Simulation Modelling Practice and Theory 49 (2014) 180–192

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

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

CloudExp: A comprehensive cloud computing experimental framework

Yaser Jararweh [a,*], Moath Jarrah [b], Mazen kharbutli [b], Zakarea Alshara [a], Mohammed Noraden Alsaleh [c], Mahmoud Al-Ayyoub [a]

[a] Department of Computer Science, Jordan University of Science and Technology, 22110, Jordan
[b] Department of Computer Engineering, Jordan University of Science and Technology, 22110, Jordan
[c] University of North Carolina Charlotte, Charlotte, NC, USA

ARTICLE INFO

Article history:
Received 5 February 2014
Received in revised form 4 August 2014
Accepted 10 September 2014
Available online 15 October 2014

Keywords:
Cloud computing
Cloud computing modeling and simulation
CloudSim
Network topologies
MapReduce
SLA management
Rain workload generator

ABSTRACT

Cloud computing is an emerging and fast-growing computing paradigm that has gained great interest from both industry and academia. Consequently, many researchers are actively involved in cloud computing research projects. One major challenge facing cloud computing researchers is the lack of a comprehensive cloud computing experimental tool to use in their studies. This paper introduces *CloudExp*, a modeling and simulation environment for cloud computing. *CloudExp* can be used to evaluate a wide spectrum of cloud components such as processing elements, data centers, storage, networking, Service Level Agreement (SLA) constraints, web-based applications, Service Oriented Architecture (SOA), virtualization, management and automation, and Business Process Management (BPM). Moreover, *CloudExp* introduces the Rain workload generator which emulates real workloads in cloud environments. Also, MapReduce processing model is integrated in *CloudExp* in order to handle the processing of big data problems.

1. Introduction

Cloud computing is an emerging computing paradigm that is continuously evolving and spreading. Many experts believe it will become the dominant IT service delivery model by the end of the decade [1]. Cloud computing is built on a wide range of different computing technologies such as high-performance computing, grid and utility computing, distributed systems, virtualization, storage, networking, security, management, automation, Service-Oriented Architecture (SOA), etc. Moreover, other concepts like Business Process Management (BPM), Service-Level Agreement (SLA), Quality of Service (QoS). This complexity presents a major challenge for researchers to conduct comprehensive cloud computing-related experiments for two main reasons: First, conducting experiments on real systems is expensive and the criticality and frangibility of the system poses many limitations and risks. Such experiments may adversely affect the system's availability, reliability, and security. Second, while simulation tools would mitigate the problems that arise from using a real system, there are no comprehensive cloud computing experimental tools that cover the wide spectrum of cloud computing components. Unfortunately, current

* Corresponding author.
E-mail addresses: yijararweh@just.edu.jo (Y. Jararweh), mjarrah@just.edu.jo (M. Jarrah), kharbutli@just.edu.jo (M. kharbutli), maalshbool@just.edu.jo (Z. Alshara), malsaleh@uncc.edu (M.N. Alsaleh).

http://dx.doi.org/10.1016/j.simpat.2014.09.003
1569-190X/

cloud computing research tools only cover a subset of the different components listed above, which make them limited and not applicable when considering new cloud computing technologies.

One of the earliest and most popular cloud computing simulators is CloudSim which an environment developed at the University of Melbourne [14]. CloudSim is a very promising tool for conducting cloud computing experiments. However, CloudSim has several limitations and shortcomings: First, it is built on top of a grid computing environment which puts limitations on the infrastructures that can be simulated. This is a major limitation since current cloud systems are built using a wide range of different hardware configurations. Second, it only includes a basic and simplified network model with limited workload traffic generator. Third, it does not provide efficient modeling for embarrassingly parallel data problems. Fourth, it lacks several important cloud computing components such as BPM and SLA. Fifth, it lacks a Graphical User Interface (GUI) which can be useful and easier for researchers to conduct their experiments.

As a result of the lack of a comprehensive cloud computing experimental tool and due to the limitations in CloudSim, we set out to develop *CloudExp* as a comprehensive and effective cloud computing experimental framework. CloudExp capabilities cover most of cloud computing related technologies such as big data management and mobile cloud computing, etc. CloudExp uses CloudSim as the base design platform and introduces many new enhancements and extensions on top of it. These enhancements and extensions include:

- Integrating of the Rain cloud workload generator from Berkeley into CloudSim simulator [16].
- Integrating of a MapReduce framework into CloudSim to handle embarrassingly parallel data processing paradigms and Big Data problems.
- Adding new modules related to SLA and BPM.
- Adding new cloud computing network models (VL2, BCube, Portland, and DCell) that represent actual topologies in real cloud environments.
- Introducing of the Mobile Cloud Computing (MCC) simulation framework.
- Introducing a monitoring outlet for most of the cloud system components.
- Adding an action module to allow researchers to reconfigure a cloud system and study the overall system behavior and performance.
- Adding a GUI for the simulator to make it more user-friendly.

This paper presents CloudExp and discusses its different components in detail. The rest of the paper is organized as follows: Section 2 provides a comprehensive survey in the field of cloud computing simulation and modeling. CloudSim, which is the base for CloudExp, is introduced in sub-Section 2.1. CloudExp is fully detailed in Section 3. A use case scenario for CloudExp is presented in Section 4 where different simulation results are presented. Finally, Section 5 concludes the paper.

2. Related work

Simulations tools are essential for carrying out research experiments in cloud computing. Several cloud computing simulators have been developed, each of which supports certain aspects/components of cloud computing. This section describes some of the most popular simulators in the fields of cloud computing and distributed systems.

2.1. CloudSim: The underlying framework for CloudExp

CloudSim is a cloud computing modeling and simulation tool that was developed at the University of Melbourne [14]. It aims to provide cloud computing researchers with an experimental tool to conduct cloud computing-related research. It supports the modeling and simulation of various cloud computing components, including power management, performance, data centers, computing nodes, resource provisioning, and virtual machines provisioning [12].

CloudSim is a layered design framework written in Java and was initially built on top of SimJava and GridSim [4]. Both SimJava and GridSim are discrete event simulators that are widely used to simulate parallel and distributed computing systems. However, SimJava and GridSim have several scalability limitations that led the developers of CloudSim to implement a new discrete event management framework which is the CloudSim core simulation engine [14]. The following is a brief description of the different cloud computing features and components modeled by CloudSim.

The *Data centers* represent the core infrastructure in a cloud system and include the hardware and software stack. In defining the data centers, the modeler needs to identify the number of hosts in each data center. The *Hosts* are modeled using the class *Host* which models a physical resource such as a computer or a server. A data center can have multiple hosts. *Cloud tasks* are represented by *Cloudlets* which are the cloud-based application services. In this class, computational metrics are used to model the complexity of applications. *Brokering* is performed through the use of the *DatacenterBroker* class. *Virtual machines (VM)* are modeled in the class *VirtualMachine*. *Coordination* is performed using the *CloudCoordinator* class which models the communication between different cloud coordinators and cloud brokers. In addition, it monitors the internal state of each data center that is connected to it. *Bandwidth provisioning services* are modeled using the *BWProvisioner* class. *Memory provisioning* and the policies for allocating memory to VMs are modeled using the *MemoryProvisioner* class. *Virtual machine provisioning* is performed using the *VMProvisioner* class which allocates VMs to hosts.*VM allocation policies* models

the policies for allocating processing power to VMs. Allocation can be space-shared or time-shared. *Power consumption* is modeled in the *PowerModel* class which models data centers power consumption. Due to the space limit, the readers are directed to the work presented in [14] for more details about CloudSim components.

2.2. iCanCloud

iCanCloud [18] is a cloud computing simulation platform capable of conducting large-scale experiments. It provides a scalable, flexible, fast, and easy to use tool that allows organizations to optimize the trade-offs between the cost and performance of their cloud computing system. The building block of a cloud computing system in iCanCloud simulator is the virtual machine. A virtual machine can be built by selecting and configuring its four main components: CPU, memory, storage, and network. Furthermore, iCanCloud provides a graphical interface that helps users in configuring their cloud computing experiments. The configuration of a cloud computing system in iCanCloud consists of three main parts: (1) the cloud environment definition, (2) the cloud hypervisor definition, and (3) the users configuration. The cloud environment definition is specified using the NED language and defines the names of virtual machine instances to be simulated in the model and the number of machines for each instance. The characteristics of the CPU, memory, storage, and network for each virtual machine are determined by users as well. The CPU is characterized by the processor speed measured in Million Instructions Per Second (MIPS) and the scheduler (iCanCloud provides a number of defined strategies for the CPU scheduler). The memory system configuration defines the size of the memory space and the access latency. The storage system in iCanCloud is modeled as file system, volume manager and disk drivers. The data requests are translated through these three components into data blocks and then read or written to the appropriate disk drive. Applications running on different virtual machines in the simulated cloud computing model can communicate and interchange data through a defined network. iCanCloud provides a networking framework that simulates different network protocols (e.g. TCP or UDP) with a high degree of accuracy. It also provides a set of modules reflecting real network devices such as routers, switches and other network protocols which gives the user the flexibility to simulate different wired or wireless network architectures.

The second part of iCanCloud configuration is the cloud hypervisor definition which specifies the hypervisor model to be used in the simulation. The hypervisor module is responsible for managing the virtual machines and executing the jobs based on customized brokering policies. iCanCloud provides an interface for customizing the hypervisor module by integrating new brokering policies that determine how to select the jobs to be executed and on which virtual machine in the cloud. The last part of the configuration is the users configuration. The users are modeled as entities that submit jobs to be executed by the existing virtual machines in the cloud. iCanCloud is provided with a list of users who are currently submitting jobs to the computing system. For each user we have a list of jobs and the configuration of virtual machines that can execute each job. The responsibility of selecting the appropriate virtual machine to execute a job can be delegated to the hypervisor since most users do not have such expertise. The datacenters in iCanCloud can be represented by a set of virtual machines. However, iCanCloud does not currently provide models for power consumption evaluation.

CloudAnalyst [19] is a simulator that enables the user to model software as a service data centers in different geographical locations. The output of the experiments in CloudAnalyst consists of the response time for each request and the cost to accomplish the requested task. The cost is based on EC2 cost policy from Amazon. GreenCloud simulator [20] is an extension of the popular network simulator NS-2. GreenCloud focuses on simulating the communication between processes running in a cloud computing infrastructure. It simulates the packets exchanged and provides power consumption analysis of data center components (e.g. server and switches).

Table 1 shows a comparison between CloudExp and the two major simulators in the field: CloudSim and iCanCloud. The table shows that CloudExp covers a much wider spectrum of features compared to the other simulators making it more effective and comprehensive for cloud computing research. Its worth to mention here that while CloudSim is already supporting some of the functionalities like the power consumption calculation, CloudExp extend this functionality by using a better calculation model that take in consideration power consumption for all the components of the data center including the network power consumption.

Table 1
Comparison of cloud simulators.

Metric	CloudSim	iCanCloud	CloudExp
GUI support	No	Yes	Yes
Dynamic workload	No	No	Yes
Networking models	Limited	Moderate	Full
Mobile cloud computing simulation	No	No	Yes
Power consumption calculations	Yes	No	Yes
Service level agreement	Yes	No	Yes
MapReduce support	No	No	Yes

3. CloudExp framework

The features that CloudSim provides are promising for cloud computing research. However, it lacks several components needed for the simulation of comprehensive cloud computing experiments. CloudExp integrates several new features and important components to CloudSim. This makes it a more comprehensive simulation environment for current cloud computing systems. The added components include a more-detailed workload generator Section 3.1, modules for MapReduce Section 3.2, modules for mobile cloud computing Section 3.3, extensions for SLA and BPM Section 3.4, a wide range of network topologies Section 3.5, and a Graphical User Interface (GUI). The new extensions and additions are explained in details in this section.

A GUI extension is provided in order to make it convenient for users to conduct research experiments by creating the main components in a cloud system. For example, Fig. 1 shows how a user can create data centers and define their parameters such as the number of hosts on each data center. Also, its shows the creation of virtual machines and brokers in CloudExp.

3.1. Workload generation and the rain workload generator

Cloud computing systems promise to handle and fulfill different customers' applications, requested services and workload requirements. These applications and services vary widely ranging from data intensive (i.e. big-data) applications to financial applications and web services. Moreover, cloud workloads differ in their size, execution times, memory footprints, network BW requirements, frequency, and QoS requirements, among other factors. Such diverse workloads are very dynamic and exhibit different patterns of execution phases during an application's life span. Hence, cloud computing systems are required to dynamically adapt their operational environment on the fly to cope with these workload dynamics. Workloads characterization is very crucial in a cloud computing system in order to achieve proper resources provisioning, cost estimation, and SLA considerations. Cloud computing application developers and hosting companies need to predict changes in the workload patterns to be able to adjust promptly when variations in the patterns occur. Moreover, a detection of any possible system bottlenecks is needed before actually porting the application to the real system. Cloud application updates after deployment may also cause problems in the actual cloud system because they might change the workload patterns and dynamics. These changes affect the cloud system resources allocation and QoS measurements. As a result, a mechanism to test cloud applications before their actual deployment is of the utmost necessity.

Based on the aforementioned facts, an accurate and comprehensive workload modeling environment is essential in any cloud computing simulation tool. Current workload modeling environments for systems such as grid computing do not fit or match the characteristics of real workloads in cloud computing environments. CloudSim provides a basic cloud workload modeling structure that fails to capture many of the characteristics of current cloud computing workloads. Hence, CloudSim users need to extend the available modules to generate the required patterns. CloudSim models the workload as an

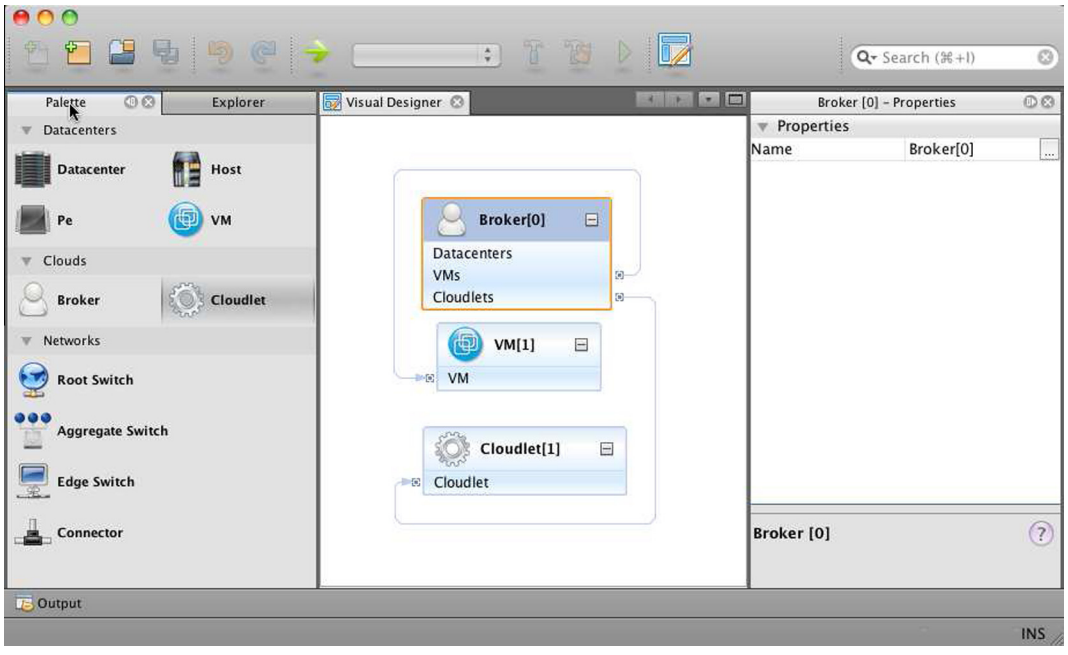

Fig. 1. Creating a data centers with virtual machines and cloud brokers in CloudExp.

extension to the Cloudlet entity and introduces the utilization model which is an abstract class that needs to be extended by the users to implement the workload patterns related to the application[14]. The Utilization Model provides methods and variables in order to specify the resources requirements of the applications during the deployment phase. The input to this method is a discrete time parameter, while the output is a percentage of the resources that are required by the Cloudlet. The main drawback of this approach stems from the fact that the user manually needs to implement an accurate model for the application's workload patterns. This is not an easy task and incur an extra overhead on the user.

In order to solve this problem, CloudExp is integrated with Rain workload generator framework from the University of California at Berkeley [16] which is an open-source software. The Rain framework presents an accurate and comprehensive cloud computing workload characteristics. Hence, CloudExp replaces the basic workload generator framework available in CloudSim by integrating the Rain workload generator. The integration process is simplified by the fact that both CloudSim and Rain were written in Java. The three main workload characteristics that Rain provides are: (1) Variations in workload intensity (e.g. small, medium, and large). (2) Variations in the mix of operations that are performed by the system components (e.g. data intensive, computing intensive, and task dependencies). (3) Variations in the data access patterns and frequency, also known as data hot spots. Predicting data hot spots enables efficient access to them which reduces access time and overhead. next we address the architecture of the Rain workload generator.

3.1.1. Rain workload generator architecture

Rain is a unique statistics-based workload generator. It provides the user with the ability to easily reuse, configure, and schedule cloud applications workloads. The variations in the requirements of the applications in Rain are achieved using empirical probability distributions. Rain architecture handles the variations using load scheduling mechanisms. The changes in the mix of operations, the access patterns, and data hot spots are done using application-specific request generators [16]. Moreover, Rain architecture differs from other workload generators in handling workload requests. While current generators couple request generation with its execution, Rain decouples request generation from request execution which adds flexibility how to manage requests. Another limitation of current generators is related to the fact that the thread that generates the request is also responsible for its execution; this is known as thread-affinity problem. Thread-affinity problem impacts the performance of the generator as one thread cannot generate new requests until it completes the current one. Rain solves this problem by allowing any request generated by a certain thread to be executed by any available thread and not only by the one who generates it. This solution provides flexibility, scalability and improves the performance.

The Rain architecture consists of the following components:

(1) The *Scenario* component is responsible for the experiment configuration parameters (e.g. maximum number of users, experiment duration, etc.).
(2) The *Generator* component uses the configuration from the *Scenario* component to create requests and operations to be used by other components.
(3) The *Scoreboard* component is used to present experiment results and summary.
(4) The *Threading* component contains the available threads for executing the tasks produced by the request generator component.
(5) The *Benchmark* component is responsible for running the entire experiment. It interacts with other components by loading the *Scenario* that needs to be handled by the threading components. It also initializes the *Scoreboard* and presents the results at the end of the experiment.

3.1.2. Rain workload generation scenario

A new workload will be initialized by using the *Scenario* where each entity, i.e. user, will be assigned to a certain generator and a thread. The thread requests a new operation from the generator. When the thread finishes executing the operations assigned by the generator, it generates a summary (e.g. status, execution results, etc.) and writes its details on the *Scoreboard*. All these steps are controlled by the *Benchmark* component. CloudExp users can use the workload generator in different ways. First, users can experiment with different workload patterns and mix of operations while using a fixed system configuration (e.g. VMs, network, etc.). This can help in understanding how a workload can impact the utilization of a cloud computing components and the SLA. Second, users can conduct experiments with different provisioning algorithms and assess how these algorithms react to workload variations. Third, users are able to study the impact of different workload patterns and characteristics. Fourth, users can apply an intensive workload scenario to detect bottlenecks in a cloud infrastructure.

3.2. MapReduce model

MapReduce is widely used as a powerful parallel data processing model [5]. It is a programming model that has efficiently solved problems of large datasets using large clusters of machines. Examples of applications that utilize MapReduce include distributed grep, distributed sorting, web-link graph reversal, web-access log stats analysis, document clustering, and machine learning [7]. Cloud computing offers a suitable environment for processing and analyzing terabytes of data through the utilization of many resources connected through a network topology [9]. Most cloud providers such as Amazon EC2, Microsoft Azure, Google, Yahoo and Facebook, adopted MapReduce in their computing environments.

Due to the fact that it is difficult to perform benchmarking experiments for MapReduce in real infrastructures, CloudExp provides a simulation solution to overcome that. Users can use CloudExp toolkit to perform experiments under non-static conditions (e.g. availability and workload pattern) in a controllable environment where tests can be re-executed and re-configured.

3.2.1. Design and implementation of MapReduce

This subsection provides finer details about the fundamental classes of MapReduce. The class design diagram for MapReduce and its correlation with CloudSim is shown in Fig. 2. The list of classes are:

1. Master: This class contains instances of all mappers and reducers that belong to the same user. Also, it contains information about the status and data locations during the execution of the mappers and reducers.
2. Map: It models the mappers' information and behavior such as the input dataset size, output dataset size, and utilization model.
3. Reduce: It models the reducers' information and behavior such as the input dataset size, output dataset size, and utilization model.
4. MapReduceSchedulerSpaceShared: This class extends an abstract class CloudletScheduler. It represents the behavior policy between the mappers and reducers. For example, it determines the share policy of processing power among mappers and reducers in a virtual machine.

In the MapReduce approach, the input data is partitioned into a set of sub-problems which in turn can be partitioned further into smaller sub-problems. This allows the computing grid or cluster to process small size problems in parallel. The output is constructed by collecting all the results back from the sub-problems in a fashion that yields a correct result for the initial problem. The illustration in Fig. 3 shows the flow of a MapReduce operation [5]. When a user program simulates MapReduce tasks, the following sequence of actions occurs:

1. The workload in the user experiment is read and parsed to initiate the list of map and reduce instances. Each instance has parameters that are used to simulate the MapReduce model. The parameters include: ID, input data size, output data size, and utilization models.
2. The master node assigns the instances created in the previous step to computing nodes in the cluster. The master also keeps track of mappers and reducers information such as datacenter id, host id, virtual machine id, and a status (created, ready, waiting, running, success, fail, canceled). Then each mapper or reducer is submitted to the cloud computing environment using the predefined utilization models with an idle state called ready state.
3. After the submission of the mappers (initially in ready state), the simulation of MapReduce starts while taking into account each mapper's status and the utilization model.
4. When a mapper finishes processing, it stores the dataset results in a specific location and informs the master of the address.
5. When all mappers finish processing, the master sends signals to all reducers to start working on the datasets that were produced.
6. During MapReduce simulation, CloudExp toolkit continuously collects data to be displayed for statistical analysis after the simulation is over.

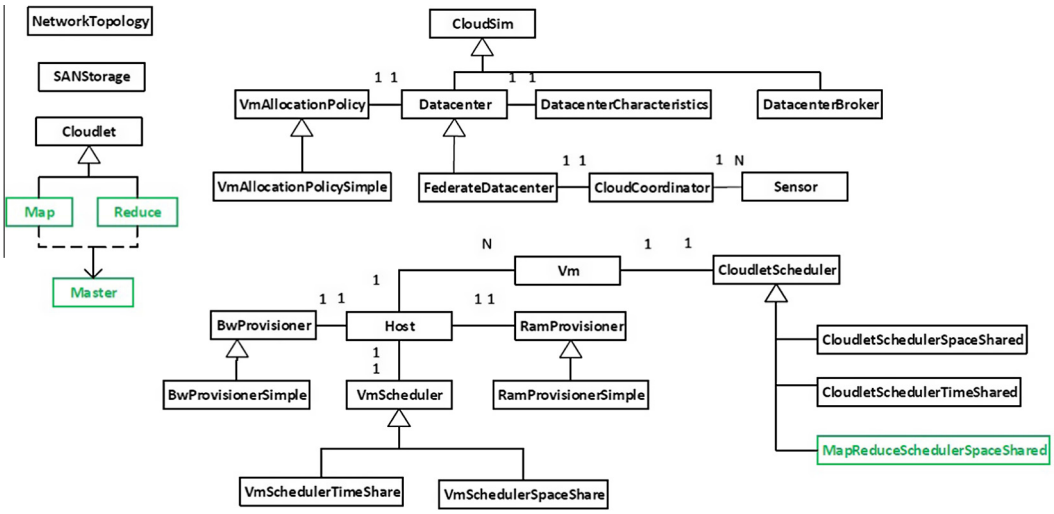

Fig. 2. MapReduce class design diagram.

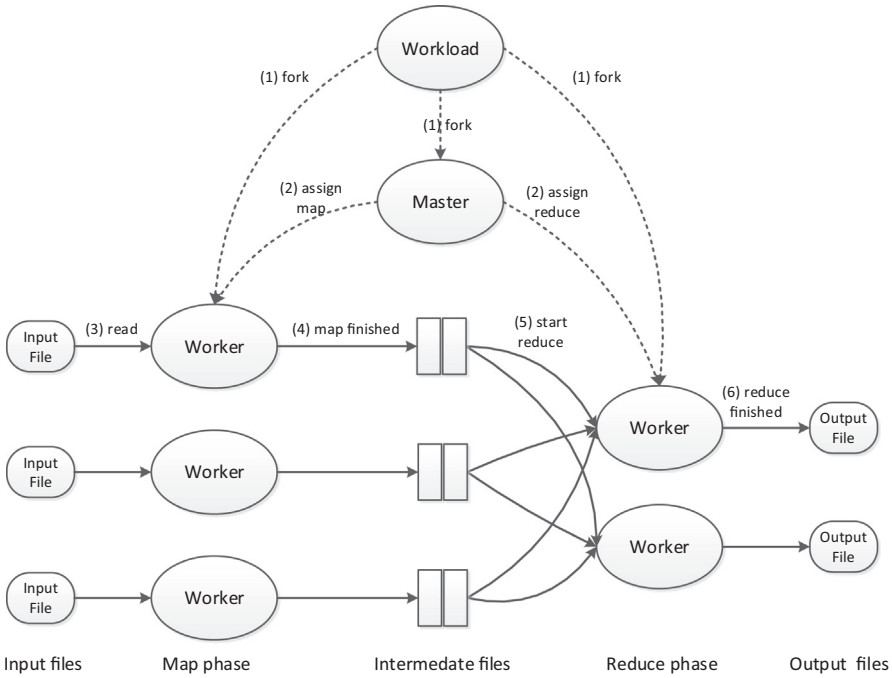

Fig. 3. Flow of MapReduce operation.

As shown in Table 5, CloudExp provides many MapReduce configuration parameters such as the scheduling algorithms like FIFO, Matchmaking and Delay algorithm. Number of Mappers and Reducers, the input and the output files size, number of files per task and number of CPUs per task. A complete example for using CloudExp to simulate MapReduce workload is shown in Section 4.3.

3.3. Mobile cloud computing

Nowadays, smartphones and tablet PCs are more widespread and users are relying more on them compared to conventional personal computers. This has lead to an unprecedented growing market for mobile devices. However, mobile devices suffer from many limitations related to their limited resources. Limitations include: connectivity issues, limited bandwidth, security vulnerabilities, applications compatibility, and a restricted power since they are mostly powered by batteries.

An emerging concept of Mobile Cloud Computing (MCC) is showing a great momentum nowadays within mobile service providers and users. It is based on the integration of cloud computing providers and the mobile service system. This integration is achieved by the possibility of offloading mobile data and intensive computation requirements to cloud computing infrastructures [21]. This enables the mobile device to migrate the intensive computation tasks to cloud infrastructures and preserve its limited resources. In addition, mobile devices would no longer require large storage capacity on the device as the user data could be stored in a cloud storage system. The user can easily retrieve his/her data whenever is needed. MCC research is expanding quickly and many researchers are engaged in MCC research topics such as data offloading, mobile-cloud subscription, and security. To the best of our knowledge, there are no simulation tools that consider MCC simulation so far. On CloudExp, we present a new simulation module that handles MCC simulation. The MCC module in CloudExp provides the basic capabilities for users to conduct MCC related experiments. Using CloudExp, the researcher can configure the MCC module to create and simulate MCC- related experiments. Many parameters can be used to configure the MCC related experiments. These parameters are ranging from number of mobile users, connection type, connection bandwidth, data rate, and authentication mechanism. The authors in [22,23] exploit CloudExp to successfully simulate MCC related experiments.

3.4. Service level agreement and business process management

Service Level Agreement (SLA) and Business Process Management (BPM) are considered among the basic pillars of today's cloud computing environments [11]. In CloudExp we made efforts to introduce SLA to the users in a simple and effective manner. Starting from the basic definition of SLA as a contract between the service provider and the service requester. CloudExp provides a set of measurable terms that the user can change while studying their impact on other system components.

These measurable terms include, but are not limited to: number of users who can use the system simultaneously (e.g. 1000 active user), service availability (e.g. 99.9%), service cost, service outage handling terms, business continuity and disaster recovery, network performance, security measures, service metering tools available to the user, user compensation in case of SLA violation, and customer support. To make things more realistic, CloudExp connects the SLA terms related to an application's sensitivity to the availability of the service. This is because some applications require an availability of not less than 100%. CloudExp enables the users to study the effect of SLA terms on the different cloud computing components. In addition, it provides the service cost with different SLA configurations and different QoS requirements. Furthermore, CloudExp integrates the Web Service Level Agreement (WSLA) framework that handles the SLA monitoring and enforcement [13]. SLA parameters used in Cloudexp are number of users, service availability, service cost, network performance and SLA violation.

3.5. Network topologies in CloudExp

Data centers in a cloud computing infrastructure normally contain thousands of physical servers that are connected through switches, routers and other network devices. Hence, the proper design of the data center network plays a critical role in the cloud environment because it directly affects the performance and the throughput of cloud applications. VMFlow [17], for example, is a framework for placement and migration of VMs to reduce the cost. It is shown in the paper that the network topology along with network traffic demands should be taken into consideration in order to meet the objective of network power reduction. Conventional data centers are organized in a hierarchy of three layers: core, aggregation and servers. The core layer consists of the core routers that connect the aggregation switches to the Internet. In the servers layer, the servers are organized in racks and connected to access switches (normally called Top-of-Rack switches) which are in turn connected to the aggregation switches. The aggregation layer may provide some functions such as domain service, location service, server load balancing, and more [1]. This simple design suffers from some limitations as stated in [2]. First, while going up in the hierarchy, it becomes very hard technically and financially to sustain high bandwidth which results in preventing idle servers from being assigned to overloaded servers. This affects the data center's performance. Second, the protocols that are normally used in the core and aggregation layers limit the utilization of the links to less than 50% of the maximum utilization capacity. Third, the performance of communication depends on the distance and the addressing scheme of the hierarchy which causes the services to scale up to nearby servers for rapid response time and performance. This requires some unused capacity to be reserved for a single service for future scale up and not to be shared with others.

CloudSim provides a module for network topology which is the NetworkCloudSim. It was developed to simulate the behavior of parallel and distributed applications in cloud computing environments [6]. NetworkCloudSim offers the basic entities and classes to simulate a network through the connection of Switchs (RootSwitch, AggregateSwitch and EdgeSwitch), NetworkDatacenter and NetworkDatacenterBroker.

However, NetworkCloudSim does not support the common network topologies in cloud computing such as VL2 [2], BCube [10], Portland [15], and DCell [3]. Hence, these topologies were integrated into our CloudExp tool [8]. Also, CloudExp supports a graphical interface to easily drag and drop entities, define their properties, and establish the connections between them which results in a customized network topology.

4. Simulation and experiments scenarios using CloudExp

CloudExp offers a wide spectrum of functionalities and capabilities to its users, which would help them in understanding the different aspects of cloud computing environments. Users can simulate experiments by specifying some infrastructure configuration. To demonstrate some of the functionalities of CloudExp, we will use two different set of experiments as a showcase examples and present a the results obtained using CloudExp. The first set of experiments is showing cloud data center resources utilization and power consumption with different work load scenario and SLA constrains. The second set of experiments is presenting Map Reduce task scheduling experiments using CloudExp. The workload characteristics used in both cases are described in the next section.

4.1. Workload characteristics

One of the main contributions of CloudExp is the integration of the Rain workload generator into the CloudSim system. Rain workload generator demonstrates its near optimality in correctly characterizing cloud like workloads [16]. All the experimental results presented in the next two sections are based on the Rain workload generator.

The workload used in the experiments represents a social-event calendar web application named *Olio* with many operations like adding events, viewing event details, attending events, adding new users, viewing user profile details, searching [16]. The workload will be initialized by using a *Scenario* where each entity, i.e. user or a task, will be assigned to a certain generator and a thread. The thread requests a new operation from the generator. This operation will be executed in the assigned VM. When the thread finishes executing the operations assigned by the generator, it generates a summary (e.g. status, execution results, etc.) and writes its details on the *Scoreboard*. The number of processes executing in each VM is based on the VM utilization and may vary with time. The experiments are lasted for 24 h with a space-shared VM allocation policy based on the resource utilization.

4.2. General CloudExp experiments

The first scenario assumes that a user with $ID = 0$ has a cloud environment that consists of one data center (*datacenter*1) with the parameters listed in Table 2. Moreover, the data center has ten identical physical nodes (*hosts*) each has the properties listed in Table 3. Up to five virtual machines (VM) are executing on each host which results in fifty VMs for the whole data center. In addition, in this scenario, VL2 network topology is used with the specifications listed in Table 4.

Fig. 4 shows the utilization of different components in the data center (*datacenter*1) along with its power consumption as a result of a continuous and steady increase in the workload intensity over time. The workload intensity is represented by the number of VMs running on the data center. As expected, the figure shows that the utilization of RAM, CPU, and bandwidth increase as the workload intensity (number of VMs) increases. The power consumption also increases.

Fig. 5 shows the utilization of different components the data center (*datacenter*1) along with its power consumption in the case of mixed workload with dynamically-changing intensity over time. As expected, the figure shows that the system utilization and power consumption increases/decreases over time based on the workload intensity.

Fig. 6 shows the data center's utilization and power consumption with an increasing workload intensity and a predefined SLA. The example assumes that the system starts violating the SLA when the utilization of system components reaches a certain threshold (e.g. RAM utilization reaches 87% and above). The user can monitor and predict possible SLA violations and can, consequently, increase the physical resources (nodes) to guarantee that the system is always running in a balanced behavior. The figure demonstrates how an increase in the number of nodes results in a drop in the system utilization to within the SLA constraints.

4.3. MapReduce experiments

In our second scenario, we will use the same cloud configuration for the data center (*datacenter*1), the hosts and the network presented in Tables 2–4, respectively, to demonstrate CloudExp's capabilities of simulating MapReduce related experiments with a large number of tasks. Table 5 shows the MapReduce task properties and the experiment configurations. The user can customize his experiments using the MapReduce configuration file which contains the parameters of Table 5. In this

Table 2
Data center configuration.

ISA	x86
Operating system	Linux
Virtual Machine Monitor (VMM)	Xen
Time zone	current time
Cost per processing second	0.03 cents
Cost per memory unit	0.05 cents
Cost per storage unit	0.001 cents
Cost per bandwidth unit	0.01 cents
Virtual machine allocation policy	Allocate VM to the host with lowest utilization

Table 3
Host properties.

Host ID	A unique ID automatically generated in the range 0–3
Storage capacity	1 TB
Number of CPUs	2
Cores per CPU	4
MIPS for each core	1024
Memory capacity	4 GB
Bandwidth	10 Mbps
Virtual machine scheduler	Space shared

Table 4
VL2 network topology structure.

Switching levels	3 levels
Types of switches	One root switch, two aggregate switches, two edge switches
Delays	Root switch = 0.00285 ms; aggregate switch = 0.00245 ms; edge switch = 0.00157 ms
Bandwidth	Root switch = 100 Mb; aggregate switch = 100 Mb; edge switch = 40 Mb

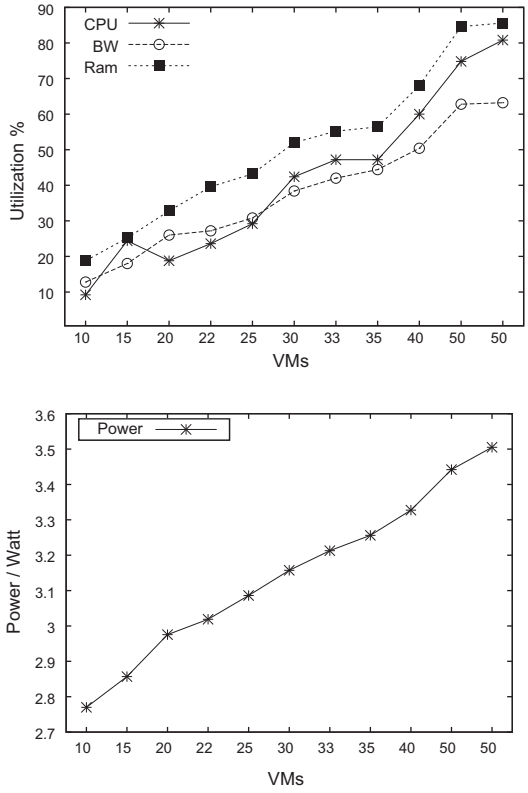

Fig. 4. Datacenter1 utilization and power consumption with a continuous increase in the workload intensity.

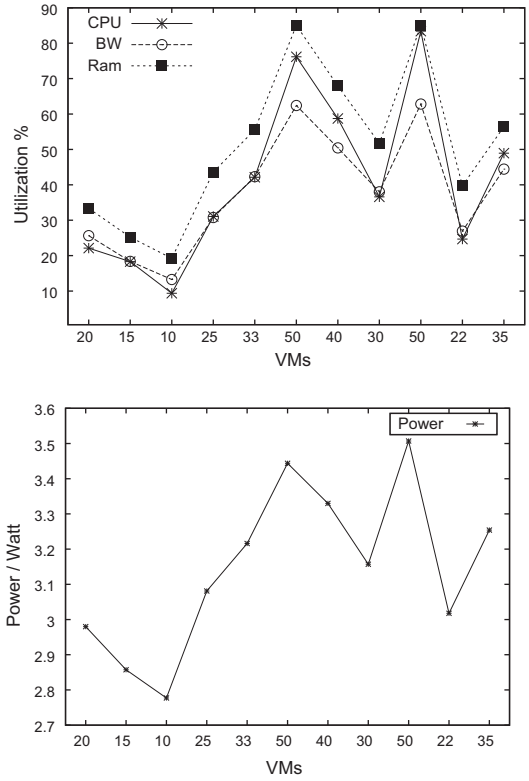

Fig. 5. Datacenter1 utilization and power consumption with a dynamic workload intensity mix.

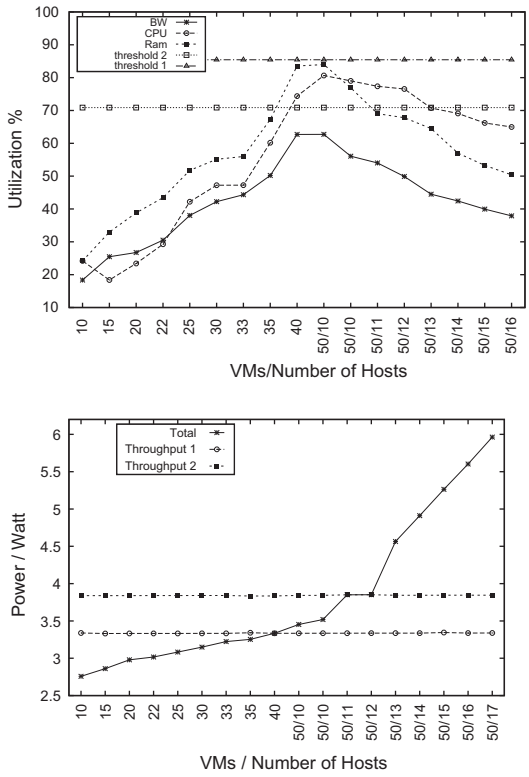

Fig. 6. Datacenter1 utilization and power consumption with dynamic resources configuration and SLA constraints.

Table 5
MapReduce task properties.

Task ID	ID (0-number of tasks)
Scheduling algorithm	Matchmaking
Task length	Random between (1–500) MI
Input file size	64 MB
Output file size	64 MB
Number of files per task	1
Number of mappers	20 CPUs with 4 cores each
Number of reducers	4 CPUs with 4 cores each
Number of VMs per node	Up to 8 VMs per node

experiment, we simulated different numbers of tasks (*Olio* search operations), e.g. 10000, 30000, 50000 and 100000, using the Matchmaking scheduling algorithm with a random task length measured in Million Instructions (MI) unit. Both input and output files' sizes are set to 64 MB. All the computing capacity for (*datacenter*1) is used in this experiments with a total of 20 CPUs (i.e. 80 cores), 4 MB of the total memory and a maximum of 8 VMs per physical node. The number of Mappers is the default value which is 20 CPUs in our experiments representing the total computing capacity available in (*datacenter*1). The user can define any other number of Mappers. The number of Reducers is set to be 4; the default number of reducer is 1. In our experiments the MapReduce workload is represented as a fixed number of *Olio* search tasks with the specifications available in Table 5. These parameters are user defined parameters and can be changed based on the user needs. The output of this experiment include the total simulation time, which represent the total execution time of each set of tasks, and the total power consumptions with different number of VMs.

Fig. 7 shows the total execution time (in seconds) and the total power consumption (in KWH) for different MapReduce workload sizes ranging from 10000 tasks to 100000 tasks using the same physical nodes with different numbers of VMs per physical node. With increasing of the workload size, more execution time and power are needed. On the other hand, using more VMs per physical node will reduce the total execution time and power consumption as we are processing more tasks in parallel without using more resources.

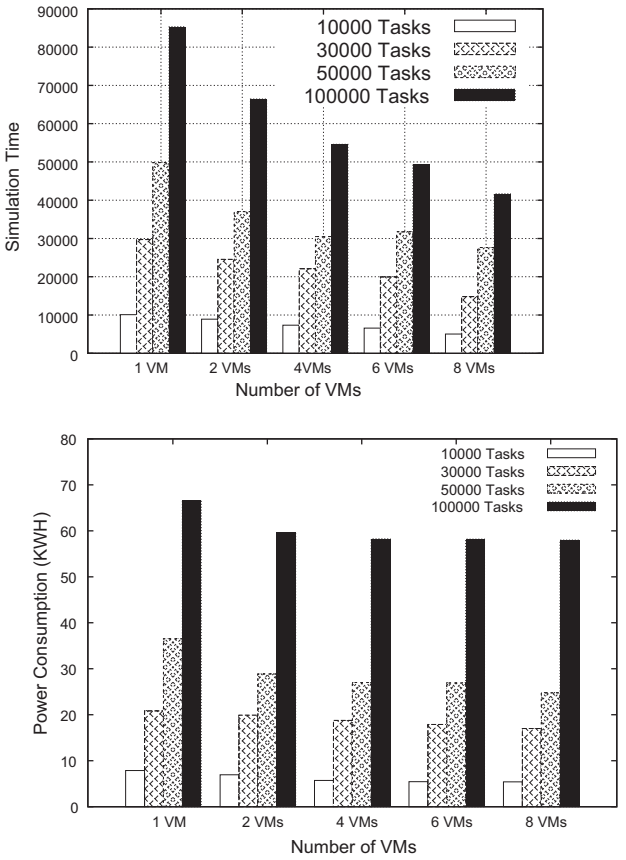

Fig. 7. MapReduce tasks simulation time and power consumption.

5. Conclusion and future work

This paper presented CloudExp, a comprehensive, easy-to-use, and efficient cloud computing modeling and simulation toolkit. CloudExp fills a large gap in cloud computing research caused by the lack of having a comprehensive and easy-to-use tool, in addition to the limited availability and the high costs of conducting experiments in real cloud computing environments. CloudExp provides a rich, yet simple, GUI to build cloud infrastructure and present results and charts to the research to enable configuration analysis and evaluation. CloudExp allows users to customize all aspects in a cloud infrastructure from the host processing nodes to the network topology. In addition, CloudExp provides models for popular and state-of-the-art technologies such as MapReduce and Mobile Cloud Computing. CloudExp allows users to investigate the SLA and other business aspects into the tool. Furthermore, it includes an extensive workload generator capable of accurately representing real cloud infrastructure workload. Moreover, the modularity in CloudExp's design allows users to integrate new components or extend existing ones easily and effectively.

CloudExp makes it easy for users to comprehend the different cloud system components and their roles in the whole system. Users can modify the different components and their parameters, run simulations, and analyze results.

References

[1] Q. Zhang, L. Cheng, R. Boutaba, Cloud computing: state-of-the-art and research challenges, J. Internet Services Appl. 1 (1) (2010) 7–18.
[2] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel, S. Sengupta, VL2: a scalable and flexible data center network, SIGCOMM Comput. Commun. Rev. 39 (4) (2009) 51–62.
[3] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, S. Lu, Dcell: a scalable and fault-tolerant network structure for data centers, in: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, Seattle, WA, USA, 2008, pp.75–86.
[4] R. Buyya, R. Ranjan, R.N. Calheiros, Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities, in: Proceedings of the International Conference on High Performance Computing and Simulation (HPCS), 2009, Leipzig, Germany, 2009, pp.1–11.
[5] J. Dean, S. Ghemawat, Mapreduce: simplified data processing on large cluster, in: Proceedings of the 6th Conference on Symposium on Operating Systems Design and Implementation, 2004, USENIX Association, Berkeley, CA, USA, 2004.
[6] S.K. Garg, R. Buyya, NetworkCloudSim: modelling parallel applications in cloud simulations, in: Proceedings of the 4th IEEE International Conference on Utility and Cloud Computing (UCC 2011), Melbourne, Australia, 2011.

[7] Z. Xiao, Y. Xiao, Accountable MapReduce in cloud computing, in: Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2011, pp. 1082–1087.
[8] Y. Jararweh, Z. Alshara, M. Jarrah, M. Kharbutli, M.N. Alsaleh, TeachCloud: Cloud Computing Educational Toolkit, in: Proceedings of the 1st International IBM Cloud Academy Conference (ICA CON 2012), North Carolina, USA, April 2012.
[9] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, et al. Above the clouds: a Berkeley view of cloud computing, EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28, 2009.
[10] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, S. Lu, BCube: a high performance, server-centric network architecture for modular data centers, in: Proceedings of the ACM SIGCOMM 2009 Conference on Data communication, Barcelona, Spain, 2009, pp. 63–74.
[11] F. Faniyi, R. Bahsoon, Engineering proprioception in SLA management for cloud architectures, in: Proceedings of the 9th Working IEEE/IFIP Conference on Software Architecture (WICSA), Boulder, CO, USA, 2011, pp. 336–340.
[12] K.M. Kim, A. Beloglazov, R. Buyya, Power-aware provisioning of cloud resources for real-time services, in: Proceedings of the 7th International Workshop on Middleware for Grids, Clouds and e-Science, Urbana Champaign, Illinois, USA, 2009.
[13] P. Patel, A. Ranabahu, A. Sheth, Service level agreement in cloud computing, in: Cloud Workshops at OOPSLA09, 2009.
[14] 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 algorithms, Software: Pract. Exp. 41 (1) (2011) 23–50.
[15] R.N. Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, A. Vahdat, PortLand: a scalable fault-tolerant layer 2 data center network fabric, SIGCOMM Comput. Commun. Rev. 39 (4) (2009) 39–50.
[16] A. Beitch, B. Liu, T. Yung, R. Griffith, A. Fox, D.A. Patterson, Rain: A Workload Generation Toolkit for Cloud Computing Applications, Technical Report UCB/EECS-2010-14, 2010.
[17] V. Mann, A. Kumar, P. Dutta, S. Kalyanaraman, VMFlow: leveraging VM mobility to reduce network power costs in data centers, in: Proceedings of the 10th International IFIP TC 6 Conference on Networking, Valencia, Spain, 2011.
[18] Alberto Nez, Jose L. Vzquez-Poletti, Agustin C. Caminero, Gabriel G. Casta, Jesus Carretero, Ignacio M. Llorente, iCanCloud: a flexible and scalable cloud infrastructure simulator, J. Grid Comput. 10 (1) (2012) 185–209.
[19] Wickremasinghe Bhathiya, N. Rodrigo, Rajkumar Buyya Calheiros, CloudAnalyst: a CloudSim-based visual modeller for analysing cloud computing environments and applications, in: Proceedings of the 24th International Conference on Advanced Information Networking and Applications (AINA 2010), Perth, Australia, April 20–23, 2010.
[20] Kliazovich Dzmitry, Bouvry Pascal, Samee Ullah Khan, GreenCloud: a packet-level simulator of energy-aware cloud computing data centers, in: 2010 IEEE Global Telecommunications Conference (GLOBECOM 2010).
[21] Niroshinie Fernando, Seng W. Loke, Wenny Rahayu, Mobile cloud computing: a survey, Future Gener. Comput. Syst. 29 (1) (2013) 84–106. ISSN: 0167-739X, doi:10.1016/j.future.2012.05.023.
[22] Muhannad Quwaider, Yaser Jararweh, Cloudlet-based for big data collection in body area networks, in: Proceedings of the International Conference for Internet Technology and Secured Transactions (ICITST), London, UK, December 2013.
[23] Muhannad Quwaider, Yaser Jararweh, Cloudlet-based efficient data collection in wireless body area networks, Simulat. Modell. Pract. Theory 50 (2015) 57–71.