A Holistic Model for Resource Representation in Virtualized Cloud Computing Data Centers

Mateusz Guzek
University of Luxembourg
6, rue R. Coudenhove-Kalergi
Luxembourg, Luxembourg
Email: mateusz.guzek@uni.lu

Dzmitry Kliazovich
University of Luxembourg
6, rue R. Coudenhove-Kalergi
Luxembourg, Luxembourg
Email: dzmitry.kliazovich@uni.lu

Pascal Bouvry
University of Luxembourg
6, rue R. Coudenhove-Kalergi
Luxembourg, Luxembourg
Email: pascal.bouvry@uni.lu

***Abstract*—Management and optimization of cloud infrastructures combine multiple challenges. The optimization of data centers targets such objectives as performance, reliability, energy consumption, and security. To achieve these goals, multiple actions can be taken, for example, task and virtual machine allocation or infrastructure management. In this work we propose a model for representation of computing, memory, storage, and communication resources in cloud computing data centers. This model is relevant for the characterization of cloud applications, virtual machines, as well as physical servers. The performance evaluation and validation of the proposed model is carried out using the GreenCloud simulator. The obtained results show good agreement with the design objectives and confirm validity of the assumptions.**

I. Introduction

The environmental impact of data centers is significant and the growth dynamics of the IT sector urges professionals to tackle this problem. The swift emergence of cloud computing technologies amplifies this trend, as more computation and storage is moved from the end user devices to data centers. Migrating to the cloud brings also new opportunities, as consolidation and economy of scale enables the use of specialized technologies and management of resources to improve system efficiency.

Outsourcing to the cloud can be a critical decision for companies or individuals, as they shift from themselves the responsibility of achieving their objectives to the service providers. These objectives are formulated in Service Level Agreement (SLA) and may depend on the user needs and the type of the provided service. They can include specific requirements for performance, reliability, and security. These objectives are often orthogonal to the objectives of the service provider, which are focused on optimal resource utilization, reduced costs, low power consumption and maximizing the profit.

Optimizing data center operation is a complex problem. It should account for heterogeneity of resources, varied set of users with their own SLAs, multiple and often conflicting objectives, and the problem of scalability. It is difficult to model mathematically such complex systems. Instead, multiple data center simulation tools are used to conduct experimental studies. Most of these tools [1] include only simple model for the representation of physical computing servers and cloud applications which is focused on the computing capabilities and demands, while other types of resources such as memory, storage and networking must be taken into account.

In this paper, we propose a model for representation of physical resources, virtual machines, and cloud applications which explicitly and accurately accounts for multiple types of resources, such as processors, memory, storage, and networking. Simulation of the resource usage is important not only for assuring the correct execution of applications and platform management, but also for proper power accounting. Our previous experimental study [2] confirms an increased accuracy of power estimation while using larger set of predictors [3]. The applicability of the model is tested by assessing its overhead as well as presenting the expressiveness of its features. The proposed model is implemented and validated using GreenCloud simulator [4]. Currently, it is one of the leading cloud computing simulation platforms focused on the detailed simulation of communication processes in and between data centers as well as with the end users [5].

The rest of the paper is organized as follows: Section II describes the state-of-the-art approaches for modeling and simulation of data centers. Section III presents the proposed data center model. Section IV gives insight into design and implementation decision. Section V shows the performance of the implementation as well as features of the new model. Section VI summarizes the paper and presents future work directions.

II. State of the Art

In this section we review the most widely known cloud computing simulators and discuss on their resource allocation models.

978-0-7695-5095-4/13 $31.00 © 2013 IEEE
DOI 10.1109/CloudCom.2013.84

A. Models for Representation of Resources in Cloud Computing Simulators

ECOFEN [6] is an energy-aware simulator based on ns-2 [7]. It focuses on the correct assessment of the energy costs of the end-to-end communications in large-scale networks. ECOFEN implements a detailed model of the networks, but it does not precisely describe the end-devices resources other than network interface card. The proposed model is therefore complementary with ECOFEN model, as it does not refine the network topology modeling.

iCanCloud [8] is a simulator which models multiple types of resources and is based on the OMNET++ [9] and INET[1] frameworks. The VMs are managed by Cloud Hypervisor. Its current implementation allows using only single computing, memory, storage, or networking resource at a time and the underlying hardware configuration is not fully expressed, while the model proposed in this paper has no such limitations. It explicitly captures the processes at the host, VM and task levels. Additionally, it does not make any assumptions about the management of the platform, while Cloud Hypervisor is the entity designed for the management of the VMs in iCanCloud.

MDCSim [10] presents a model with CPU, disk and network interface resources. Each resource is modeled as M/M/1 queue. MDCSim describes cloud application using the following three typically used layers: web, application, and database. The communication between layers is also modeled using queuing models.

DCSim [11] supports simulations of systems composed of processor, memory, storage, and networking resources. To ensure high scalability, DCSim neglects data center network topology, but supports virtualization with migrations and replications, and provides resource managers for each type of the resources. The resource model proposed in this paper is more general and allows higher flexibility as it is adapted for more resource types, fine-grain resource specification, and multi-level virtualization.

CloudSim [12] is another well-known cloud computing simulator. The main difference between CloudSim, which is Java-based, and the class of ns-2 based simulators is in the simulation of networking. CloudSim uses delay matrix to simulate network topology and time for information delivery between any pair of nodes. This method is computationally efficient and allows the simulation of large-scale scenarios, but on the other hand it neglects a number of important processes in networking, such as traffic contention, bandwidth utilization, and communication protocol dynamics. CloudSim offers models for processing, memory, and network resources which are handled by separate modules.

[1] http://inet.omnetpp.org

B. GreenCloud Simulator

GreenCloud [4] is a well-known simulation tool which offers a fine-grained simulation of modern cloud computing environments focusing on data center communications and energy efficiency. GreenCloud is based on ns-2 [7] simulation platform. It offers a detailed modeling of the energy consumed by the elements of the data center, such as computing servers, switches, and network links, and implements energy-efficient resource allocation solutions with network awareness [13]. GreenCloud offers a thorough investigation of workload distributions with a specific focus devoted to the packet-level simulations of communications in the data center infrastructure, which provide the finest-grain control. The models of cloud applications explicitly account for communication process and network awareness [14]

GreenCloud supports the most widely used three-tier data center architecture as well as modern data center architectures, such as DCell, BCube, FiConn, and DPillar. Fig. 1 presents the three-tier architecture. It consists of the core tier at the root of the tree, the aggregation tier that is responsible for routing, and the access tier that holds the pool of computing servers arranged into racks. Computing servers are interconnected using 1 Gigabit Ethernet (GE) links, while aggregation and core switches are equipped with 10 GE ports. As a result, with standard racks supporting of up to 48 servers, the corresponding bandwidth oversubscription ratio is equal to 48/20 = 2.4:1 in the access and 1.5:1 in the aggregation networks. According to the basic model, an idle server consumes around two-thirds of its peak load to keep memory, disks, and I/O resources running, while the rest of the power is consumed by the CPU and can be scaled with the offered computing load. The energy consumption of network switches depends on: (a) type of switch, (b) number of ports, (c) port transmission rates, and (d) employed cabling solutions.

In previous versions of the GreenCloud simulator the primary focus was devoted to modeling of communication processes, while models used for describing computing servers remained simplistic. The servers were represented as single-core nodes, identical through the whole data center. A new model, which is proposed in this paper and was already implemented in GreenCloud, enables elastic and heterogeneous definitions of various types of resources. Moreover, as virtualization is fundamental for modern cloud computing scenarios, the proposed model also explicitly models virtual machines.

III. Proposed Model

The proposed model is designed to represent physical resources, virtual machines, and applications in cloud computing environments. It explicitly and accurately captures dependency on multiple types of resources, such as processors, memory, storage, and networking. The proposed model is based on the observations that the power consumed by

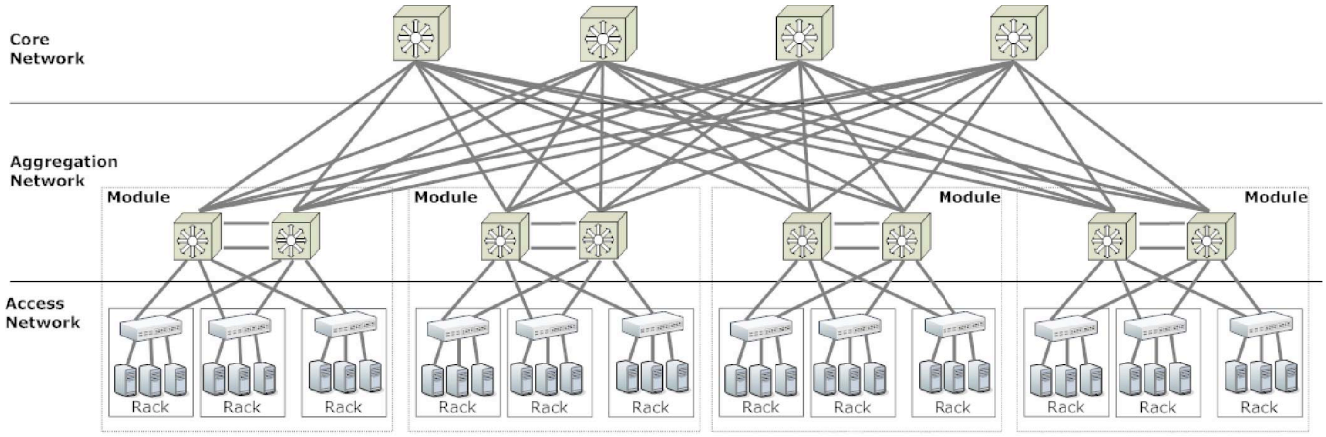

Figure 1. The GreenCloud simulator structure [4].

the system, as well as its performance, depends on the load levels of the hardware components. The first introduction of the proposed holistic model was focused on individual servers and experimental evaluation in terms of system power consumption and performance [2].

The proposed model is composed of the three main dimensions that represent system resources, cloud applications, and communication fabric.

The dimension of system resources represents computing nodes according to the hierarchical structure presented in Figure 2. Each node contains a layer specifying types of resources available to the consumers. In this work we distinguish computing, memory, storage, and networking types of resources. However, the model can easily scale to include other types of resources as well, e.g., additional GPGPU units. The resource types are further extended into a vector of *Resource Supplies* which models individual hardware components. The hardware components are described by their architecture and a vector of *Capacities*. The Capacities are used to quantitatively represent capabilities of a component. For example, a quad-core Intel CPU is a resource supply that belongs to the Computing resource type, with a Capacity vector containing four elements that correspond to the cores of the processor. Each core is expressed in terms of its computing capacity measured in MIPS or FLOPS in the vector. The Architecture of the Resource Supply is a label, which can be used during resource allocation. Further in the paper we assume a total ordering among the Architectures for each of the resource types and thus express Architecture as a real number, with superior Architectures labeled by a number greater than inferior ones. Figure 2 presents a graphical example of a server in this representation. As there is only one Resource Supply for each of the resource types, we can arbitrarily label their Architectures as 1.

The dimension of cloud applications describes an affect of the installed software stack. This dimension is further

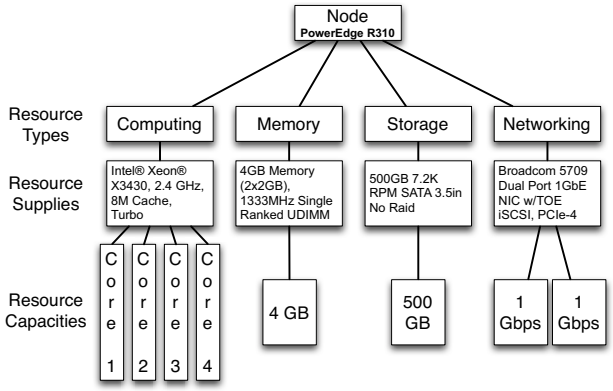

Figure 2. An exemplary representation of a server [2].

divided into three layers: *Cloud Applications, Virtual Machines, and Physical Nodes* (see Figure 3). The Cloud Applications layer models the properties of the load of the data center, the Virtual Machines layer describes the management and the configuration of the virtual machines hosted by the servers. Finally, the Physical Nodes layer describes hardware properties of the servers. In addition, we introduce the concepts of *Resource Provider* and *Resource Consumer*.

The Resource Provider is an entity that makes its resources available to the Resource Consumer. The process of resource allocation is in the matching of the *Resources* available at the Resource Providers with the *Resource Demands* of the Resource Consumers. A valid allocation must meet two basic conditions: the demands cannot exceed capacity of the resource provider and must not be allocated to a provision with an inferior Architecture.

The presented abstraction allows creating an arbitrary number of intermediary layers that act as Resource Providers and Resource Consumers at the same time. The lowest layer

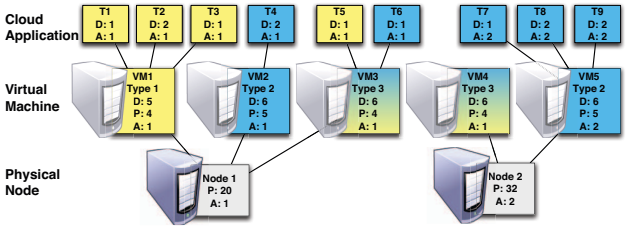

Figure 3. Resource allocation in a system with two nodes. Colors represent Cloud applications and VM types, which must be compatible. VM3 and VM4 are of the universal type, i.e., they can process both types of tasks. T ? Cloud Applications Tasks D – Resource Demand, P – Resource Provision, A – Architecture label.

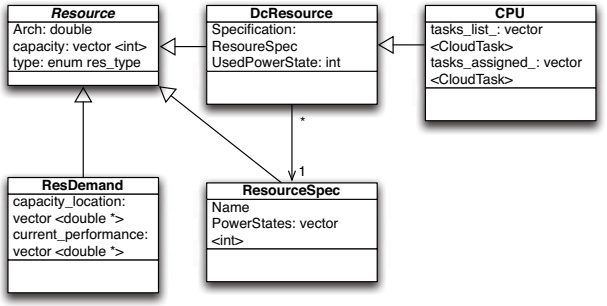

Figure 4. Diagram of resource-related classes.

is formed by physical nodes. The intermediary layer contains VMs. The VMs can reserve resources of the physical hosts and make them available for task computation. For this reason they can act as both the Resource Providers and the Resource Consumers. Finally, cloud applications are executed at the highest layer. They are final Resource Consumers representing the datacenter workload.

Tasks in a system are grouped into cloud applications that correspond to running services. Similarly, each VM has a type that corresponds to its functionality. Each cloud application has a defined set of VM Types that can process it. Finally, each task is defined with input and output sizes that must be transmitted over the network respectively prior and after the task execution.

Figure 3 presents an example of the single resource type allocation in a system with two nodes. The nodes are heterogenous in terms of the resource provisioning and their architecture. The sum of the VM demands is smaller than or equal to the resources available at the nodes. Similarly, the sum of the application demands can fit into the resource available at the Resource Providers.

The dimension of communication fabric is represented as a graph. The nodes of this graph, representing computing servers and network switches, form network topology. The edges of the graph represent communication links with a predefined bandwidth and delay.

IV. Design and Implementation

This section describes the design and implementation decisions of the proposed model. Due to the size and complexity of underlying implementation and the space limitations we present only key features defining efficiency of the proposed solution.

A. Multiple Resource Types

Data center servers can contain multiple types of resources such as computing (or CPU), memory, storage, and networking. The availability of these resources has impact on the performance and energy consumption of the system. Another argument for explicitly mapping distinct types of resources is associated with management interfaces of modern hypervisors and cloud management systems, which allocate and consolidate VMs according to their demands and available resources of physical servers. Figure 4 presents software classes that represent system resources.

Resource is the abstract class and the root of resource-related class hierarchy. It is composed of a vector of capacities, represented with double precision floating point numbers (doubles). The choice of vector representation allows including multiple types components of resources, such as multiple cores of a CPU or several ports in a network card. Each resource is described by its type (represented by enumeration that includes values for computing, memory, storage, and networking types) and the architecture (represented as double). The main purpose of this class is to unify the description of resources in a system.

Resource is also a superclass of the following three classes: *ResourceSpec*, *DcResource*, and *ResDemand*. These classes describe different concepts of the resource usage. The *ResourceSpec* class describes hardware specification. It also stores the unique name of the hardware component type and a vector of power states available for the component represented with integer numbers. The capacities of the ResourceSpec define maximum achievable level of the resource performance. During initialization of the simulation all the required ResourceSpec instances are created in TCL script and are assigned to the DataCenter object. This approach allows storing hardware component specification in configuration files, elastically deciding on what kinds of ResourceSpecifications are used, and dynamically adding new ResourceSpec specification during simulations.

DcResource is a class used to represent instances of ResourceSpec, which are intended to be components of servers. It stores the pointer to the ResourceSpec, which defines its behavior. The state of each component is expressed by a capacity vector inherited from the Resource. In addition, an integer field specifies a power state of the resource. The *CPU* class is derived from the DcResource class and extends its functionality by managing lists of tasks executed and assigned to a CPU.

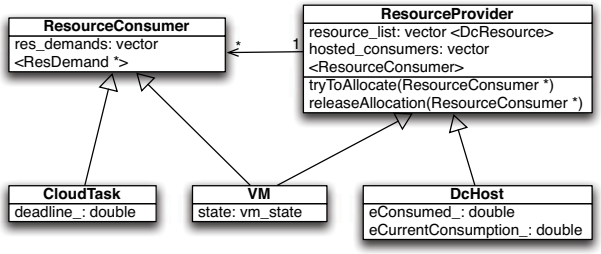

Figure 5. Diagram of allocation-related classes.

Finally, *ResDemand* class, implemented for the demand side, represents resource requirements. For ResDemand, the capacity parameter defines the requirement for the resource (for static resources, such as memory or storage) or the amount of the unprocessed workload (for dynamic resources, such as computing).

B. Resource Allocation

The process of resource allocation is based on two abstract classes: ResourceConsumer and ResourceProvider. Figure 5 presents a simplified class diagram. *ResourceConsumer* is in the root of the hierarchy of classes representing resource consumers, while the *ResourceProvider* is in the root of the hierarchy of classes of the resource provides. Most of the functionalities needed for the resource allocation are implemented in these classes, making the whole process general and in line with the concepts outlined in Section III.

Allocation of a ResourceConsumer is performed by the *tryToAllocate* function of the ResourceProvider. The function sequentially iterates over the elements of *res_demands* vector of the ResourceConsumer and their capacities, and allocates them using a first-fit algorithm applied on the DcResource objects in the list of resources. The usage of the simple first-fit algorithm is intended; the change of order in each of the vectors (e.g., sorting using a metric) is the preferred and elastic way of influencing the resource allocation strategy by a ResourceProvider. Another possibility is to provide new implementation of the tryToAllocate function. *releaseAllocation* is a function with an opposite effect: it releases the resources allocated to a consumer, either in the end correct allocation (e.g., at the end of task execution), or when the allocation attempt was unsuccessful.

The CloudTask, VM, and DcHost classes, are subclasses of the ResourceConsumer and ResourceProvider classes, and directly represent the three layers of the model presented in Fig. 3. The mechanism of abstraction makes these classes simple and specific at the same time. *CloudTask* extends ResourceConsumer by adding a *deadline*, which is expressed in the terms of the simulator time and represented as double. *DcHost* is the class associated with computing nodes in a data center. Being the first in the chain of ResourceProviders in the model, it is composed of resources that follow resource specification of the hardware elements. In addition, DcHost is responsible for calculation of energy consumption (*eConsumed_*) of a server. The *VM* class represents virtual machines. It is used to create elements that form intermediary layer and therefore, it is derived from both the ResourceConsumer and the ResourceProvider classes. VMs reserve resources available at the lower layer (typically at DcHost) and make them available as virtual resources to the higher layer entities (typically CloudTasks). The VM class contains information about the current state of the VM, that can be *Ready, Running, Suspended, Stopped*, or *Crashed*. The state of a VM is used for determining server utilization and guiding the progress of tasks allocation to the VM.

V. Experiments

The following sections are devoted to the presentation of the performance and expressiveness of the proposed model. Section V-A compares the performance of the proposed model with the performance of previous release version of the GreenCloud simulator, while Section V-B presents new features of simulator, enabled by the proposed model.

A. Performance of the Proposed Model

The runtime and memory requirements are selected as the main metrics for evaluating the simulation performance of the proposed model. Both of them are gathered using a standard linux `time` command. The runtime is the time spent by the process in user mode and the memory requirement is the maximum resident set size of the process. All simulations are executed on a reference VM with Ubuntu 12.04 operating system installed, which is provided with the GreenCloud 1.0.6[2]. The VM runs using VirtualBox 4.2.12[3] on a notebook with Intel i7-2720QM processor (quad core, 2.20 GHz), 8 GB of 1333 Mhz DDR3 RAM and SSD disk. The VM has allocated 2 VCPUs and 1536 MB RAM to ensure high availability of memory.

Standard configurations available in GreenCloud 1.0.6 include typical three-tier data center architecture of standard size of 1536 servers and smaller size of 30 servers, used mostly for simulator debugging. The specifics of these two configurations are presented in Table I. The core switches are connected with the aggregation switches by 10 Gb/s links, as well as aggregation and rack switches. Computing servers are connected with 1 Gb/s links to the rack switches. Each configuration is executed with load levels varied between 0 and 1 with a step of 0.1. The load is defined as the ratio of the computational requirements of the tasks to the computational power of the hosts in the data center.

All other setup parameters are based on the default configuration of GreenCloud 1.0.6. In the proposed model

[2]http://greencloud.gforge.uni.lu
[3]https://www.virtualbox.org

for resources, host runs a single VM that processes tasks. In such case Hosts, VMs and Tasks are described using four distinct types of resources (processing, memory, storage, and networking). The capacities of resource provisions and demands are set on each level of allocation to the values that result in the processing and the networking being the only bottlenecks. More precisely, each host is equipped with processor capable of computing 1000100 MIPS and with 1 GbE networking interface for both models. Moreover, in case of the proposed model the hosts include 4 GB of RAM and 250 GB disk. The hosted VM has VCPU with MIPS the same as for hosting processor, 1 GB of RAM and 10 GB of disk space. Each generated task requires 1 MB of RAM and 10 KB of storage. The aforementioned setup is selected to guarantee that the only overhead is the one induced by virtualization and multiple resource types.

Table I
TESTED REFERENCE CONFIGURATIONS

Configuration	Three-tier Debug	Three-tier Standard
Core Switches	2	8
Aggregation Switches	4	16
Access Switches	3	64
Hosts in a rack	5	3
Total hosts	30	1536

Figures 6 and 7 present the obtained results. The multiple resource model proposed in this paper is referred as Holistic, while the default model implemented in previous version if the GreenCloud simulator is referred as Simple. For both three-tier debug and three-tier standard topologies, the runtime is linear to the data center load. This behavior can be explained by the nature of the simulation of network based on ns-2: higher load levels correspond to more tasks being generated, that result in more packets send over the network. The data series for both models overlap, which confirms that the Holistic model induces only negligible overhead with respect to the cost of simulating networking events. This observation may be also valuable for the perspective users of the simulator, as it means that it is possible to implement additional lightweight functionalities without significant increase in simulation time. Both figures present also the overhead of initialization. It is negligible for smaller Debug configuration, but for the Standard case it is almost 500 s. Additionally, by comparing the runtime values for both configurations at the peak load, one can observe that an increase in the data center size of approximately 50 times leads to 100 times longer simulation.

Figures 8 and 9 present memory overhead of the Holistic model implementation. The holistic model requires more memory, but it always stays within additional 10% with the respect to the Simple model. This additional memory is required for the creation a separate objects for each resource demand and provision. Contrary to that, in the

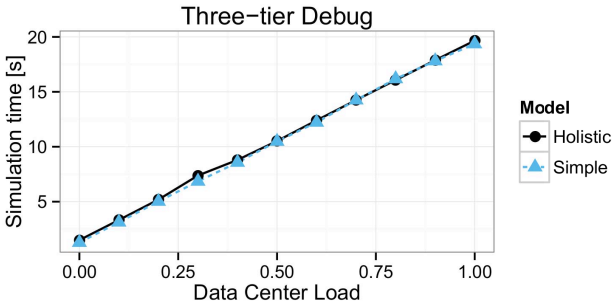

Figure 6. Runtime for Three-tier Debug topology.

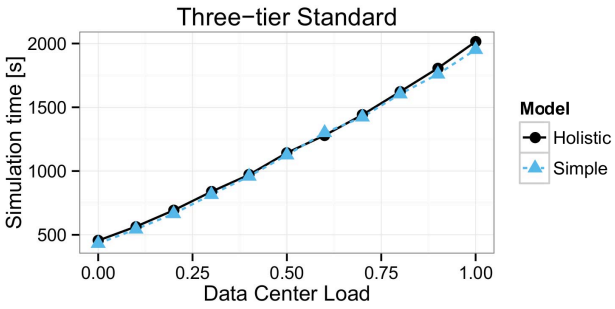

Figure 7. Runtime for Three-tier Standard topology.

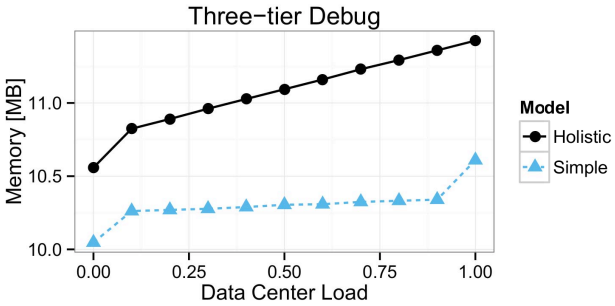

Figure 8. Memory requirement for various data center loads.

case of Simple model the corresponding values are stored as simple variables. The memory requirements do not follow strict linear pattern, however there is a partial linearity as shown in Figures 8 and 9. In the case of Debug topology only the memory requirements corresponding to idle states of the systems for both models and the peak load for the Simple model deviate from the linear pattern. In the case of Standard scenario, for the simulated data center load larger than 0.4, the memory requirements trend is more steep. Comparing the memory requirements of the Debug and Standard scenarios for load equal 1 reveals that increasing the data center size approximately 50 times results in 60 times higher memory requirement for both models.

The presented results prove that the Holistic model implemented in GreenCloud simulator does not increase execution time and has an acceptable memory overhead. However, its

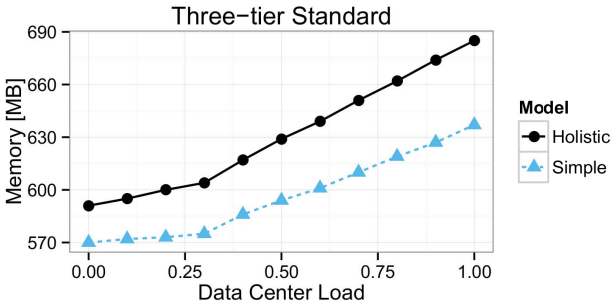

Figure 9. Memory requirement for various data center loads.

implementation gives user more flexibility in the creation of scenario for simulations. The following section presents new use cases enabled by Holistic model.

B. Applications of the Proposed Model

Practical implications of the proposed Holistic model include the usage of multiple resource types, the exploitation of hardware heterogeneity, and the virtualization capabilities. The following sections discuss the enabled scenarios and present the impact of varying configurations on the metric directly connected with the data center productivity and energy consumption: total load of data center.

1) Multiple Resource Types: The need to correctly allocate multiple resource types creates new challenges for resource allocation policies, which must ensure that all of the requirements of the resource consumers are fulfilled. Figure 10 presents a scenario with two factors that influence processing of cloud applications: task deadlines and memory requirements. In this case each host has a memory of 6 GB and a processor capacity which allows finishing a single task in one second. In cases when the deadline is equal to 1 s, each host can only execute one task at a time, so the memory requirement has no impact on the processing. In case of longer deadlines, multiple tasks can be executed by a single server. As DVFS is disabled, the processors always operate at their peak frequency, which results in a stepwise trace of the datacenter load. The width of a step is dependent on the number of tasks that can be concurrently executed on a single server, which in turn is function of the task deadline. In the case of tasks that require 2GB of memory, only 3 tasks can be executed concurrently on each server, which nullifies the impact of deadlines larger than 3s.

2) Heterogeneity: Real data centers are often heterogeneous, which can be due to different requirements of the applications being run, or simply caused by the presence of various generations of the hardware. The proposed holistic model allows to include individual descriptions for each computing server and hardware component in a data center. It makes the simulation convenient for investigating the effects of adding, modifying, or removing new hosts from a data center. The dynamic creation of resources and topology also enables the study of the transition periods. In addition, the proposed holistic model supports multiple commodity and experimental architectures; each host can have multiple supplies of each resource type, and each supply can have multiple capacities. For example, it is possible to define a node that combines multiple processors, with symmetric or asymmetric cores, a few banks of memory with various capacities, SSDs and standard disks, and several network interface cards. The holistic model can simulate the behavior of each configuration, e.g., it can distinguish between the cases of a single processor with the capacity of $2n$ MIPS, a single processor with two cores, each with capacity of n MIPS, or two processors with n MIPS. Whenever a cloud application imposes strict constraints on the hardware requirements, the architecture labels can be used to ensure that tasks of this application are running on compatible hardware. This flexibility can be also used to simulate the influence of the prospective hardware changes on the data center operation. The detailed information about utilization of resources can be used to feed more accurate power models, as increasing the set of predictors leads to a more accurate estimation of power [2].

An example of the impact of the various heterogeneous servers on the processing of the load of the data center is presented in Figure 11. With DVFS enabled, all the servers tend to increase the processing time until the deadline, which is set to 5 seconds, in order to reduce power consumption. For the memory there are two configurations: homogeneous, when all servers have either 4GB or 2GB of memory ("4GB All" and "2GB All"), and heterogeneous with two settings: "4GB / 2GB", where the first half of the hosts has 4GB of memory, while another half has only 2 GB, and the last "4GB mod 2GB" setting where hosts with 4GB and 2GB are interleaved. All executions with insignificant memory requirement of a single task (of 1MB) produce identical traces that achieve the highest data center utilization and successfully execute all generated tasks. In the case of large (2GB) memory demand, the prolonged execution of tasks results in blocking the servers and task failures, which can be observed as a decreased area under the trace of data center load. This effect becomes stronger for the systems with lower memory provisions. Of the 431 tasks, "4GB All" failed to complete 71, "2GB All" failed 251, and both mixed configurations 161 tasks. The mixed settings present exactly the same data center load trace, but various link utilization.

3) Virtualization: Multi tenancy is currently a standard practice for increasing the utilization of resources in virtualized data centers. Thanks to the explicit mapping of the VMs, it is possible to study the effects of VM consolidation. The proposed holistic model ensures isolation of the VMs, as they are restricted to the virtual resources that are allocated to them using the resource provider and resource consumer abstractions. As a result, it is possible that a physical host runs multiple VM instances, each associated with a different

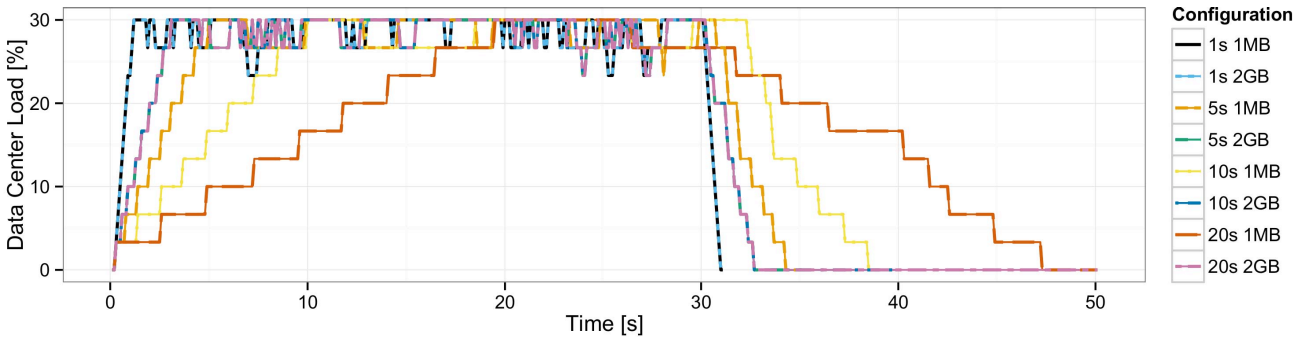

Figure 10. Configurations with varying task deadline and memory demand as limiting factors for multiprocessing.

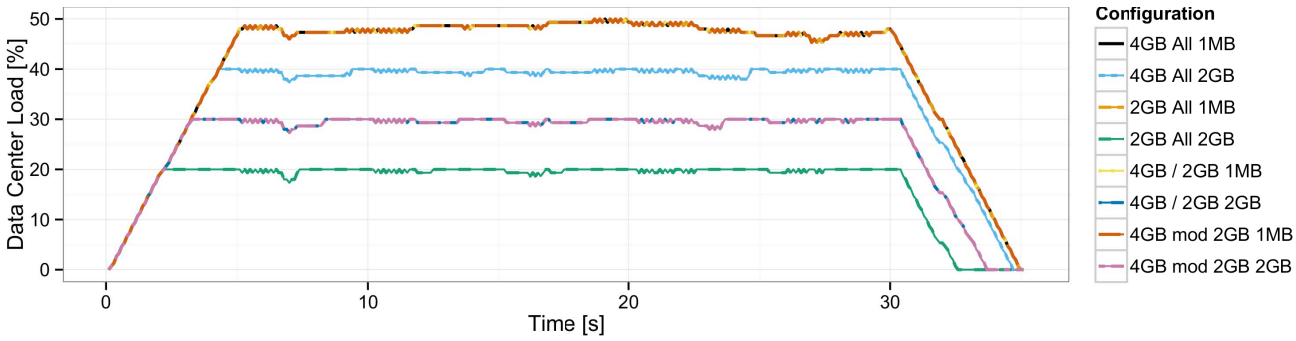

Figure 11. Interaction of memory provision and memory demand on heterogenous servers with enabled DVFS .

cloud application. Sharing processors is possible by the concept of VCPU and enables effective multi tenancy with the performance bounds. As VMs are ResourceProviders and ResourceConsumers, it is possible to elastically create multi-level virtualization scenarios in which a VM hosts another VM. Moreover, the resource consumers and providers can be grouped in lists specific for each service running in a data center, enabling customization of resource allocation strategies for each service.

Figure 12 presents the impact of the memory amount of VMs that execute tasks assigned to a cloud application. The processing capacities are set to achieve 50% of maximum data center load. The DVFS is disabled and the tasks have 5 s deadline and 2GB memory requirement. The plot presents the increased multitasking capabilities for the configurations with larger amounts of memory available. This behavior proves the correct isolation and resource management properties of implemented virtualization.

VI. Conclusion

The paper presents a novel holistic model for cloud computing data centers. The model can be applied to represent cloud applications, VMs, and physical hosts. Each of these entities is described by multiple resources: computing, memory, storage, and networking. The implementation of the model is carried out in the GreenCloud simulator [4]. The experiments prove the scalability of the proposed holistic model, as it does not increase the simulation time and creates a limited memory overhead of less than 10%. The proposed model increases the precision of simulations and enables a number of new simulation scenarios focused on heterogeneity of the hardware resources and virtualization.

The future work includes modeling and designing virtual machine migration processes, designing novel task and VM scheduling algorithms for the model, exploring efficient data centers usage strategies, and optimizations of the models implementation. In addition, the application of the proposed model to data replication techniques [15] will be investigated.

Acknowledgment

The authors would like to acknowledge the funding from National Research Fund, Luxembourg in the framework of ECO-CLOUD project (C12/IS/3977641) and Tri-ICT under the AFR contract no. 1315254.

References

[1] B. Aksanli, J. Venkatesh, and T. S. Rosing, “Using datacenter simulation to evaluate green energy integration,” *Computer*, vol. 45, no. 9, pp. 56–64, 2012.

[2] M. Guzek, S. Varrette, V. Plugaru, J. E. Pecero, and P. Bouvry, “A Holistic Model of the Performance and the Energy-Efficiency of Hypervisors in an HPC Environment,” in *Proc.*

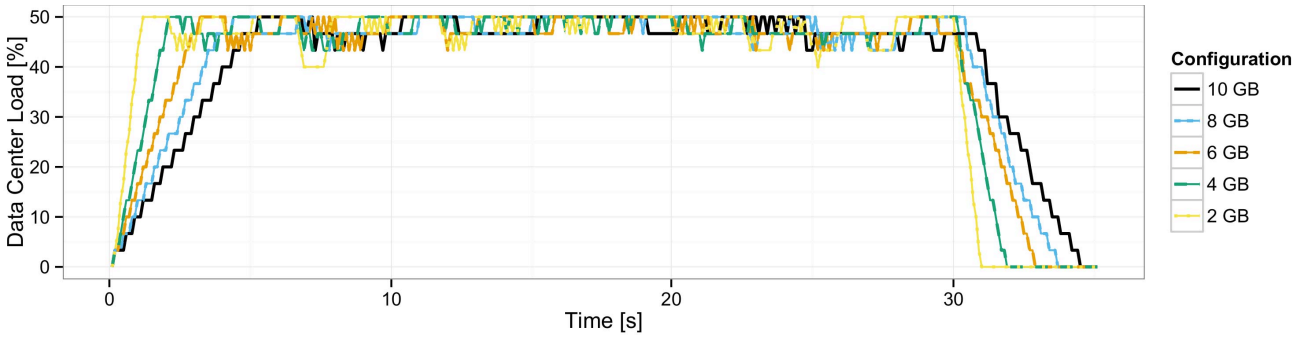

Figure 12. Impact of VMs memory capacity size on the load of data center. The VMs process the tasks of a single cloud application.

of the Intl. Conf. on Energy Efficiency in Large Scale Distributed Systems (EE-LSDS'13), ser. LNCS. Vienna, Austria: Springer Verlag, Apr 2013.

[3] S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A comparison of high-level full-system power models," in *Proceedings of the 2008 conference on Power aware computing and systems*, ser. HotPower'08, Berkeley, CA, USA, 2008, pp. 3–7.

[4] D. Kliazovich, P. Bouvry, and U. Khan, Samee, "Greencloud: A packet-level simulator of energy-aware cloud computing data centers," *The Journal of Supercomputing*, vol. 62, pp. 1263–1283, 2012, 10.1007/s11227-010-0504-1. [Online]. Available: http://dx.doi.org/10.1007/s11227-010-0504-1

[5] G. Sakellari and G. Loukas, "A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing," *Simulation Modelling Practice and Theory*, 2013.

[6] A.-C. Orgerie, L. Lefevre, I. Guerin-Lassous, and D. Pacheco, "Ecofen: An end-to-end energy cost model and simulator for evaluating power consumption in large-scale networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, 2011, pp. 1–6.

[7] S. McCanne and S. Floyd, "The network simulator – ns2." [Online]. Available: http://www.isi.edu/nsnam/ns/

[8] A. Núñez, J. L. Vázquez-Poletti, A. C. Caminero, G. G. Castañé, J. Carretero, and I. M. Llorente, "icancloud: A flexible and scalable cloud infrastructure simulator," *Journal of Grid Computing*, vol. 10, no. 1, pp. 185–209, 2012.

[9] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, ser. Simutools '08. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, pp. 60:1–60:10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1416222.1416290

[10] S.-H. Lim, B. Sharma, G. Nam, E. K. Kim, and C. R. Das, "Mdcsim: A multi-tier data center simulation, platform," in *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on*. IEEE, 2009, pp. 1–9.

[11] M. Tighe, G. Keller, M. Bauer, and H. Lutfiyya, "Dcsim: A data centre simulation tool for evaluating dynamic virtualized resource management," in *Network and Service Management (CNSM), 2012 8th International Conference on*. IEEE, 2012, pp. 385–392.

[12] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011. [Online]. Available: http://dx.doi.org/10.1002/spe.995

[13] D. Kliazovich, P. Bouvry, and S. Khan, "Dens: data center energy-efficient network-aware scheduling," *Cluster Computing*, vol. 16, no. 1, pp. 65–75, 2013. [Online]. Available: http://dx.doi.org/10.1007/s10586-011-0177-4

[14] D. Kliazovich, J. E. Pecero, A. Tchernykh, P. Bouvry, S. U. Khan, and A. Y. Zomaya, "Ca-dag: Communication-aware directed acyclic graphs for modeling cloud computing applications," in *IEEE 6th International Conference on Cloud Computing (CLOUD)*, Santa Clara, CA, USA, 2013.

[15] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, and A. Y. Zomaya, "Energy-efficient data replication in cloud computing datacenters," in *IEEE Globecom 2013 International Workshop on Cloud Computing Systems, Networks, and Applications (GC13 WS - CCSNA)*, Atlanta, GA, USA, 2013.