle big data by parallel data processing techniques. (2) It offers several network models such as VL2, BCube, Portland, and DCell. Therefore, developers can evaluate their applications based on actual topologies in the real cloud system. (3) It supports user-friendly GUI and so developers can reconfigure a cloud environment and monitor the overall system behavior easily.
CM Cloud	(1) It supports different cost models that can be designed using XML. Therefore, developers can execute a cost model of different providers of cloud. (2) It provides an automated searching for information from three important cloud providers such as Amazon, Google, and Azure. Therefore, developers can apply results best suited to the current market price in their implementations.
MR-CloudSim	(1) It simulates the MapReduce computing model and hence developers can validate the MapReduce operations in an easier and cheaper way. (2) It supports data-intensive applications since MapReduce provides a programming model for Big Data processing.
UCloud	(1) Its architecture is based on a hybrid cloud and so developers can evaluate their policies in the public and private clouds. (2) It simulates cloud for university and so developers can consider educational goals.
MDCSim	(1) It considers multi-tier data center structure and so developers can study the performance of application in a scalable platform under different network loads with different tier configurations. (2) It supports power utilization and switches connected along with nodes and hence developers can compare energy-efficient policies for cloud environments. (3) It has low simulation overhead and so developers can evaluate large-scale and three-tiered applications with varying the configuration of each tier.

Table 9d
A summary of strengths.

Simulator	Strengths/focus
GDCSim	(1) It offers a completely green cloud and so a developer can investigate their energy-aware techniques and resource allocation algorithms with varying workload distribution and topology. (2) It considers the thermal side of energy efficiency and so developers can use temperature and airflow patterns in their management algorithm and cooling policies in the cloud platform. (3) It presents the online analysis and so developers can control evaluation scenario during simulation running.
CloudNetSim	(1) It presents end-to-end network communications among clients and providers in a cloud environment. Therefore, developers can study resource management for interactive and real-time cloud applications. (2) It supports additional modularity and extensibility policies for application models and so developers can effectively evaluate virtual machine (VM) deployment algorithms.
CloudNetSim + +	(1) It presents energy consumption modeling and so developers can study an energy -aware algorithm by varying the number of nodes. (2) It offers various evaluation parameters for network performance such as delay and throughput based on different topologies. (3) It supports friendly GUI and so developers can test their strategies like scheduling without worrying about low-level details. (4) Developers can model a realistic network with communication among different nodes through packets.
GreenCloud	(1) It offers enhanced capabilities for energy modeling. Therefore, developers can obtain energy usage for servers, switches, and links. (2) It presents several architectures such as a Two-tier and Three-tier data center. Therefore, developers can use multiple options for topologies and control transmission rates to manage power saving.
iCanCloud	(1) It presents an appropriate GUI and so developers can configure the cloud system and obtain graphical reports. (2) It models the hypervisor module of cloud providers and so developers can define their brokers between cloud users and cloud data centers. (3) It defines storage systems by different capabilities like local or remote storage systems and parallel storage. Therefore, developers can execute multiple independent experiments in parallel to utilize the resources of system.
SecCloudSim	(1) It offers basic security aspects (e.g., authentication and authorization). Therefore, developers can evaluate different policies in terms of packet rate, response time, and time delay.
GroudSim	(1) It is appropriate for scientific workflow applications in the cloud and grid. Therefore, developers can study their policies in both environments with one simulation thread only. (2) It uses the ASKALON environment and so developers can import experiments that show real applications from that environment.

power-on methods with the objective of optimizing the energy consumption of the data centers. GAME-SCORE simulator implements strategy based on a non-zero sum Stackelberg Game to provide a balancing between high performance throughput and low energy consumption.

Table 14 lists the improvements of simulators that have been made year by year. We can see from Table 14 that most recent studies have focused on accurate modeling of energy consumption in cloud data centers. In addition, the simulator's ability to manage complex topologies (like real cloud environment) is another concern. In other words, a good cloud simulator should work with large the number of data centers (e.g., 100,000 hosts).

Table 9e
A summary of strengths.

Simulator	Strengths/focus
CloudSched	(1) It takes into account resource from different dimensions (e.g., CPU, storage, and network bandwidth) for both physical machines and virtual machines. Therefore, a developer can design new resource scheduling algorithms to achieve optimal solutions in the cloud platform. (2) It offers scheduling of VMs at IaaS layer and hence developers can consider both computing servers and user workloads. (3) It has a lightweight design and scalability. Hence, developers can simulate thousands of requests in a few minutes.
SimIC	(1) It provides the heterogeneity of multiple cloud configurations and developers can design an inter-cloud scheduling algorithm based on different distributed parameters. (2) It offers meta-brokers that transfer information in P2P networks and hence developers can evaluate resource discovery implementations in the decentralized topology of meta-brokers. (3) It models VM migration mechanisms based on requirements of cloud provider and so developers can give backup of VMs to storage devices for sensitive cases.
SPECI	(1) It focuses on the performance of data centers as a unit and then applies an efficient middleware schema in a cloud data center. Hence, developers can propose middleware as the layer of software to manage tasks, virtual networks, and resilience. (2) It considers different architectures of state (e.g., central node and hierarchical design) to investigate the behavior of the whole system. Therefore, developers can study communication policies with varying setup parameters and protocols.
SCORE	(1) It presents an energy-efficient model and so developers can evaluate different energy-aware scheduling algorithms for cloud platforms with large data centers. (2) It offers shutting-down and powering-on computational server mechanisms and hence developers can apply them to optimize the energy consumption of the data center. For example, idle machines should be shut down to overcome the negative impact on the overall performance of cloud.
GAME-SCORE	(1) It uses Stackelberg-Game to present a game-based scheduling model. Therefore, developers can use this model to provide a fair balance between energy consumption and makespan. (2) It models large-cluster platforms and so developers can evaluate their resource management methods with popular cloud computing services (i.e., tens of thousands of nodes).
DISSECT-CF	(1) It provides a unified resource-sharing schema and so developers can consider in-data-center networking during experiments. (2) It offers energy modeling and so developers can monitor the energy usage of different elements of resources such as network links and disks. (3) It determines the internal details of the cloud environments and so developers can define new physical machines, cloud topology, VM schedulers, and power states to obtain more accurate results.

Table 10a
A summary of limitations.

Simulator	Limitations
CloudSim	(1) It considers the limited network model (only transmission delay). Therefore, developers cannot test their application in realistic network topologies. Since it does not define switches and assumes that each VM is connected with all the others. (2) It does not consider the intra–data center communication and bandwidth sharing of links. (3) When a load of application is more than the capability of hosts then the time delay is considerably increased. The main reason is related to limitations of the VM placement strategies.
NetworkCloudSim	(1) It does not suitable cost model. Therefore, developers cannot evaluate the price of the used services. (2) It models simple energy modeling and hence the energy efficiency of data centers is not comprehensively investigated.
CloudAnalyst	(1) It does not support network inside the data center and pricing model. Therefore, developers cannot evaluate the profit of providers for choosing the provider that satisfies the client's resource request with the lowest cost. (2) It has no TCP/IP implementation and offers limited support for power modeling.
EMUSim	(1) It has scalability limitations due to emulation and hardware constraints. Therefore, it is not very suitable in generating large and realistic workloads. (2) It does not present the evaluation of performance for different types of applications.
CDOSim	(1) It has simplistic communication modeling and does not support the parallel experiment. Therefore, it may have some issues with large-scale applications. (2) It assigns resources to virtual machines based on the fixed performance measures. Therefore, developers cannot study resource provision algorithms based on compute units. (3) It does not GUI support and so configurations of simulator are not simply and efficiently.
TeachCloud	(1) It is unproven for general purposes due to it is designed only for academic goals. (2) It does not support failures during task execution. While the effect of this parameter of uncertainty is not negligible. Therefore, developers cannot study influence failure probability on the runtime of a given application on public or commercial cloud.
DartCSim	(1) It does not provide complete power modeling and so developers cannot implement efficient network-based selection strategies. (2) It does not model dynamic changes of performance during execution. Therefore, developers cannot consider the instability of cloud due to external loads as a consequence of sharing resources with users during their implementation of applications.

5. Simulators performance evaluation

This section consists of two main parts: (1) we analyze which simulators have received the most attention in the research papers, and (2) we discuss the performance of simulators with focusing on the scalability factor that indicates how fast a toolkit can execute as the number of requests is increasing. This parameter is investigated in term of scheduling problem. Scheduling is one of the most important challenges in the distributed systems, especially cloud environments due to the dynamic and heterogeneous nature of resources.

Table 10b
A summary of limitations.

Simulator	Limitations
DartCSim +	(1) It does not cover the cost model and so fails to model provider profit for various applications. (2) It does not focus on the security features, and hence developers cannot analyze the security aspects of the cloud.
ElasticSim	(1) It contains basic energy consumption and so developers can study the limited power consumption metrics. (2) It cannot simulate security-related experiments for the cloud platform.
FederatedCloudSim	(1) Developers cannot define multi-level (more than three levels) cloud federation schedulers. (2) It does not cover the optimization of the total resource usage for each cloud provider and the power consumption of each data center.
FTCloudSim	(1) It considers only the basic energy concept and so developers cannot integrate the energy efficiency mechanism with the reliability enhancement mechanisms. (2) It does not support workflow parsing and so developers cannot implement workflow scheduler easily.
WorkflowSim	(1) It includes limited types of failures. Therefore, developers cannot simulate the situation when a task is not successfully sent due to network problems or workflow scheduler issues. (2) It does not consider the performance characteristic of file I/O. Therefore, developers cannot obtain suitable simulation for data-intensive applications since these applications involve reading or writing huge data files. (3) It supports only simple workflow techniques and so developers cannot use other important approaches like workflow partitioning in their implementations.
CloudReports	(1) It suffers from a lack of statistical analysis in reports. (2) It does not define the security layer and so developers cannot consider the basic security features in a cloud-simulated environment.
CEPSim	(1) During the simulation, queries cannot be arrived and left. Therefore, developers cannot run their applications in a dynamic environment. (2) It neglects the impact of the network in energy modeling of data center and so for applications where energy consumption is the key concern, the results of this simulator will not be accurate.
DynamicCloudSim	(1) Each virtual machine only processes one task at a time. (2) It provides a limited power modeling and so developers cannot accurately calculate the energy consumption of their experiments.
CloudExp	(1) It does not model dynamic changes in VM performance and failures during task execution. Therefore, developers may fall short in their modeling of the instability in the cloud environment. (2) It assumes a simplistic energy model and so developers cannot investigate energy and traffic-aware strategies very well.

Table 10c
A summary of limitations.

Simulator	Limitations
CM Cloud	(1) It does not consider failed task execution. Therefore, developers cannot evaluate their executing applications in a dynamic environment where failure is likely due to failure data retrieving and failure during the startup of machine. (2) It does not support congestion and traffic patterns. Therefore, developers cannot investigate the overall impact of traffic sent and received by several hosts in a shared link of network.
MR-CloudSim	(1) It does not consider the time and cost related to file processing and so developers cannot determine the cost of service usage. (2) It does not model the mitigation approach. Therefore, developers cannot determine that when and where VMs are mitigated while it has an impact on an application's overall performance.
UCloud	(1) It does not support security policies and so developers concern with security concepts for users and resources in the cloud platform. (2) It does not focus on the cost model and hence developers cannot determine the costs involves in data center communications.
MDCSim	(1) It does not support federation policy and so developers cannot evaluate their applications in the heterogeneous cloud with various domains. (2) It does not present complete network model. Therefore, developers cannot explore the characteristics of data centers in detail and model realistic network requests in terms of topology.
GDCSim	(1) It does not consider the parallel experiments and so developers suffer from the evaluation of large-scale applications. Since it cannot execute the experiments through several machines. (2) It does not consider the security aspects of the cloud platform. Therefore, developers cannot design security schema in terms of authentication and authorization.
CloudNetSim	(1) It presents basic energy modeling and so developers cannot study power-saving models and investigate the heat produced by data center in an accurate way.
CloudNetSim + +	(1) It does not support the migration concept to determine when, which and where a virtual machine should be mitigated. (2) It suffers from security layer and so the lack of this critical factor results in a negative impact on sensitive tasks. In addition, this limitation leads to ethical and financial losses.
GreenCloud	(1) It has a scalability problem since it requires very large simulation time and high memory spaces. (2) It does not provide a full traffic aggregation model and so developers cannot study the congestions control strategies for cloud data centers.

5.1. Simulators based study analysis

We try to determine the popularity of simulators in lots of research papers available in various fields. We have selected papers from various reputed publications such as IEEE, Elsevier, and Springer from 2009 to 2020 as shown in Fig. 32. In the initial search, we find that there are too many articles on the CloudSim, so this simulator is dropped as the most popular tool. Therefore, the results of this section do not include the CloudSim tool. From Fig. 32, we can see that most of the papers are published after 2012 and maximum article belong to Springer publication.

Table 10d
A summary of limitations.

Simulator	Limitations
iCanCloud	(1) It does not present a complete energy utilization model. Therefore, developers cannot efficiently investigate power management techniques. (2) It does not focus on security factors, and so developers cannot investigate a broad set of mechanisms for the protection of data and applications.
SecCloudSim	(1) It suffers from the lack of advanced security mechanisms (e.g., privacy, integrity and encryption). Therefore, developers cannot study existing cloud services in various attacking scenarios and investigate vulnerabilities. (2) It does not consider workflow parsing and so developers cannot implement workflow schedulers for cloud platforms.
GroudSim	(1) It presents only basic concepts on the network side. Therefore, developers cannot configure a realistic network based on topology, request size, and, hierarchy. (2) It suffers from performance degradation in large-scale applications, and so developers cannot evaluate the scale of the experiment with hundreds of nodes.
CloudSched	(1) It does not consider failures during task execution and so developers cannot integrate fault-tolerant design in scheduling policy. (2) It does not focus on federation policy and so developers cannot investigate multiple clouds connected together.
SimIC	(1) It has less focus on energy consumption of the system and so a developer cannot explore the energy challenges that are faced by cloud entities. (2) It does not offer full network model and so developer cannot investigate the traffic and congestions control approaches in cloud platform.
SPECI	(1) It has scalability problem. When developers increase the data center size then some characteristics of middleware do not scale in a linear way. (2) It does not model the changes of virtual machine performance during execution. While this parameter has a great impact on the runtime of application in the commercial cloud (e.g., Amazon's Elastic Compute Cloud (EC2)).
SCORE	(1) It has no specific modules of security and so developers cannot investigate the security challenges of cloud entities. (2) Data locality issues, the communication costs, and dynamic query analysis have not been addressed. Therefore, selecting a suitable virtual machine to execute a task can be delegated.
GAME-SCORE	(1) It does not support VM migration and consolidation. (2) It does not model fault-tolerance in some ways and so developers cannot simulate cloud service reliability enhancement methods.
DISSECT-CF	(1) It does not offer specific modules of security aspects for cloud platforms. (2) It provides a typical network model and so developers cannot define real network devices (e.g., routers, switches) and simulate various network architectures.

Table 11a
Examples of applications for cloud simulators.

Cloud Simulator	Topic/problem
CloudSim	Cloud workflow execution, Web session modeling, Data replication [92], Fault tolerance [56]
NetworkCloudSim	Modeling network of cloud [93], Resource allocation strategy [94-95]
CloudAnalyst	Presenting service brokering algorithm [96-97], Load balancing techniques [98-99]
EMUSim	Designing a particular networks [100], Emulation of cloud network [101], Evaluating the performance of SSLB [102]
CDOSim	Deployment cloud options [103], Modeling the provider migration [104]
TeachCloud	Presenting resource allocation strategy [105,65], MapReduce modeling simulation [106]
DartCSim	Configurations of tasks and network topology [52]
DartCSim +	Power-aware scheduling algorithms, Management of transmission in network links [53]
ElasticSim	Evaluating the effectiveness of EDA-NMS strategy [107] and DDS strategy [108]
FederatedCloudSim	Evaluation of scheduling algorithms at federation level [109], Presenting the SLA approaches for federated clouds [55]
FTCloudSim	Fault tolerance [110], Implementing task rescheduling method (TRM) [111], Evaluating the effectiveness of fault tolerance virtual machine placement strategy [112]
WorkflowSim	Simulation of workflow management system [113], Verifying the performance of SSLB [102], Implementation of workflow scheduling [114], Evaluation of PGA strategy [115] and DBCSO algorithm [116]
CloudReports	Implementation of energy-aware task scheduling algorithm [117], Modeling the VM allocation algorithm [118], Running several simulations at the same time [119]
cepsim	Modeling the complex event processing systems [120]
DynamicCloudSim	Fault tolerance [56], Simulating the service fail during execution of service [113]
cloudexp	Modeling the mobile cloud computing framework [121], evaluation of the drpm system [122], providing mapreduce system [123], simulating the cloud-based wbans model [124]
CM Cloud	Evaluation of SSLB [102], Simulation of the real cases for research purposes [67]
MR-CloudSim	Processing the large amount of data based on MapReduce paradigm [68]

In Fig. 33, we can observe the distribution of simulators among articles. The second popular choice (after CloudSim) in the research community is WorkflowSim that can evaluate different workflow optimization techniques in cloud computing environments.

From Fig. 33, we can interpret that approximately 85% of articles used CloudAnalyst, WorkflowSim, GroudSim, ElasticSim, iCanCloud, CloudSched, NetworkCloudSim, and GreenCloud as evaluation toolkits. Therefore, we select only these simulators as commonly used tools to evaluate their scalability in solving a scheduling problem.

Table 11b
Examples of applications for cloud simulators.

Cloud simulator	Topic/problem
UCloud	Implementation of DisSetSim [125]
MDCSim	Cloud resource allocation [126], Detecting the malicious activity based on predictive modeling [127]
GDCSim	Providing cooling method [72], Evaluating the hybrid simulator [128]
CloudNetSim	Provide resource management and scheduling polices [74]
CloudNetSim + +	Evaluating the performance of SSLB [102], Providing the DCN architect [129], Modeling the migration policies [130]
GreenCloud	Providing the communication models [131-132], Data replication methods [133], Implementation of Network-as-a-service (NaaS) [134], Scheduling for opportunistic grids [135], Presenting a high availability for cloud [136]
iCanCloud	Presenting a service brokering method [137], Storage modeling [138], Providing the live migration [139], Energy modeling [140]
SecCloudSim	Analyzing the security features [141]
GroudSim	Scheduling of scientific workflows [142], Simulate workflows with a high degree of fault tolerance [143]
CloudSched	Implementing dynamic load-balancing scheduling strategy [144]
SimIC	Parallel performance tendency [145], Analyzing inter-cloud meta-scheduling (ICMS) framework [145]
SPECI	Monitoring of subscription networks [146], Modeling the hierarchical cloud network [147]
SCORE	Evaluation of energy-aware scheduling algorithms [148], Parallel-scheduling modeling, Execution of heterogeneous, realistic and synthetic workloads [87]
GAME-SCORE	Analyzing the efficiency of the game-based scheduling model, Efficient energy methods [89]
DISSECT-CF	Providing the events and infrastructure components [142], Modeling the generic IoT sensors [149], Simulating the TCG workflow [150], Modeling low power clusters for cloud simulation [151], Presenting a model for data center energy consumption [152]

Table 12
Evaluation cloud services requirements for cloud simulators.

Simulator	Support IaaS	Support PaaS	Support SaaS	Modeling of Public Cloud Providers	MigrationPolicy	Security
CloudSim	✓	✗	✗	✗	✓	✗
NetworkCloudSim	✓	✗	✗	✗	✓	✗
CloudAnalyst	✓	✗	✗	✗	✗	✗
EMUSim	✓	✗	✗	Amazon EC2	✗	✗
CDOSim	✓	✗	✗	✓	✗	✗
TeachCloud	✓	✗	✗	✓	✗	✗
DartCSim	✓	✗	✗	✗	✓	✗
DartCSim +	✓	✗	✗	✗	✗	✗
ElasticSim	✓	✗	✗	✗	✗	✗
FederatedCloudSim	✓	✗	✗	✗	✗	✗
FTCloudSim	✓	✗	✗	✗	✗	✗
WorkflowSim	✓	✗	✗	✗	✗	✗
CloudReports	✓	✗	✓	✗	✗	✗
CEPSim	✓	✗	✗	✗	✗	✗
DynamicCloudSim	✓	✗	✗	✗	✗	✗
CloudExp	✓	✓	✓	✗	✗	✓
CM Cloud	✓	✗	✗	✗	✓	✗
MR-CloudSim	✓	✗	✗	✗	✗	✗
UCloud	✓	✓	✓	Amazon EC2	✓	✓
MDCSim	✗	✗	✗	✗	✗	✗
GDCSim	✓	✗	✗	✗	✗	✗
CloudNetSim	✓	✓	✓	✗	✗	✗
CloudNetSim + +	✓	✗	✗	✗	✗	✗
GreenCloud	✓	✗	✗	✗	✗	✗
iCanCloud	✓	✗	HPC	Amazon EC2	✗	✗
SecCloudSim	✓	✗	HPC	✗	✗	✓
GroudSim	✓	✗	✗	✗	✗	✗
CloudSched	✓	✗	✗	Amazon EC2	✗	✗
SimIC	✗	✗	✗	✗	✗	✗
SPECI	✓	✗	✗	✗	✗	✗
SCORE	✓	✗	✗	✗	✗	✓
GAME-SCORE	✓	✗	✗	Google data-center traces	✓	✓
DISSECT-CF	✓	✗	✗	✗	✓	✗

5.2. Evaluation of commonly used simulators

Eight tools are appraised in a practical way based on various scenarios in order to evaluate their performance regarding resource consumption and execution time for solving scheduling problems.

The scheduling problems in a cloud environment can be classified into three groups: resource scheduling, workflow scheduling, and task scheduling. In this paper, only task scheduling and workflow scheduling problems are investigated. Task scheduling tries to map a set of tasks to a set of virtual machines (VMs) in order to fulfill users' demands. While, workflow scheduling is a general form of

Table 13
Energy consumption models in the simulators.

Simulator	CPU	Network	Memory	Storage	PSU	Cooling model	Shutdown/ power-on	Game theory
CloudSched	✓	✗	✗	✗	✗	✗	✗	✗
CloudSim	✓	✗	✗	✗	✗	✗	✗	✗
DCSim	✓	✗	✗	✗	✗	✗	✗	✗
GDCSim	✓	✗	✗	✗	✗	✓	✗	✗
GreenCloud	✓	✓	✓	✓	✗	✗	✗	✗
iCanCloud	✓	✓	✓	✓	✓	✗	✗	✗
SCORE	✗	✗	✗	✗	✗	✗	✓	✗
GAME-SCORE	✗	✗	✗	✗	✗	✗	✓	✓

Table 14
Improvements of simulators year by year.

Year	Simulators	Improvements
2009	CloudSim, MDCSim, SPECI	(1) Present cloud application models (2) Improve the platform portability
2010	CloudAnalyst, GreenCloud, GroudSim	(1) Focus on energy consumption modeling (2) Present geographical factors by GUI (3) Support high number of requests
2011	NetworkCloudSim, CloudReports, GDCSim, iCanCloud	(1) Present extra communication models (2) Consider low-level power saving mechanisms and cooling (e.g., thermal side) (3) Support a full GUI and configurable factors (4) Introduce storage modeled in more detail (5) Discuss trade-off between costs and performance
2012	EMUSim, CDOSim, TeachCloud, DartCSim, WorkflowSim, MR-CloudSim, UCloud	(1) Point out SLA and deployment options (2) Enhance workflow scheduling method (3) Present MapReduce computing model (4) Focus on educational purposes
2013	DartCSim+, FTCloudSim, CloudNetSim, CloudSched, SimIC	(1) Improve VM migration mechanism (2) Focus on a reliability model (3) Extend scheduling of VMs at IaaS layer
2014	FederatedCloudSim, DynamicCloudSim, CloudExp, CloudNetSim++, SecCloudSim, DISSECT-CF	(1) Offer the federated cloud scenarios (2) Support dynamic changes of performance at runtime (3) Focus on a realistic network model (4) Support large scale problems
2015	CEPSim	(1) Support complex event processing request
2016	ElasticSim, CM Cloud	(1) Provide resource runtime auto-scaling
2017	–	–
2018	SCORE, GAME_SCORE	(1) Present energy-efficient modeling with shutting-down and powering-on computational servers mechanisms (2) Introduce game-based scheduling strategy

task scheduling and aims to find the most suitable resources for executing workflow tasks in an appropriate order.

In Section 5.2.1, we compare task scheduling based simulators (i.e., CloudAnalyst, iCanCloud, CloudSched, NetworkCloudSim, and GreenCloud). In Section 5.2.2, we evaluate WorkflowSim, GroudSim, and ElasticSim for solving workflow scheduling problems since these three tools support workflow optimization techniques.

5.2.1. Task scheduling problem

In cloud environment, task scheduling procedure consists of three phases as follows.

- Discovery of Resource: Firstly, the potential resources and their working status are specified by the cloud service provider. Then, some resources that do not have minimum requirements of tasks are filtered.
- Selection of Resources: Secondly, the scheduler should select a single resource from the obtained set from previous phase. Therefore, it gathers detailed information and makes a decision based on some QoS parameters.
- Submission of Task: Finally, task is submitted to the selected resource for execution.

The problem is to schedule the cloud tasks on the pool of resources in a given time-frame for utilizing the cloud resources optimally. In other words, task scheduling is about mapping n independent/dependent tasks onto m resources based on various QoS requirements. The overall task scheduling process is presented in Fig. 34. The aim of using scheduling technique in a cloud environment is to enhance load balancing, and resource utilization, save energy, reduce costs, and minimize the total processing time. Therefore, cloud scheduler should consider the capability of resources and user demands to find efficient mapping among tasks and

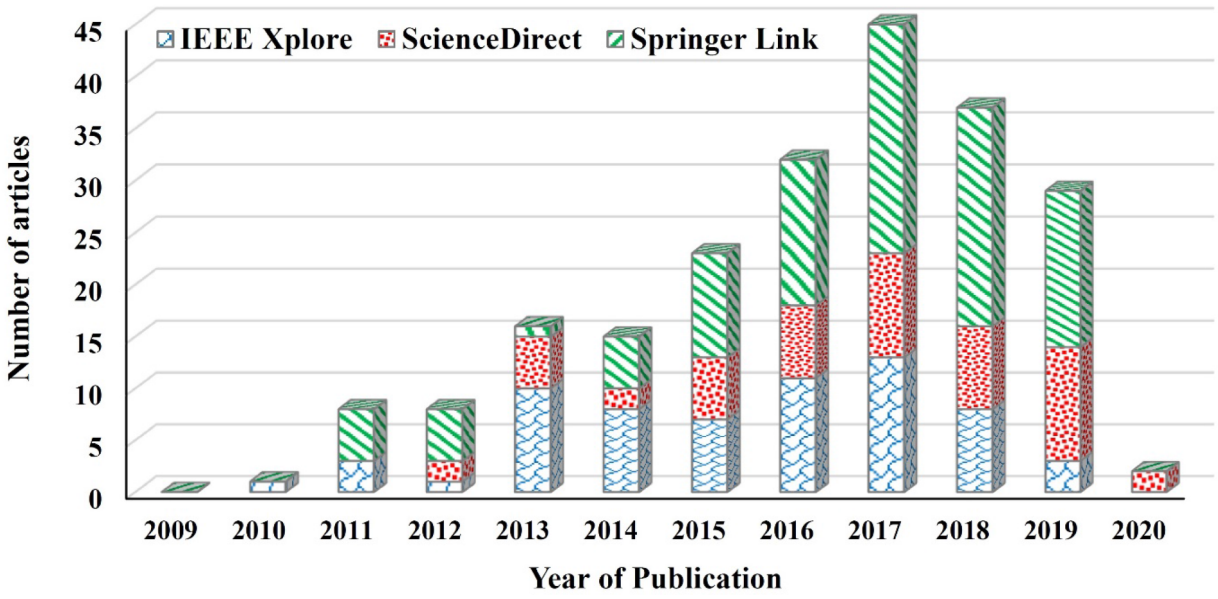


Fig. 32. Articles distribution over time including Elsevier, IEEE, and Springer (without CloudSim).

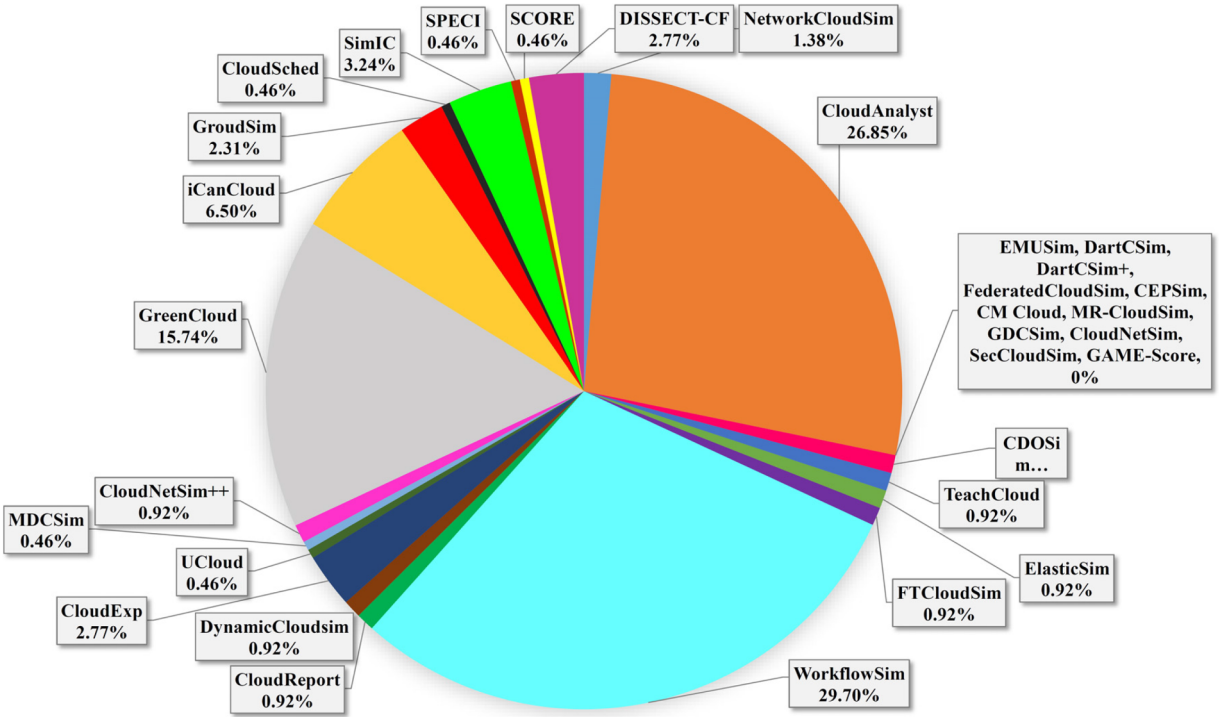


Fig. 33. Comparisons between simulator usages in reviewed articles.

resources.

In the following sections, Round Robin (RR) method is used for solving scheduling problems. RR is one of the most common techniques, which assigns an equally small unit of time to each task. The main concern of RR is dividing load to all resources equally and so uses a cyclic schema. In other words, it assigns all tasks to the controller at least once and then scheduler returns to the first task.

Experimental setup for task scheduling based simulators: In task scheduling process, firstly tasks are generated and then they are prepared to be allocated (i.e., the process of mapping tasks to resources). In different simulators, task generation approaches may vary and the preparation process before task allocation would have minor differences. We list the task definition in different

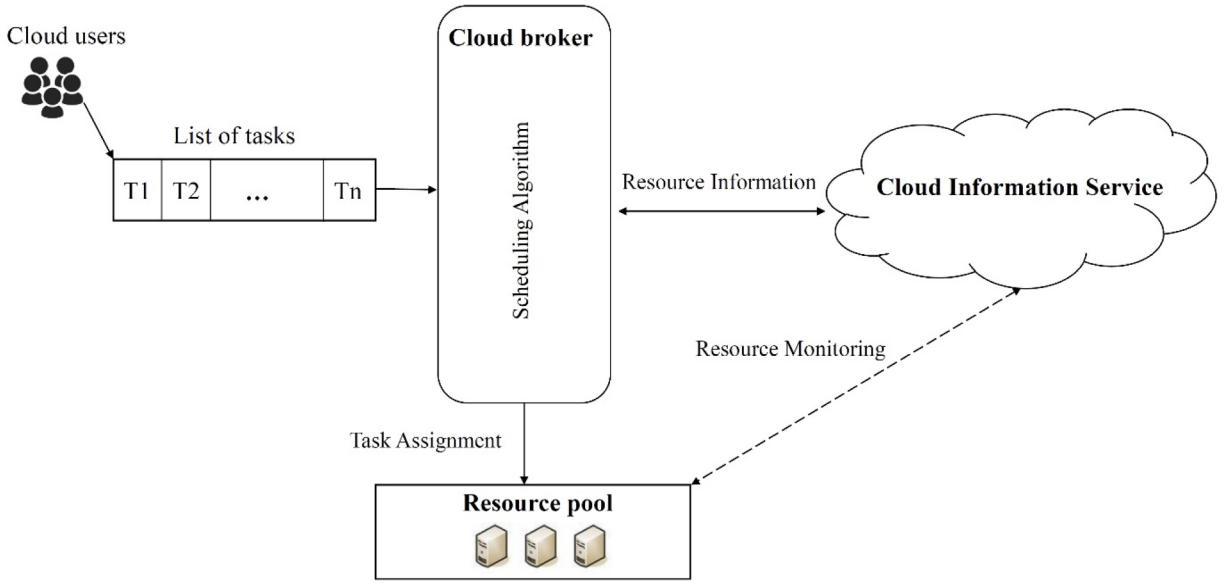


Fig. 34. Task scheduling process.

simulators as follows:

- In NetworkCloudSim, CloudSched, and CloudAnalyst, tasks are generated as VM instances and put into various queues. For example, the waiting queue consists of requests that are waited for execution.
- In GreenCloud, workloads are created with specific size satisfying exponential distribution.
- In iCanCloud, tasks can be submitted by the user or pre-defined model as a list and then can be added into the waiting queue for execution.

In this work, we assume that tasks are independent and not preemptive. Also, task scheduling includes when and how to allocate the particular task (cloudlet) to a suitable server (VM) based on some objectives. For example, load balancing and energy saving are important goals in scheduling policies of cloud environment. In CloudAnalyst, NetworkCloudSim, and iCanCloud, Round Robin (RR) policy is implemented as a basic choice. CloudSched develops some load balancing policies for evaluation and GreenCloud contains DVFS (Dynamic Voltage Frequency Scaling) policies to evaluate energy-saving effects.

In CloudAnalyst, NetworkCloudSim, and GreenCloud, tasks are modeled by configuring input size, processing length, and output size. In our simulation, task characteristics are 5 MB input size, 30 MB output size, and 1,200,000 MI processing length. Moreover, tasks can use all the available CPU capability on VMs (i.e., 9500 MIPS).

For comparing CloudSched with other simulators, a new method is created with start-time and end-time parameters, which refers to the lifecycle of a task. Servers (named VMs in CloudAnalyst) and tasks (named Cloudlet in CloudAnalyst) are adopted based on the Amazon EC2 specifications. The experimental environment is on a computer with a CPU core i5 and 8 GB of RAM memory. In the following, we illustrate some important and common performance metrics that are used in cloud simulators such as improving resource utilization and energy-efficient. Other metrics for different objectives can be extended easily based on these usual metrics. Note that the compared simulators use quite different metrics, here we just try to cover the metrics, which are common in the five simulators as follows.

Improving resource utilization: We can investigate the resource utilization based on two factors as follows:

- Average resource utilization: Average utilization of CPU, memory, hard disk, and network bandwidth can be obtained and used for the evaluation of different algorithms in various simulators. In addition, we can determine an integration utilization of all these resources to assessing the functionalities.
- The total number of VMs used: It can show how a cloud data center is used.

If we consider 60 s and CPU utilization of CPU i is recorded every 10 s, then the average CPU utilization of CPU i (i.e., CPU_i^U) is obtained by the average of six recorded values. Therefore, the average load on a CPU i can be determined by CPU_i^U during a specific time interval. The average utilization of all CPUs for server i can be given by Eq. (1) [144].

$$CPU_u^A = \frac{\sum_i^N CPU_i^U}{\sum_i^N CPU_i^n} \tag{1}$$

Where, CPU_i^n denotes the total number of CPUs on server i and N shows the total number of servers on a data center.

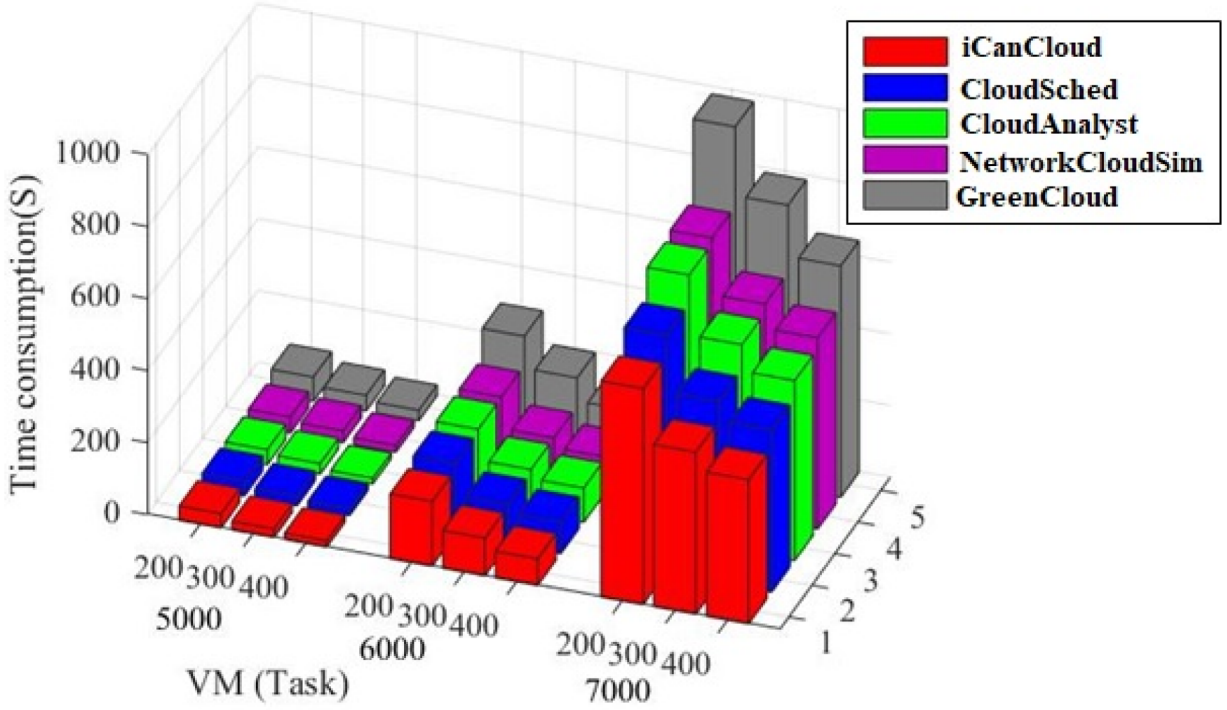


Fig. 35. Time consumption.

Similarly, the average utilization of other elements in a cloud data center (e.g., memory, network bandwidth) can be determined.

Energy consumption model: Most energy consumption modeling in data centers consist of computation processing, disk storage, network, and cooling systems. For power consumption modeling, we consider Eq. (2) that is presented in [144].

$$14.5 + 0.2\,U_{CPU} + (4.5\,e^{-8})\,U_{memory} + 0.003\,U_{disk} + (3.1\,e^{-8})U_{network} \tag{2}$$

Where, U_{CPU}, U_{memory}, U_{disk}, and $U_{network}$ indicate the utilization of CPU, memory, hard disk, and network interface, respectively.

Performance evaluation for task scheduling based simulators: In this part, we will discuss the performance comparison of iCanCloud, CloudAnalyst, NetworkCloudSim, GreenCloud, and CloudSched in term of scalability. Fig. 35 illustrates the best time consumption for different simulators, where the x-axis shows the tasks number (i.e., 5000, 6000, and 7000) and the z-axis shows the time required in a millisecond to simulate each experiment. For each experiment with a different number of tasks, three experiments with 200, 300, and 400 virtual machines are performed. In addition, Table 15 shows the result of maximum, minimum, and the average time consumption of each simulator in 20 experiments where the number of VMs is equal to 300.

From Fig. 35 and Table 15, it is observed that a larger number of tasks and VMs need more time in all simulators. It can be seen that in low load (i.e., the number of tasks is equal to 5000), CloudSched and iCanCloud complete the simulation at the same speed and faster than other simulators. One of the main features of iCanCloud is its flexible hypervisor model to present a set of brokering methods and customize them based on the requirements of system. Therefore, it can reduce task execution time by managing a set of fully customizable VMs. In iCanCloud machine, Message Passing Interface (MPI) library is installed and so it can perform parallel simulations. In other words, one experiment can be performing through several machines. Therefore, an extremely large experiment can use all the available machines in the cluster. But, other simulators perform each experiment on one machine.

The CloudSched simulator focuses on load-balancing scheduling algorithms and can simulate tens of thousands of requests in a few minutes. Unlike most cloud simulators (e.g., NetworkCloudSim and CloudAnalyst) that are based on GridSim, which makes the

Table 15
Average time consumption with 300 VMs.

Tasks	5000			6000			7000		
Simulators	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg
iCanCloud	30	24	27.5	111	105	109	462	450	457.2
CloudSched	32	25	27.6	118	110	114.3	515	503	508.6
CloudAnalyst	38	30	34.9	130	120	125	600	573	591.8
NetworkCloudSim	40	33	38.2	132	120	126.6	614	598	607.3
GreenCloud	52	44	48.5	208	199	204.3	804	785	794

system of simulation very large and complicated, CloudSched has a lightweight design to model a large-scale cloud system quickly. Additionally, CloudSched simulator presents a uniform view for all resources (i.e., physical and virtual) like Amazon EC2 applications and so the management of system and users are simplified.

While GreenCloud has the worst performance in terms of time consumption since:

- The GreenCloud simulator uses the LAG technology [76] which is a suitable methodology to increase link capacities. But this technique causes several drawbacks that limit network flexibility and performance. For example, LAG complicates the planning for the capacity of large tasks and is unpredictable even when a link fails [76]. Therefore, failure in the capacity allocation and the prediction of a link failure can increase the waiting time of tasks (i.e., simulation time) and as a result the memory consumption is increased. Moreover, common traffic patterns like ICMP and broadcast are usually routed by a single link only.
- The GreenCloud simulator is a packet-level simulator and a packet structure and protocol headers are placed in the memory during message transmission between system entities. On the contrary, other simulators such as CloudSched and iCanCloud are event-based simulators and avoid building and processing small simulation objects (like packets) individually. Therefore, CloudSched and iCanCloud show lower simulation time by considering only the effect of object interaction.
- The GreenCloud simulator models a full TCP/IP protocol to present the integration of different communication protocols such as IP, TCP, and UDP. But, using this communication modeling can limit the scalability concept and is applicable only for small data centers due to large simulation time and high memory requirements.

It can be seen from Fig. 35 and Table 15 that CloudAnalyst simulator has a better performance compared to NetworkCloudSim (in average reduces time consumption by 5.5% with 300 VMs) since it implements “CloudAppServiceBroker” component to model the service brokers and handle traffic routing between users and data centers (i.e., it routes traffic to the closest data center based on network latency). In addition, CloudAnalyst groups the elements of simulation into three levels to enhance efficiency modeling. The first level indicates a cluster of users. The second level creates user requests. The last level processes the requests. Therefore, it presents a high degree of management for experiments by defining several configuration options. For example, Internet dynamics (i.e., network delays and available bandwidth options) and the group of users with their distributions (e.g., geographically and temporally) are supported. Moreover, CloudAnalyst considers an experimental brokerage technique for peak load sharing and distributes the load of a data center among other data centers when the performance of data center is degraded from a specific threshold. Consequently, the simulation time of CloudAnalyst becomes lower than NetworkCloudSim due to load sharing.

We can observe from Fig. 35 that NetworkCloudSim has better performance in terms of time consumption compared to GreenCloud, this is because:

- It presents scalable and fast simulations by designing a network flow model for data centers utilizing bandwidth sharing and latency. This technique can significantly enhance the simulation speed for large network transfers [32].
- It considers bandwidth sharing to improve traffic flow, reduce user isolation, eliminate most bottlenecks, and thus bring a more stable service [32].
- It implements a new scheduler for each VM and so it considers the communication and computation stages of applications. Furthermore, NetworkCloudSim defines two levels for scheduling (i.e., Host level and VM level).

In high load, (i.e., when the number of tasks is equal to 6000 and 7000 tasks), the difference between time consumed by CloudSched and iCanCloud to finish the simulation becomes more apparent. The iCanCloud simulator shows better performance in execution time than CloudSched simulator due to better scalability capability. The scalability of a simulator means that it can model large-scale environments without performance degradation [78]. The main reason is related to the initialization phase. Although iCanCloud simulator needs more time to initialize each component with a great level of detail (e.g., node, switch, and links). But for large-scale simulations, the time spent in this phase is practically insignificant and those details improve the remaining simulation processes. Moreover, iCanCloud supports different configurations (e.g., RAID systems and parallel file systems) for using parallel I/O architectures that are critical for large cloud models.

Fig. 36 presents the best memory consumed by each simulator in nine different experiments. In addition, Table 16 shows the result of maximum, minimum, and average memory consumption of each simulator in 20 experiments where the number of VMs is equal to 300.

It can be noticed that iCanCloud and GreenCloud require more memory than CloudAnalyst and NetworkCloudSim. Since CloudAnalyst uses CloudAppServiceBroker component to provide load balancing among data centers. The NetworkCloudSim simulator uses a flow network model with the low computational overhead and as a result memory usage is lower in comparison with iCanCloud and GreenCloud.

In Fig. 36, iCanCloud has better performance in terms of memory consumption compared to GreenCloud. From Table 16, GreenCloud reduces this metric in average 12% since iCanCloud uses a “memory system” and presents a wide range of configurations for storage systems (e.g., NFS). The main responsibility of the memory system is assigning the necessary memory to each application. Therefore, this system determines where and how memory has to be assigned for different requests. This feature is useful for analyzing and managing the amount of memory used for each application, especially for large distributed environments.

In the low load and up to 400 VMs, the memory required by both iCanCloud and CloudAnalyst is similar. However, by increasing the number of tasks and with more than 200 VMs, the memory required by iCanCloud goes up much faster than CloudAnalyst. Generally, iCanCloud is faster for large-scale experiments and provides better scalability, but requires more memory than other

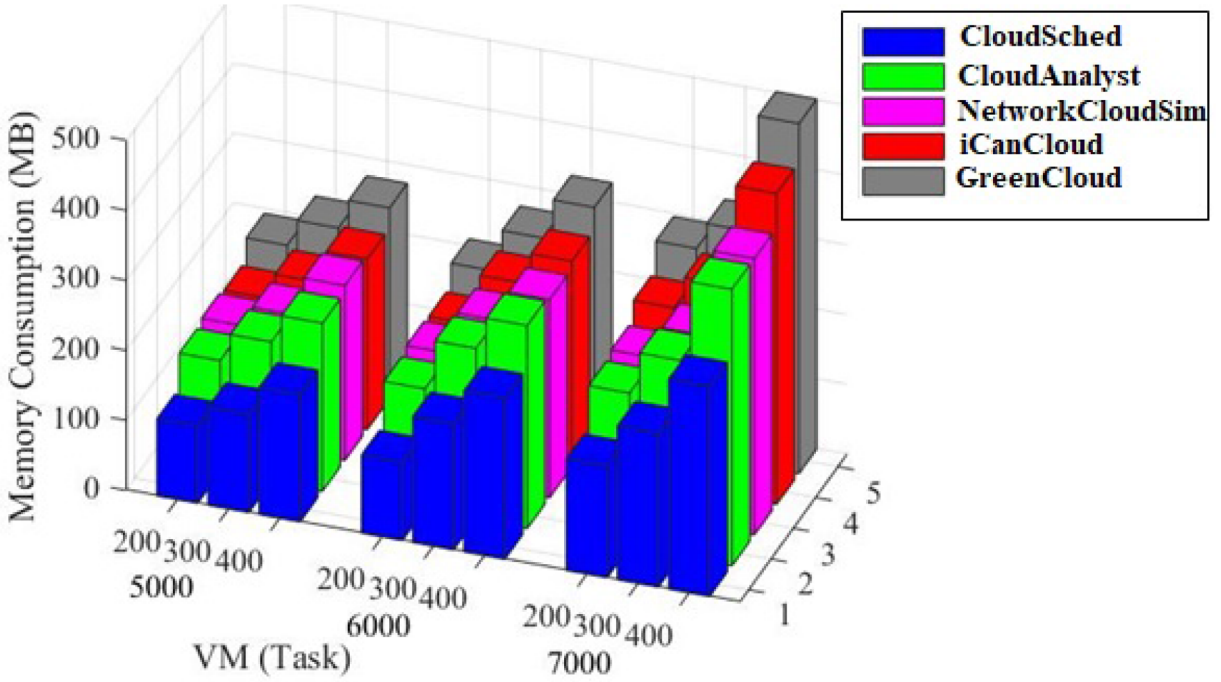


Fig. 36. Memory consumption.

Table 16
Average memory consumption with 300VMs.

Tasks	5000			6000			7000		
Simulators	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg
CloudSched	150	140	144.7	192	180	186.5	245	220	229.4
CloudAnalyst	210	200	204	252	244	248.3	286	280	282.7
NetworkCloudSim	206	195	200	245	240	242.3	280	272	277
iCanCloud	208	198	203.7	257	250	255	310	300	305
GreenCloud	240	231	236.8	279	270	273.1	340	333	337.6

simulators except for GreenCloud. GreenCloud contains an exploratory system to monitor different parameters of the system and consider various performance factors such as application workload, resource utilization, and power consumption. Therefore, it can dynamically adapt workload and resource utilization by using VM migration techniques. GreenCloud reduces unnecessary power consumption, while requires more system memory in a cloud environment.

It can be seen from Fig. 36 that CloudSched uses the lowest memory among other simulators (in average 10%, 10%, 13%, and 16% consumes memory lower than CloudAnalyst, NetworkCloudSim, iCanCloud, and GreenCloud respectively). Since CloudSched focuses on VMs scheduling at the IaaS layer, unlike existing tools (e.g., CloudAnalyst and NetworkCloudSim) which consider task level. In NewtworkCloudSim and CloudAnalyst, each request may require one or more VMs, but each request only consumes a fraction of VM capacity.

Fig. 37 shows the estimations of the energy consumption (in kWh) for the different simulation frameworks, with different numbers of VMs. In addition, Table 17 indicates the result of maximum, minimum, and the average energy consumption of each simulator in 20 experiments where number of VMs is equal to 300.

It can be observed that the estimated energy consumption in all simulation frameworks almost is at the same level in most experiments. Nevertheless, iCanCloud and GreenCloud consume lower energy during task execution since they provide the information about energy consumption in cloud data centers for the scheduling step. The main reason for improving the GreenCloud is that this tool presents detailed modeling for the energy consumption of several elements of the data center (e.g., servers, switches, and links). For specific details, GreenCloud provides a more balanced trade-off between computing performance (CPU power) and the energy consumption of server with using three different power saving modes (i.e., Dynamic Voltage/Frequency Scaling (DVFS) [76], dynamic network shutdown (DNS) [76], and combination of them DVFS + DNS). The iCanCloud simulator obtains the energy consumption of a network interface card for each state (i.e., "network on" and "network off") based on the integral of consumed power during activation time of that state. Moreover, the INET framework [78] and an energy framework known as $E - mc^2$ [153] are two features that enable iCanCloud to reduce energy consumption by management of resources. INET framework presents a set of

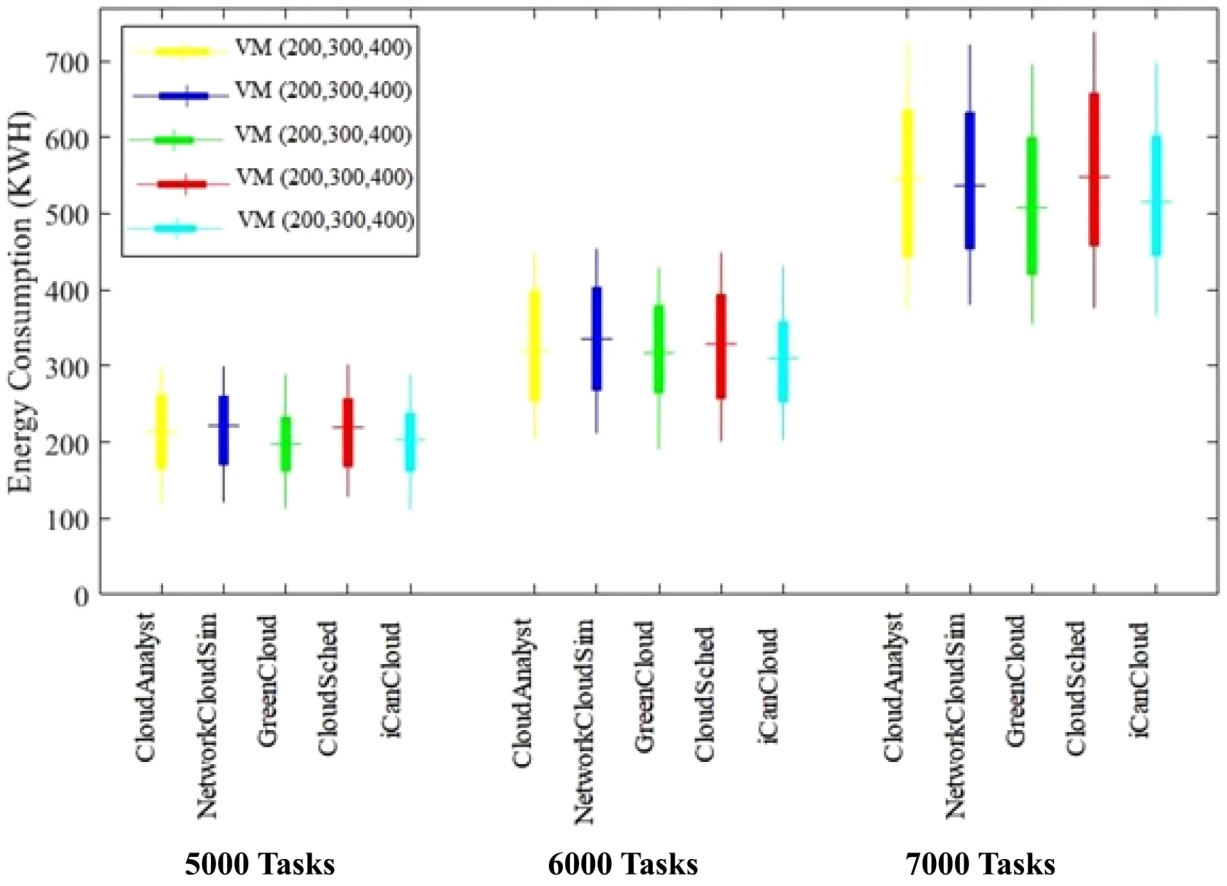


Fig. 37. Energy consumption.

Table 17
Average energy consumption with 300VMs.

Tasks	5000			6000			7000		
Simulators	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg
CloudAnalyst	210	218	213.5	307.4	326	316.8	535	550.4	544.1
NetworkCloudSim	220.9	233.6	225	312.6	338	321.7	532	548.6	542.8
GreenCloud	200	208.5	204.1	258	270.2	263.5	490.6	500.2	494.3
CloudSched	222	230	226.2	310	321.3	317.6	539.7	560.8	550.7
iCanCloud	203	211	206.3	265	275.6	270.6	492	504	500.6

important elements (e.g., routers and switches) and network protocols to control resource utilization.

5.2.2. Workflow scheduling based simulators

In the workflow scheduling problem, applications are defined by the directed acyclic graphs (DAGs). A workflow includes a set of interdependent tasks that are bounded together through data or functional dependencies. In cloud environment, effective management of scientific workflows is the main challenge since there are numerous virtual machines and many user tasks that should be scheduled by considering dependencies and various objectives.

We consider a workflow W as a graph $G = (T, D)$, where $T = \{T_1, T_2,\ldots,T_n\}$ indicates n tasks and $D = \{(T_i,\ T_j)|T_i,\ T_j \in T\}$ shows the data flow dependencies among them. The data flow dependency $(T_i,\ T_j)$ means that task T_i is an immediate predecessor of task T_j and task T_j is an immediate successor of task T_i. In some cases, there are several predecessors and successors and so $Pr(T_i)$ and $Su(T_i)$ are defined to indicate the set of immediate predecessors and successors for task T_i.

$$\mathrm{Pr}(T_i) = \{T_j|(T_j,\ T_i) \in D\} \tag{3}$$

$$Su(T_i) = \{T_j|(T_i,\ T_j) \in D\} \tag{4}$$

T_{entry} and T_{exit} are a task without any predecessors and a task without any successors, respectively.

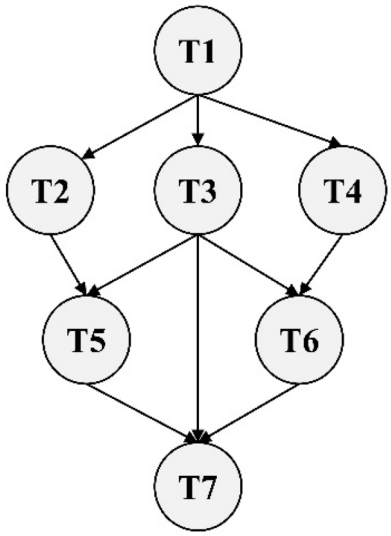


Fig. 38. An example of workflow and data precedence.

Fig. 38 presents an example of a workflow with 7 nodes and 10 edges. Generally, researchers consider that every workflow consists of only one T_{entry} and one T_{exit} [154].

Experimental setup for workflow scheduling based simulators: The makespan, cost, and CPU runtime are considered as performance metrics for evaluating workflow-based simulators (i.e., WorkflowSim, GroudSim, and ElasticSim). In this section, the experimental setups such as real-world workflows, IaaS models, simulators, and performance metrics used in the simulation are explained. We consider three types of a real-world workflow (i.e., CyberShake 30, CyberShake 50, and CyberShake 100) that are presented by the Pegasus workflow management system [155]. CyberShake is one of the well-known scheduling problem that is applied for assessing the performance of scheduling strategies. Table 18 indicates the characteristics of CyberShake 30, CyberShake 50, and CyberShake 100 workflows. Fig. 39 shows the structure of Cybershake workflow.

IaaS presents some pre-defined VMs of a cloud infrastructure for deploying user applications and is most appropriate for workflows execution. The running of VM is also named an instance in an IaaS platform. There are different types of instances like CPU capacity, network bandwidth, and memory storage (e.g., Amazon EC2 presents VM instances with various CPU capabilities for different applications). In this work, we consider the CPU capabilities and bandwidths that are affecting task execution time and data transformation time. In addition, all the instances that are available in an IaaS platform are represented by $I = \bigcup_{s=1}^{\infty} \{I_s\}$. We define the m types of instance with $R = \bigcup_{k=1}^{m} \{R_k\}$ and so $R_k = Type(I_s)$ denotes the relation that instance I_s belongs to type R_k.

The providers of IaaS define the CPU capabilities of different instance types by the concept of computing unit (CU) to determine the execution time of tasks. The larger the CU means the higher computing performance of the instance. Actual execution time $AET(T_i)$ of task T_i on instance I_s can be obtained based on Eq. (5) [154].

$$AET(T_i) = \frac{RT(T_i)}{CU(Type(I_s))} \tag{5}$$

Where $CU(T_i)$ represents the compute unit of T_i and $RT(T_i)$ indicates the reference execution time of task T_i.

The communication bandwidths among instances and the size of transferred data can be used for determining the communication time. In the same communication link, two data transfers do not occur simultaneously and so we can consider that using full bandwidth is allowed by each data transfer.

The communication time $CT(T_i, T_j)$ between tasks T_i and T_j that is executed on instances I_α and I_β, is obtained by Eq. (6).

Table 18
Real-workflow characteristics [154].

Workflow	No. of nodes	No. of edges	Average size of data	Average task execution (CU = 1)
CyberShake 30	30	112	747.48 MB	23.77s
CyberShake 50	50	188	864.74MB	29.32s
CyberShake 100	100	380	849.60MB	31.53s

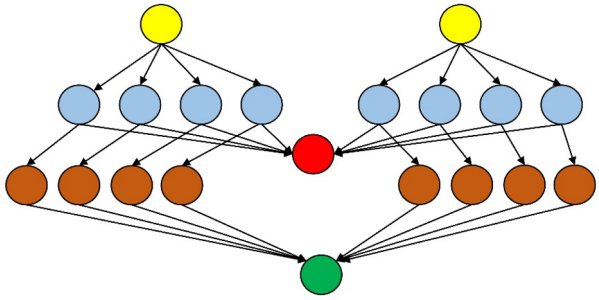


Fig. 39. Structure of Cybershake workflow [155].

$$CT(T_i, T_j) = \begin{cases} \frac{SizeD(T_i, T_j)}{Min\{Bw(Type(I_\alpha), Bw(Type(I_\beta))\}} & I_\alpha \neq I_\beta \\ 0 & I_\alpha \neq I_\beta \end{cases} \quad (6)$$

Where $Bw(I_k)$ is the bandwidth of instance type I_k and $SizeD(T_i, T_j)$ denotes the size of data that is transferred from T_i to T_j.

We try to minimize makespan and cost simultaneously during workflows scheduling in cloud. The makespan is calculated by Eq. (7) as follows [154].

$$Makespan = FT(T_{exit}) \quad (7)$$

Where $FT(T_{exit})$ can be computed by Eq. (8):

$$FT(T_i) = ST(T_i) + AET(T_i) \quad (8)$$

Where $ST(T_i)$ and $AET(T_i)$ indicate the start time and actual execution time of task T_i, respectively.

If we consider the set of pricing options ($Pr = \{Pr_1, Pr_2, ..., Pr_h\}$) for services of IaaS platform and $CM(Pr_h, R_k, I_s)$ as the monetary cost of executing instance I_s with type R_k based on the pricing model Pr_h. Then, the total monetary cost for all tasks execution in the workflow is obtained by Eq. (9) [154]:

$$Cost = \sum_{I_s \in I^*} CM(Pr_h, Type(I_s), I_s) \quad (9)$$

Where $I^* = \{I_s | \exists T_i \in T: Ins(T_i) = I_s\}$ denotes the necessary instances for running all tasks of T.

In this work, we consider the well-known pricing model of Amazon EC2 (i.e., General Purpose instance group in US East region with On-Demand Instance [154]) as our pricing scheme that is illustrated in Table 19.

Now, we can define workflow scheduling problem based on a given a workflow (i.e., $G = (T, D)$) and an IaaS platform (i.e., $S = (I, R, Pr)$). The proposed algorithm aims to determine the scheduling order of tasks based on the dependency constraints and task-to-instance mapping and instance-to-type mapping in such a way makespan (given in Eq. (7)) and cost (given in Eq. (9)) are minimized.

Note that the dependencies among tasks are considered in the task scheduling order. We perform all the experiments on a desktop PC equipped with a 2.3 GHz Intel Core i5 CPU and 8GB RAM. We develop RR algorithm as scheduler method in WorkflowSim, GroudSim, and ElasticSim to run Cybershake workflow. Two comparative experiments are carried out to see the performance of different simulators. Firstly, we compare the CPU runtime taken for different simulators. Secondly, we compare the obtained cost-makespan trade-off in WorkflowSim, GroudSim, and ElasticSim.

Performance evaluation for workflow scheduling based simulators: In this section, we compare the performance of three simulators (i.e., WorkflowSim, GroudSim, and ElasticSim) that mostly used for workflow scheduling problems in terms of average CPU runtime and cost-makespan. Table 20 gives the average CPU runtime statistics by comparative simulators for executing Cybershake workflow.

Table 19
Parameters of IaaS [154].

Instance Type	Compute Unit	Bandwidth (Mb/s)	Price ($)
M4.large	6.5	56.25	0.120
M4.xlarge	13	93.75	0.239
M4.2xlarge	26	125	0.479
M4.4xlarge	53.5	250	0.958
M4.10xlarge	124.5	500	2.394
M3.medium	3	56.25	0.067
M3.large	6.5	56.25	0.133
M3.xlarge	13	62.5	0.266
M3.2xlarge	26	125	0.532

Table 20
Average runtime and runtime ratios of three simulators for CyberShake workflow.

Workflow	Average Run time of three simulators			Runtime ratios(against ElasticSim)	
	WorkflowSim	Elasticsim	Groudsim	WorkflowSim	GroudSim
CyberShake30	7.41	5.13	9.02	1.45	1.78
CyberShake50	9.82	7.34	11.12	1.35	1.75
CyberShake100	12.53	8.51	13.44	1.47	1.57

It is obvious that the execution time of all simulators increases with the number of tasks increasing. The results show that ElasticSim is much more computationally efficient than the two comparative simulators. Especially, the CPU run-time of ElasticSim, WorkflowSim, and GroudSim in the ranges of [5.13 s, 8.51 s], [7.41 s, 12.53 s], and [9.12 s, 13.24 s], respectively. Since ElasticSim presents a suitable structure for virtual machines and also it has a resource runtime auto-scaling and stochastic task execution time modeling.

It can be seen from Table 20 that WorkflowSim achieves better CPU runtime compared to GroudSim (i.e., reduces CPU runtime by 13.5%) since the clustering engine module is implemented in WorkflowSim simulator. The clustering engine merges tasks into jobs to decrease execution overhead. The computational granularity of workflow tasks is increased since the number of jobs to be executed in a workflow is reduced after clustering [58]. On the other hand, WorkflowSim achieves higher CPU runtime compared to ElasticSim (in average 30% higher than ElasticSim) and the main reason is that WorkflowSim consists of two components for the simulation of failures (i.e., Failure Generator and Failure Monitor). Failure Generator randomly creates task failures and Failure Monitor returns the failure information to workflow management so that it can adapt the scheduling process dynamically.

We list the runtime ratios of the two comparative algorithms against ElasticSim, which directly validate the time efficiency of ElasticSim as a better simulator in terms of CPU runtime. For example, the CPU runtime of GroudSim is higher than ElasticSim since GroudSim follows a task queuing approach by placing the tasks into a waiting queue until a CPU becomes available.

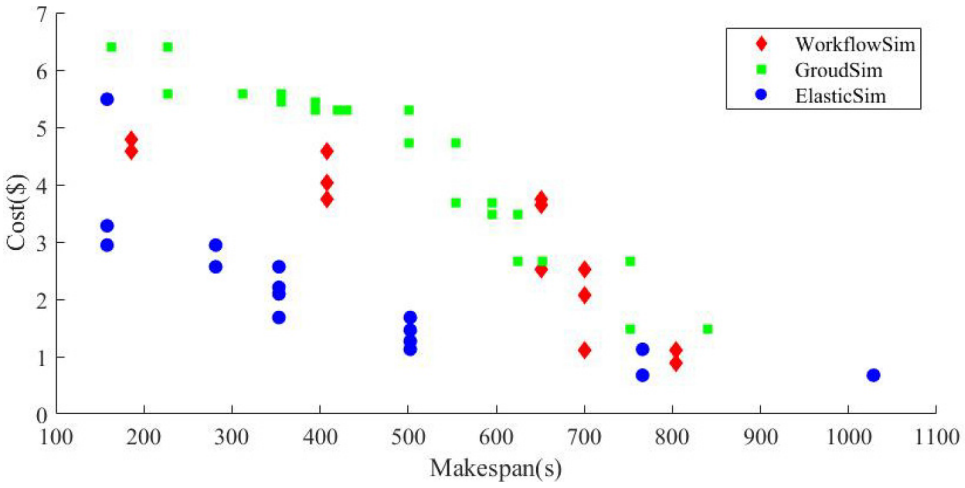
Fig. 40 plots the produced cost-makespan trade-offs for ElasticSim, WorkflowSim, and GroudSim, based on a real-world workflow CyberShake with varying number of nodes. Note that all the x-axes are logarithmic and each point on the plot represents a possible task schedule. The plots are given in Fig. 40 indicates that the trade-off fronts obtained by ElasticSim are significantly superior to those obtained by the comparative algorithms. In other words, ElasticSim can provide better cost-makespan trade-offs in task scheduling. This is because for two reasons:

(1) ElasticSim applies a resource runtime auto-scaling and stochastic task execution time modeling. Adjusting the scale of resources during the workflow runtime dynamically (i.e., resource runtime auto-scaling) can improve the utilization of resources and decrease the resource rental cost. Furthermore, it introduces a delegate algorithm to transform the static scheduling algorithm into practical VM renting based on task execution failures. While WorkflowSim and GroudSim which are an extension of CloudSim and GridSim consider a higher layer of workflow management. They assume that the resources of the system are rented in advance and kept unchanged during the workflow runtime. At the same time, task execution times in these two simulators are also assumed to be deterministic. Therefore, they do not succeed in the evaluation of the workflow scheduling algorithms based on resource runtime auto-scaling.
(2) ElasticSim applies *Rent on-predict* as a VM renting strategy. In this strategy, the requests of VM renting are submitted to guarantee that necessary VMs are available at the predicted required times and so the workflow makespan is reduced.

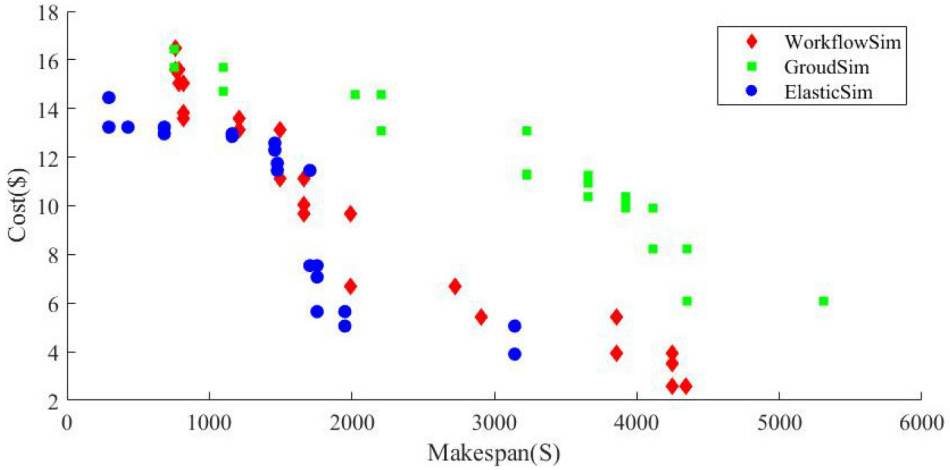
6. Conclusions and future works

The popularity and complexity of distributed computing systems (e.g., cloud) makes simulation toolkit as an essential choice for developing, configuring, and evaluating the system performance. During the past few years, we can see a significant increase for designing tools to model cloud computing which vary in terms of their utility and capabilities. In this paper, we present a review and multi-level feature analysis of 33 simulators for cloud environment. Based on this evaluation, we observe that none of them is complete and ideal for all aspects and they still need some improvement. One solution is to use various software or their combinations for various optimization objectives like load balancing and energy-efficiency. For future work, some open research challenges are listed as following:

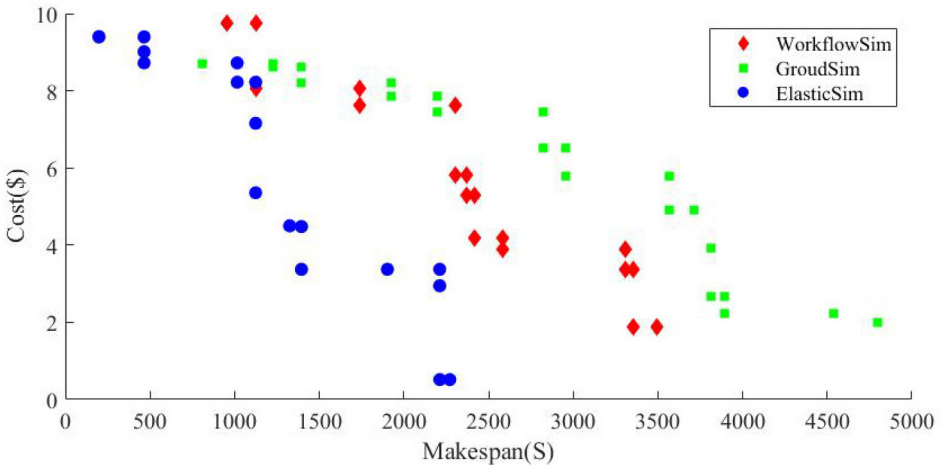
(1) Security aspects of cloud: As we compared in the paper, most tools focus on performance and energy modeling. Currently there is still lack of tools that can consider all the security approaches as encryption, decryption, encapsulation, authentication, and privacy assurance for components of cloud.
(2) User priority: If toolkit provides priority methods for users to have various priorities types of resources then it can model the realistic scenarios of cloud.
(3) Easy to use and customizable: It is essential that toolkit can accept inputs from text file and user can easily set all necessary parameters. In addition, toolkit guarantees that developer can repeat experiments and obtain the identical results.
(4) Profit and cost model: The definition of different profit models in cloud simulator is necessary. Since cloud service provider tries to maximize profit within the satisfactory level of service quality specified by the service consumer.



(a) CyberShake_30



(b) CyberShake_50



(c) CyberShake_100

Fig. 40. Cost-makespan trade-offs for different simulators.

(5) Energy consumption model: Due to the poor scalability of the existing simulators, accurate prediction of power consumption for large systems is difficult. Therefore, considering the parallel versions of the simulators in order to scale to millions of cloud data centers and obtain the safe results is necessary.
(6) Distributed execution: The review indicates a lack of tool support for distributed execution. This ability scaling up for load balancing if the several simulation runs require to be executed at the same time.
(7) Congestion control and datacenter traffic patterns: Absence of various types of congestion control and real patterns for datacenter traffic in most simulators is a main challenge. Since structure, protocols, and real time traffic patterns affect network performance. This leads to the conclusion that network addressing schemes and traffic flows demand some attention.
(8) Other related fields: The available simulators can be extended to provide suitable tool for another emerging field known as mobile cloud computing. It is necessary to focus on emerging cloud use cases e.g. HPC in the cloud, Edge and Fog computing, and IoT.
(9) Cloud federation models: The standardization of cloud service descriptions and exchange leads to appearance of cloud federations. It is necessary to design a framework to simulate many federated cloud scenarios while respecting SLAs. These scenarios should be consisting of the optimization of the total resource usage, the power usage, and the obtained profit per cloud service provider.

References

[1] N. Subramanian, A. Jeyaraj, Recent security challenges in cloud computing, Comput. Electr. Eng. 71 (2018) 28–42.
[2] X. Zhang, T. Wu, M. Chen, T. Wei, J. Zhou, S. Hu, R. Buyya, Energy-aware virtual machine allocation for cloud with resource reservation, J. Syst. Softw. 147 (2019) 147–161.
[3] N. Mansouri, B. Mohammad Hasani Zade, M.M. Javidi, Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory, Comput. Ind. Eng. 130 (2019) 597–633.
[4] Y.K. Suh, K.Y. Lee, A survey of simulation provenance systems: modeling, capturing, querying, visualization, and advanced utilization, Hum. Centric Comput. Inf. Sci. 8 (2018) 1–29.
[5] S.S. Devesh, C. Jinwala, S. Garg, A survey of simulators for P2P overlay networks with a case study of the P2P tree overlay using an event-driven simulator, Eng. Sci. Technol. Int. J. 20 (2) (2017) 705–720.
[6] W. Zhao, Y. Peng, F. Xie, Z. Dai, Modeling and simulation of cloud computing: a review, Proceedings of the IEEE Asia Pacific Cloud Computing Congress, 2012, pp. 20–24.
[7] R. Singh, P. Patel, P. Singh, Cloud simulators: a review, Int. J. Adv. Comput. Electron. Technol. 2 (2) (2015) 62–67.
[8] M.A. Sharkh, A. Kanso, A. Shami, P. Öhlén, Building a cloud on earth: a study of cloud computing data center simulators, Comput. Netw. 108 (2016) 78–96.
[9] M.S. Bhatia, M. Sharma, A critical review & analysis of cloud computing simulators, Int. J. Latest Trends Eng. Technol. (2016) 29–36.
[10] J. Byrne, S. Svorobej, K.M. Giannoutakis, D. Tzovaras, P.J. Byrne, P.O. Östberg, A. Gourinovitch, T. Lynn, A review of cloud computing simulation platforms and related environments, Proceedings of the 7th International Conference on Cloud Computing and Services Science, 2017, pp. 651–663.
[11] F. Fakhfakh, H. Hadj Kacem, A. Hadj Kacem, Simulation tools for cloud computing: a survey and comparative study, Proceedings of the IEEE/ACIS 16th International Conference on Computer and Information Science, 2017, pp. 221–226.
[12] N. Mansouri, M.M. Javidi, A review of data replication based on meta-heuristics approach in cloud computing and data grid, Soft. Comput. (2020).
[13] N. Mansouri, M.M. Javidi, Cost-based job scheduling strategy in cloud computing environments, Distrib. Parallel Databases (2019).
[14] D. Perez, A. Karima, V.M. Curado, E. Monteiro, A comparative analysis of simulators for the Cloud to Fog continuum, Simul. Model. Pract. Theory 101 (2020) 1–63.
[15] Amazon Elastic Compute Cloud (EC2). Available at:http://www.amazon.com/ec2/.
[16] D. Chappell, Introducing The Azure Services Platform, White Paper, 2008.
[17] Google App Engine. Available at:http://appengine.google.com.
[18] H. Mi, H. Wang, H. Cai, Y. Zhou, M.R. Lyu, Z. Chen, P-Tracer: path-based performance profiling in cloud computing systems, Proceedings of the 36th IEEE International Conference on Computer Software and Applications, 2012, pp. 509–514.
[19] M. Kumar, S.C. Sharma, PSO-COGENT: cost and energy efficient scheduling in cloud environment with deadline constraint, Sustain. Comput. Inform. Syst. 19 (2018) 147–164.
[20] K. Gupta, R. Beri, Cloud computing: a survey on cloud simulation tools, Int. J. Innov. Res. Sci. Technol. 2 (11) (2016) 430–434.
[21] R.N. Calheiros, R. Ranjan, C.A.F. De Rose, R. Buyya, CloudSim: A Novel Framework For Modeling and Simulation of Cloud Computing Infrastructures and Services, Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, 2009, pp. 1–9. Technical ReportGRIDS-TR-2009-1.
[22] B. Quetier, F. Cappello, A survey of grid research tools: simulators, emulators and real life platforms, In: Proceedings of the 17th IMACS World Congress, 1–8.
[23] I.C. Legrand, H.B. Newman, The MONARC toolset for simulating large network-distributed processing systems, Proceedings of the 2000 Winter Simulation Conference, 2000, pp. 1794–1801.
[24] A. Legrand, L. Marchal, H. Casanova, Scheduling distributed applications: the SimGrid simulation framework, Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid, 2003, pp. 1–8.
[25] R. Buyya, M. Murshed, Gridsim: a toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing, Concurr. Comput. Pract. Exp. 14 (2002) 1175–1220.
[26] W.H. Bell, D.G. Cameron, L. Capozza, A.P. Millar, K. Stockinger, F. Zini, Simulation of dynamic grid replication strategies in OptorSim, Proceedings of the 3rd International Workshop on Grid Computing, 2002, pp. 46–57.
[27] H.J. Song, X. Liu, D. Jakobsen, R. Bhagwan, X. Zhang, K. Taura, A. Chien, The MicroGrid: a scientific tool for modeling computational grids, Proceedings of the 2000 ACM/IEEE Conference on Supercomputing, 2000, pp. 1–22.
[28] W. Tsai, G. Qi, Y. Chen, A cost-effective intelligent configuration model in cloud computing, Proceedings of the International Conference on Distributed Computing Systems Workshops, 2012, pp. 400–408.
[29] M. Jammal, T. Singh, A. Shami, R. Asal, Y. Li, Software defined networking: state of the art and research challenges, Comput. Netw. 72 (2014) 74–98.
[30] H. Hawilo, A. Shami, M. Mirahmadi, R. Asal, NFV: state of the art, challenges, and implementation in next generation mobile networks (vEPC), IEEE Netw. 28 (6) (2014) 18–26.
[31] B. Wang, Z. Qi, R. Ma, H. Guan, A survey on data center networking in cloud era, Commun. Survey Tutor. J. (2013).
[32] S.K. Garg, R. Buyya, Networkcloudsim: modelling parallel applications in cloud simulations, Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing, 2011, pp. 105–113.
[33] M.A. Sharkh, M. Jammal, A. Ouda, A. Shami, Resource allocation in a network-based cloud computing environment: design challenges, IEEE Commun. Mag. 51 (11) (2013) 46–52.
[34] S.T. Milan, L. Rajabion, H. Ranjbar, N.J. Navimipour, Nature inspired meta-heuristic algorithms for solving the load-balancing problem in cloud environments, Comput. Oper. Res. 110 (2019) 159–187.
[35] C. Mergenci, I. Korpeoglu, Generic resource allocation metrics and methods for heterogeneous cloud infrastructures, J. Netw. Comput. Appl. 146 (2019) 1–16.
[36] A. Wilczyński, J. Kołodziej, Modelling and simulation of security-aware task scheduling in cloud computing based on Blockchain technology, Simul. Model.

Pract. Theory 99 (2020) 1–45.
[37] T. Biswas, P. Kuila, A.K. Ray, M. Sarkar, Gravitational search algorithm based novel workflow scheduling for heterogeneous computing systems, Simul. Model. Pract. Theory 96 (2019) 1–21.
[38] M. Safari, R. Khorsand, Energy-aware scheduling algorithm for time-constrained workflow tasks in DVFS-enabled cloud environment, Simul. Model. Pract. Theory 87 (2018) 311–326.
[39] X. Xu, J. Li, H. Yu, L. Luo, X. Wei, G. Sun, Towards Yo-Yo attack mitigation in cloud auto-scaling mechanism, Digit. Commun. Netw. (2019) 1–10.
[40] N. Mansouri, Network and data location aware approach for simultaneous job scheduling and data replication in large-scale data grid environments, Front. Comput. Sci. 8 (3) (2014) 391–408.
[41] N. Mansouri, M.M. Javidi, A hybrid data replication strategy with fuzzy-based deletion for heterogeneous cloud data centers, J. Supercomput. 74 (10) (2018) 5349–5372.
[42] N. Mansouri, M.M. Javidi, B. Mohammad Hasani Zade, Using data mining techniques to improve replica management in cloud environment, Soft Comput. (2019).
[43] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A.F. De Rose, R. Buyya, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithm, Softw. Pract. Exp. 41 (1) (2011) 23–50.
[44] B. Wickremasinghe, R.N. Calheiros, R. Buyya, CloudAnalyst: a CloudSim-based visual modeller for analysing cloud computing environments and applications, Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications, 2010, pp. 446–452.
[45] R.N. Calheiros, M.A.S. Netto, C.A.F. De Rose, R. Buyya, EMUSIM: an integrated emulation and simulation environment for modeling, evaluation, and validation of performance of cloud computing applications, Softw. Pract. Exp. 43 (5) (2013) 595–612.
[46] R.N. Calheiros, R. Buyya, C.A.F. De Rose, Building an automated and self-configurable emulation testbed for grid applications, Softw. Pract. Exp. 40 (5) (2010) 405–429.
[47] F. Fittkau, S. Frey, W. Hasselbring, CDOSim: simulating cloud deployment options for software migration support, Proceedings of the IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems, 2012, pp. 37–46.
[48] S. Frey, W. Hasselbring, B. Schnoor, Automatic conformance checking for migrating software systems to cloud infrastructures and platforms, J. Softw. Maint. Evol. Res. Pract. 25 (10) (2013) 1089–1115.
[49] http://www.cloudmig.org, last accessed 2012-07-05.
[50] Y. Jararweh, Z. Alshara, M. Jarrah, M. Kharbutli, M.N. Alsaleh, TeachCloud: a cloud computing educational toolkit, Proceedings of the IBM Cloud Academy Conference, 2012, pp. 1–19.
[51] J. Dean, S. Ghemawat, MapReduce: simplified data processing on large clusters, Commun. ACM 51 (1) (2008).
[52] X. Li, X. Jiang, P. Huang, K. Ye, DartCSim: an enhanced user-friendly cloud simulation system based on CloudSim with better performance, Proceedings of the IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, 2012, pp. 392–396.
[53] X. Li, X. Jiang, K. Ye, P. Huang, DartCSim+: enhanced CloudSim with the power and network models integrated, Proceedings of the IEEE Sixth International Conference on Cloud Computing, 2013, pp. 644–651.
[54] Z. Cai, Q. Li, X. Li, ElasticSim: a toolkit for simulating workflows with cloud resource runtime auto-scaling and stochastic task execution times, J. Grid Comput. 15 (2016) 257–272.
[55] A. Kohne, M. Spohr, L. Nagel, O. Spinczyk, FederatedCloudSim: a SLA-aware federated cloud simulation framework, Proceedings of the 2nd International Workshop on CrossCloud Systems, 2014, pp. 1–5.
[56] A. Zhou, S. Wang, Q. Sun, H. Zou, F. Yang, FTCloudSim: a simulation tool for cloud service reliability enhancement mechanisms, Proceedings Demo and Poster Track of ACM/IFIP/USENIX International Middleware Conference, 2013.
[57] B. Schroeder, G.A. Gibson, A large-scale study of failures in high-performance computing systems, IEEE Trans. Depend. Secure Comput. 7 (4) (2009) 337–350.
[58] W. Chen, E. Deelman, WorkflowSim: a toolkit for simulating scientific workflows in distributed environments, Proceedings of the IEEE 8th International Conference on E-Science, 2012.
[59] T. Teixeira Sá, R.N. Calheiros, D.G. Gomes, CloudReports: an extensible simulation tool for energy-aware cloud computing environment, Cloud Comput. (2014) 127–142.
[60] W.A. Higashino, M.A.M. Capretz, L.F. Bittencourt, CEPSim: a simulator for cloud-based complex event processing, Proceedings of the IEEE International Congress on Big Data, 65 2015, pp. 122–139.
[61] D.C. Luckham, Rapide: A Language and Toolset For Simulation of Distributed Systems By Partial Orderings of Events, Partial Order Methods in Verification, Computer Systems Laboratory, Stanford University, 1996.
[62] D.J. Abadi, D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, M. Stonebraker, N. Tatbul, S. Zdonik, Aurora: a new model and architecture for data stream management, VLDB J. 12 (2003) 120–139.
[63] A. Arasu, S. Babu, J. Widom, The CQL continuous query language: semantic foundations and query execution, VLDB J. 15 (2) (2005) 121–142.
[64] M. Bux, U. Leser, DynamicCloudSim: simulating heterogeneity in computational clouds, Futur. Gener. Comput. Syst. 46 (2015) 85–99.
[65] Y. Jararweh, M. Jarrah, M. kharbutli, Z. Alshara, M.N. Alsaleh, M. Al-Ayyoub, CloudExp: a comprehensive cloud computing experimental framework, Simul. Model. Pract. Theory 49 (2014) 180–192.
[66] A. Beitch, B. Liu, T. Yung, R. Griffith, A. Fox, D.A. Patterson, Rain: A Workload Generation Toolkit For Cloud Computing Applications, University of California at Berkeley, 2010 Technical Report.
[67] D.C. Alves, B.G. Batista, D.M.L. Filho, M.L. Peixoto, S. Reiff-Marganiec, B.T. Kuehne, CM Cloud simulator: a cost model simulator module for Cloudsim, Proceedings of the IEEE World Congress on Services, 2016, pp. 99–102.
[68] J. Jung, H. Kim, MR-CloudSim: designing and implementing MapReduce computing model on CloudSim, Proceedings of the International Conference on ICT Convergence, 2012, pp. 504–509.
[69] M.H. Sqalli, F.B. M.Al-saeedi, M. Siddiqui, UCloud: a simulated Hybrid Cloud for a university environment, Proceedings of the IEEE 1st International Conference on Cloud Networking, 2012, pp. 170–172.
[70] S.H. Lim, B. Sharma, G. Nam, E.K. Kim, C.R. Das, MDCSim: a multi-tier data center simulation, platform, Proceedings of the IEEE International Conference on Cluster Computing and Workshops, 2009, pp. 1–9.
[71] C. Amza, E. Cecchet, A. Chanda, A.L. Cox, S. Elnikety, E.N. Elnozahy, R. Gil, J. Marguerite, K. Rajamani, W. Zwaenepoel, Bottleneck characterization of dynamic web site benchmakrs, Proceedings of the Third IBM CAS Conference, 2002.
[72] S.K.S. Gupta, R.R. Gilbert, A. Banerjee, Z. Abbasi, T. Mukherjee, G. Varsamopoulos, GDCSim: a tool for analyzing green data center design and resource management techniques, Proceedings of the International Green Computing Conference and Workshops, 2011, pp. 1–8.
[73] U. Singh, A. Singh, S. Parvez, A. Sivasubramaniam, CFD-Based operational thermal efficiency improvement of a production data center, Proceedings of the First USENIX Conference on Sustainable Information Technology (SustainIT'10), 2010, pp. 1–7.
[74] T. Cucinotta, A. Santogidis, CloudNetSim - simulation of real-time cloud computing applications, Proceedings of the 4th International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems, 2013.
[75] A.W. Malik, K. Bilal, K. Aziz, D. Kliazovich, N. Ghani, S.U. Khan, R. Buyya, CloudNetSim++: a toolkit for data center simulations in OMNET++, Proceedings of the 11th Annual High Capacity Optical Networks and Emerging/Enabling Technologies, 2014, pp. 104–108.
[76] D. Kliazovich, P. Bouvry, S.U. Khan, GreenCloud: a packet-level simulator of energy-aware cloud computing data centers, J. Supercomput. 62 (2012) 1263–1283.
[77] The Network Simulator Ns2 (2010) Available at:http://www.isi.edu/nsnam/ns/.
[78] A. Núñez, J.L. Vázquez-Poletti, A.C. Caminero, G.G. Castañé, J. Carretero, I.M. Llorente, iCanCloud: a flexible and scalable cloud infrastructure simulator, J. Grid Comput. 10 (2012) 185–209.
[79] U.U. Rehman, A. Ali, Z. Anwar, secCloudSim: secure cloud simulator, Proceedings of the 12th International Conference on Frontiers of Information Technology,

2014, pp. 208–213.
[80] W. Simpson, PPP challenge handshake authentication protocol (CHAP), Internet Eng. Task Force (1996).
[81] S. Ostermann, K. Plankensteiner, R. Prodan, T. Fahringer, GroudSim: an event-based simulation framework for computational grids and clouds, Proceedings of the European Conference on Parallel Processing, 2011, pp. 305–313.
[82] W. Tian, Y. Zhao, M. Xu, Y. Zhong, X. Sun, A toolkit for modeling and simulation of real-time virtual machine allocation in a cloud data center, IEEE Trans. Autom. Sci. Eng. 12 (1) (2015) 153–161.
[83] Hebrew University, Experimental Systems Lab, 2012. [Online]. Available:www.cs.huji.ac.il/labs/parallel/workload.
[84] S. Sotiriadis, N. Bessis, N. Antonopoulos, A. Anjum, SimIC: designing a new inter-cloud simulation platform for integrating large-scale resource management, Proceedings of the IEEE 27th International Conference on Advanced Information Networking and Applications, 2013, pp. 90–97.
[85] I. Sriram, SPECI, a Simulation tool exploring cloud-scale data centres, Proceedings of the IEEE International Conference on Cloud Computing, 2009, pp. 381–392.
[86] A. Buss, Component based simulation modeling with Simkit, Proceedings of the Winter Simulation Conference, 2002, pp. 243–249.
[87] D. Fernández-Cerero, A. Fernández-Montes, A. Jakóbik, J. Kołodziej, M. Toro, SCORE: simulator for cloud optimization of resources and energy consumption, Simul. Model. Pract. Theory 82 (2018) 160–173.
[88] M. Schwarzkopf, A. Konwinski, M. Abd-El-Malek, J. Wilkes, Omega: flexible, scalable schedulers for large compute clusters, Proceedings of the 8th ACM European Conference on Computer Systems, 2013, pp. 351–364.
[89] D. Fernández-Cerero, A. Jakóbik, A. Fernández-Montes, J. Kołodziej, GAME-SCORE: game-based energy-aware cloud scheduler and simulator for computational clouds, Simul. Model. Pract. Theory 93 (2019) 3–20.
[90] D. Fernández-Cerero, A. Jakóbik, A. Fernández-Montes, J. Kołodziej, Stackelberg Game-based models in energy-aware cloud scheduling, Proceedings of the 32nd Conference on Modelling and Simulation, 2018, pp. 460–467.
[91] G. Kecskemeti, DISSECT-CF: a simulator to foster energy-aware scheduling in infrastructure clouds, Simul. Model. Pract. Theory 58 (2) (2015) 188–218.
[92] N. Mansouri, M.M. Javidi, A new prefetching-aware data replication to decrease access latency in cloud environment, J. Syst. Softw. 144 (2018) 197–215.
[93] I. Moreno, P. Garraghan, P. Townend, J. Xu, An approach for characterizing workloads in google cloud to derive realistic resource utilization models, Proceedings of the IEEE Seventh International Symposium on Service-Oriented System Engineering, 2013, pp. 49–60.
[94] P. Samimia, Y. Teimourib, M. Mukhtar, A combinatorial double auction resource allocation model in cloud computing, Inf. Sci. 357 (2014).
[95] I.S. Moreno, P. Garraghan, P. Townend, J. Xu, Analysis, Modeling and simulation of workload patterns in a large-scale utility cloud, IEEE Trans. Cloud Comput. 2 (2) (2014) 208–221.
[96] D. Limbani, B. Oza, A proposed service broker strategy in CloudAnalyst for cost effective data center selection, Int. J. Eng. Res. Appl. 2 (1) (2012) 793–797.
[97] R.K. Mishra, S.N. Bhukya, Service broker algorithm for cloud-analyst, Int. J. Comput. Sci. Inf. Technol. 5 (3) (2014) 3957–3962.
[98] A. Ahmed, Y. Singh, Analytic study of load balancing techniques using tool cloud analyst, Int. J. Eng. Res. Appl. 2 (2) (2012) 1027–1030.
[99] B. Mondal, K. Dasgupta, P. Dutta, Load balancing in cloud computing using stochastic hill climbing-a soft computing approach, Procedia Technol. 4 (2012) 783–789.
[100] S.E. Kafhali, K. Salah, Modeling and analysis of performance and energy consumption in cloud data centers, Arab. J. Sci. Eng. 43 (12) (2018) 7789–7802.
[101] P. Kathiravelu, L. Veiga, Software-defined simulations for continuous development of cloud and data center networks, Proceedings of the OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", 2016, pp. 3–23.
[102] C. Li, H. Zhuang, Q. Wang, X. Zhou, SSLB: self-similarity-based load balancing for large-scale fog computing, Arab. J. Sci. Eng. 43 (2018) 7487–7498.
[103] S. Frey, F. Fittkau, W. Hasselbring, Optimizing the deployment of software in the cloud, Proceedings of the Conference: Software Engineering & Management, 2015.
[104] S. Frey, F. Fittkau, W. Hasselbring, Search-based genetic optimization for deployment and reconfiguration of software in the cloud, Proceedings of the 35th International Conference on Software Engineering, 2013, pp. 512–521.
[105] M. Quwaider, Y. Jararweh, Cloudlet-based efficient data collection in wireless body area networks, Simul. Model. Pract. Theory 50 (2015) 57–71.
[106] Q. Althebyan, O. ALQudah, Y. Jararweh, Q. Yaseen, Multi-threading based map reduce tasks scheduling, Proceedings of the 5th International Conference on Information and Communication Systems, 2014, pp. 1–6.
[107] Y. Zhang, X. Cheng, L. Chen, H. Shen, Energy-efficient tasks scheduling heuristics with multi-constraints in virtualized clouds, J. Grid Comput. 16 (2018) 459–475.
[108] Z. Cai, X. Li, R. Ruiz, Q. Li, A delay-based dynamic scheduling algorithm for bag-of-task workflows with stochastic task execution times in clouds, Futur. Gener. Comput. Syst. 71 (2017) 57–72.
[109] A. Kohne, M. Krüger, M. Pfahlberg, O. Spinczyk, L. Nagel, Financial evaluation of SLA-based VM scheduling strategies for cloud federations, Proceedings of the 4th Workshop, 2017.
[110] D. Oliveira, A. Brinkmann, N. Rosa, P. Maciel, Performability evaluation and optimization of workflow applications in cloud environments, J. Grid Comput. 17 (4) (2019) 749–770.
[111] A. Zhou, S. Wang, C.H. Hsu, Q. Sun, F. Yang, Task rescheduling optimization to minimize network resource consumption, Multimed. Tools Appl. 75 (20) (2015) 12901–12917.
[112] A. Zhou, S. Wang, C.H. Hsu, M.H. Kim, K.S. Wong, Virtual machine placement with (m, n)-fault tolerance in cloud data center, Cluster Comput. 22 (5) (2019) 11619–11631.
[113] Z. Wen, J. Cała, P. Watson, A. Romanovsky, Cost effective, reliable and secure workflow deployment over federated clouds, IEEE Trans. Serv. Comput. 10 (6) (2017) 929–941.
[114] V. Prakash, A. Bala, A Novel scheduling approach for workflow management in cloud computing, Proceedings of the International Conference on Signal Propagation and Computer Technology, 2014, pp. 610–615.
[115] G. Kaur, M. Kalra, Deadline constrained scheduling of scientific workflows on cloud using hybrid genetic algorithm, Proceedings of the 7th International Conference on Cloud Computing, Data Science and Engineering - Confluence, 2017, pp. 276–280.
[116] B. Kumar, M. Kalra, P. Singh, Discrete binary cat swarm optimization for scheduling workflow applications in cloud systems, Proceedings of the 3rd International Conference on Computational Intelligence and Communication Technology, 2017, pp. 1–6.
[117] A. Alahmadi, D. Che, M. Khaleel, M.M. Zhu, P. Ghodous, An innovative energy-aware cloud task scheduling framework, Proceedings of the IEEE 8th International Conference on Cloud Computing, 2015, pp. 493–500.
[118] K. Bahwaireth, L. Tawalbeh, E. Benkhelifa, Y. Jararweh, M.A. Tawalbeh, Experimental comparison of simulation tools for efficient cloud and mobile cloud computing applications, EURASIP J. Inf. Secur. 15 (2016) 1–14.
[119] N.H. Shahapure, P. Jayarekha, Virtual machine migration based load balancing for resource management and scalability in cloud environment, Int. J. Inf. Technol. (2018) 1–12.
[120] W.A. Higashino, M.A.M. Capretz, L.F. Bittencourt, CEPSim: modelling and simulation of complex event processing systems in cloud environments, Futur. Gener. Comput. Syst. 65 (2015) 122–139.
[121] M. Quwaider, Y. Jararweh, M. Al-Alyyoub, R. Duwairi, Experimental framework for mobile cloud computing system, Procedia Comput. Sci. 52 (2015) 1147–1152.
[122] M. Al-Ayyoub, Y. Jararweh, M. Daraghmeh, Q. Althebyan, Multi-agent based dynamic resource provisioning and monitoring for cloud computing systems infrastructure, Cluster Comput. 18 (2) (2015) 919–932.
[123] Q. Althebyan, Y. Jararweh, Q. Yaseen, O. AlQudah, M. Al-Ayyoub, Evaluating map reduce tasks scheduling algorithms over cloud computing infrastructure, Concurr. Comput. Pract. Exp. 27 (2015) 5686–8699.
[124] M. Quwaider, Y. Jararweh, A cloud supported model for efficient community health awareness, Pervasive Mob. Comput. 28 (2016) 35–50.
[125] Y. Hu, L. Zhao, Z. Liu, H. Ju, H. Shi, P. Xu, Y. Wang, L. Cheng, DisSetSim: an online system for calculating similarity between disease sets, J. Biomed. Semant. 8

(2017) 19–25.
[126] S.H. Lim, B. Sharma, B.C. Tak, C.R. Das, A dynamic energy management scheme for multi-tier data centers, Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software, 2011, pp. 257–266.
[127] A. Almaatouq, A. Alabdulkareem, M. Nou, M. Alsaleh, A. Alarifi, A malicious activity detection system utilizing predictive modeling in complex environments, Proceedings of the IEEE 11th Consumer Communications and Networking Conference, 2014, pp. 371–379.
[128] A. Banerjee, J. Banerjee, G. Varsamopoulos, Z. Abbasi, S.K.S. Gupta, Hybrid simulator for cyber-physical energy systems, Proceedings of the Workshop on Modeling and Simulation of Cyber-Physical Energy Systems, 2013, pp. 1–6.
[129] R. Qi, W. Liu, J. Gutierrez, M. Narang, Sustainable and resilient network infrastructure design for cloud data centers, Proceedings of the Engineering and Management of Data Centers, 2017, pp. 227–259.
[130] H.C.S. Filho, G.F. Carneiro, E.S.M. Costa, M. Monteiro, Tools to support SMEs to migrate to the cloud: opportunities and challenges, Inf. Technol. New Gener. (2018) 159–165.
[131] D. Kliazovich, P. Bouvry, S.U. Khan, DENS: data center energy-efficient network-aware scheduling, Cluster Comput. 16 (1) (2013) 65–75.
[132] X. Li, L. Nie, S. Chen, Approximate dynamic programming based data center resource dynamic scheduling for energy optimization, Proceedings of the IEEE International Conference on Internet of Things (iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom), 2014, pp. 494–501.
[133] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, A.Y. Zomaya, Energy-efficient data replication in cloud computing datacenters, Cluster Comput. 18 (2015) 385–402.
[134] J. Zhihua, Greencloud for simulating QoS based NaaS in cloud computing, Proceedings of the Ninth International Conference on Computational Intelligence and Security, 2013, pp. 766–770.
[135] I.D. Faria, M.A.R. Dantas, M.A.M. Capretz, W.A. Higashino, Energy-aware resource selection model for opportunistic grids, Proceedings of the IEEE 23rd International WETICE Conference, 2014, pp. 167–172.
[136] M.A. Sharkh, A. Shami, P. Ohlen, A. Ouda, A. Kanso, Simulating high availability scenarios in cloud data centers: a closer look, Proceedings of the IEEE 7th International Conference on Cloud Computing Technology and Science, 2015, pp. 617–622.
[137] A. Núñez, G.G. Castañé, J.L. Vázquez-Poletti, A.C. Caminero, J. Carretero, I.M. Llorente, Design of a flexible and scalable hypervisor module for simulating cloud computing environments, Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems, 2011, pp. 265–270.
[138] C. Esposito, M. Ficco, F. Palmieri, A. Castiglione, Smart cloud storage service selection based on fuzzy logic, theory of evidence and game theory, IEEE Trans. Comput. 65 (8) (2015) 2348–2362.
[139] T. Hirofuchi, A. Lebre, L. Pouilloux, Adding a live migration model into SimGrid: one more step toward the simulation of infrastructure-as-a-service concerns, Proceedings of the IEEE 5th International Conference on Cloud Computing Technology and Science, 2013, pp. 96–103.
[140] G.G. Castanea, A. Nunez, P. Llopisa, J. Carreteroa, E-mc2: a formal framework for energy modelling in cloud computing, Simul. Model. Pract. Theory 39 (2013) 56–75.
[141] V. Casola, A.D. Benedictis, M. Rak, U. Villano, A proposal of a cloud-oriented security and performance simulator provided as-a-service, Proceedings of the Conference on Complex, Intelligent, and Software Intensive Systems, 2019, pp. 1002–1011.
[142] G. Kecskemeti, S. Ostermann, R. Prodan, Fostering energy-awareness in simulations behind scientific workflow management systems, Proceedings of the IEEE/ACM 7th International Conference on Utility and Cloud Computing, 2014, pp. 29–38.
[143] K. Plankensteiner, R. Prodan, Meeting soft deadlines in scientific workflows using resubmission impact, IEEE Trans. Parallel Distrib. Syst. 23 (5) (2012) 890–901.
[144] W. Tian, M. Xu, A. Chen, G. Li, X. Wang, Y. Chen, Open-source simulators for Cloud computing: comparative study and challenging issues, Simul. Model. Pract. Theory 58 (2) (2015) 239–254.
[145] S. Sotiriadis, N. Bessis, N. Antonopoulos, Towards Inter-cloud simulation performance analysis: exploring service-oriented benchmarks of clouds in SimIC, Proceedings of the 27th International Conference on Advanced Information Networking and Applications Workshops, 2013, pp. 765–771.
[146] I. Sriram, D. Cliff, Effects of component-subscription network topology on large-scale data centre performance scaling, Proceedings of the 15th IEEE International Conference on Engineering of Complex Computer Systems, 2010, pp. 72–81.
[147] I. Sriram, D. Cliff, Hybrid complex network topologies are preferred for component-subscription in large-scale data-centres, Commun. Comput. Inf. Sci. 116 (2011) 130–137.
[148] D. Fernández-Cerero, A. Jakóbik, D. Grzonka, J. Kołodziej, A. Fernández-Montes, Security supportive energy-aware scheduling and energy policies for cloud environments, J. Parallel Distrib. Comput. 119 (2018) 191–202.
[149] A. Markus, A. Marques, G. Kecskemeti, A. Kertesz, Efficient simulation of IoT cloud use cases, Auton. Control Reliab. Internet Serv. (2018) 313–336.
[150] G. Kecskemeti1, Z. Nemeth, A. Kertesz, R. Ranjan, Cloud workload prediction based on workflow execution time discrepancies, Cluster Comput. 22 (2018) 737–755.
[151] G. Kecskemeti, W. Hajji, F.P. Tso, Modelling low power compute clusters for cloud simulation, Proceedings of the 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing, 2017, pp. 39–45.
[152] V.D. Maio, G. Kecskemeti, R. Prodan, An improved model for live migration in data centre simulators, Proceedings of the 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2016, pp. 527–530.
[153] A.T. Makaratzis, K.M. Giannoutakis, D. Tzovaras, Energy modeling in cloud simulation frameworks, Futur. Gener. Comput. Syst. 79 (2018) 715–725.
[154] X. Zhou, G. Zhang, J. Sun, J. Zhou, T. Wei, S. Hu, Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT, Futur. Gener. Comput. Syst. 93 (2019) 278–289.
[155] E. Deelman, G. Singh, M.H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, B.G. Berriman, J. Good, A. Laity, J.C. Jacob, D.S. Katz, Pegasus: a framework for mapping complex scientific onto distributed systems, Sci. Program 13 (3) (2005) 219–237.