Received: 26 March 2020 | Revised: 27 March 2021 | Accepted: 30 March 2021

DOI: 10.1002/cpe.6326

RESEARCH ARTICLE

WILEY

Evaluation of self-organizing and self-managing heterogeneous high performance computing clouds through discrete-time simulation

Konstantinos M. Giannoutakis[1] | Christos K. Filelis-Papadopoulos[2] | George A. Gravvanis[3] | Dimitrios Tzovaras[1]

[1]Information Technologies Institute, Centre for Research and Technology Hellas, Thessaloniki, Greece

[2]Department of Computer Science, University College Cork, Cork, Ireland

[3]Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi, Greece

Correspondence
Konstantinos M. Giannoutakis, Information Technologies Institute, Centre for Research and Technology Hellas, 6th km Harilaou, Thermi, Thessaloniki, Greece.
Email: kgiannou@iti.gr

Funding information
Horizon 2020 Framework Programme, Grant/Award Number: Grant Agreement No. 643946

Abstract

Recently, a new management and resource delivery model based on self-organization and self-management (SOSM) principles for heterogeneous cloud infrastructures was proposed. The novel architecture was designed and implemented for the provision of cloud resources for delivering high performance computing services. The cost of deployment of such framework on large-scale cloud infrastructures is prohibitive, thus discrete-time simulation techniques were adopted in order to evaluate its efficiency and scalability. In this work, extensive simulation experimentation is being reported and discussed, concerning four evaluation criteria, for comparing the traditional centralized resources allocation scheme to the alternative of SOSM. From the results obtained, it can be derived that SOSM can provide improved service delivery, computational efficiency, power consumption, scalability and management of resources for millions of cloud nodes, compared to the centralized resource allocation approach.

KEYWORDS

cloud simulation, evaluation, high performance computing, self-management, self-organization

1 | INTRODUCTION

Recent technological advances and increased needs for efficient low energy computing lead the convergence of cloud computing and high performance computing (HPC) paradigms. This is realized, except from the increased research efforts during the last years,[1-3] by the cloud providers tendency to adopt energy and computing efficient accelerators in their data centers, such as graphics processing units (GPUs), field-programmable arrays (FPGAs) machines, and many integrated cores (MICs) chips. These infrastructures have a high degree of homogeneity and tight control due to the particularities of the involved hardware and the needs of the HPC software. The main challenge for infrastructure providers and HPC end users is the balance between performance, scalability, energy consumption, and minimal interference.[4]

Self-organization and self-management (SOSM) principles, originating from the concept of autonomic computing,[5] can minimize the interaction of end users with the cloud infrastructure, by introducing blueprints for detailing deployment plan, constraints and QoS parameters for the service execution. With this approach, the end user focuses on the actual service implementation (what needs to be done), while the cloud service provider concentrates the deployment and resource allocation (how it should be done).[4] The CloudLightning architecture was recently proposed[6,7] for addressing those challenges in heterogeneous cloud infrastructures. The proposed architecture enables SOSM in the cloud resource virtual hierarchy addressing the problems arising from the traditional centralized resource management at scale.

The CloudLightning framework was validated and evaluated in a small scale testbed, with the use of three HPC applications, namely, (i) oil and gas exploration, (ii) ray tracing, and (iii) genomics, that were properly adopted to be executed on GPU, MIC, and FPGA (data flow engine)

environments, respectively.[8] Details and insights regarding the use cases executions on heterogeneous cloud environment using the CloudLightning framework can be found in Reference 8.

For the evaluation of the aforementioned framework in hyper-scale infrastructures, simulation was used since experimentation at this scale is prohibitive. A parallel discrete-time simulation framework was developed[9,10] for eliminating the high memory and computational requirements of discrete-event simulators. This allowed the execution of experiments at large scale, by keeping the granularity at the levels needed by the defined evaluation criteria. Additionally, dynamic phenomena, such as SOSM, were simulated in an efficient manner.

In this article, the evaluation framework of the SOSM resource allocation scheme on heterogeneous HPC clouds is presented. The use of the simulation framework is defined and validated against other simulators, while experimentation is performed in order to measure the effect of SOSM in terms of service delivery, energy consumption, computational efficiency, and scalability. Numerical results indicate that, as the number of simulated cloud nodes increases, SOSM is efficient in the above evaluation criteria, yielding a competent resource allocation scheme for heterogeneous cloud infrastructures.

The remainder of this article is organized as follows. Section 2 briefly introduces the CloudLightning simulation framework that was used in order to support the simulation of hyper-scale heterogeneous HPC cloud environments. Section 3 presents the validation of the simulation framework, by direct comparison with the most used and known cloud simulators. Additionally, it provides some scalability validation against CloudSim, the most used simulator in the literature. The evaluation setup is given in Section 4along with the defined evaluation criteria. The extensive experimentation performed and obtained numerical results are given in Section 5. Concluding remarks and discussion are then given in Section 6.

2 | SIMULATING SOSM HETEROGENEOUS CLOUDS

SOSM principles for resource allocation in heterogeneous HPC cloud infrastructures were recently proposed.[7] The developed framework considers a hierarchical architecture where all nodes in all vertical layers are equipped with a set of strategies toward local or global goals. Local decisions are influencing evolution in the adjacent levels, in two-ways (top-down or bottom-up), so that each level is aligned with the global goals defined by the business or system objectives. This framework was designed and adopted for cloud infrastructures that follow the warehouse scale computer architectural model.[11] In this model, data centers (cells) contain multiple racks of usually homogeneous servers, while heterogeneity is realized in the rack level, where cells may contain racks of different CPU types or CPU-accelerator pairs.[9] The abstract architecture of the CloudLightning framework is depicted in Figure 1.

The evaluation of the proposed framework has been performed with the use of simulation, due to the fact that the cost of deployment on real cloud infrastructures is prohibitive, especially for large scale installations. For producing results with the same level of accuracy, the CloudLightning

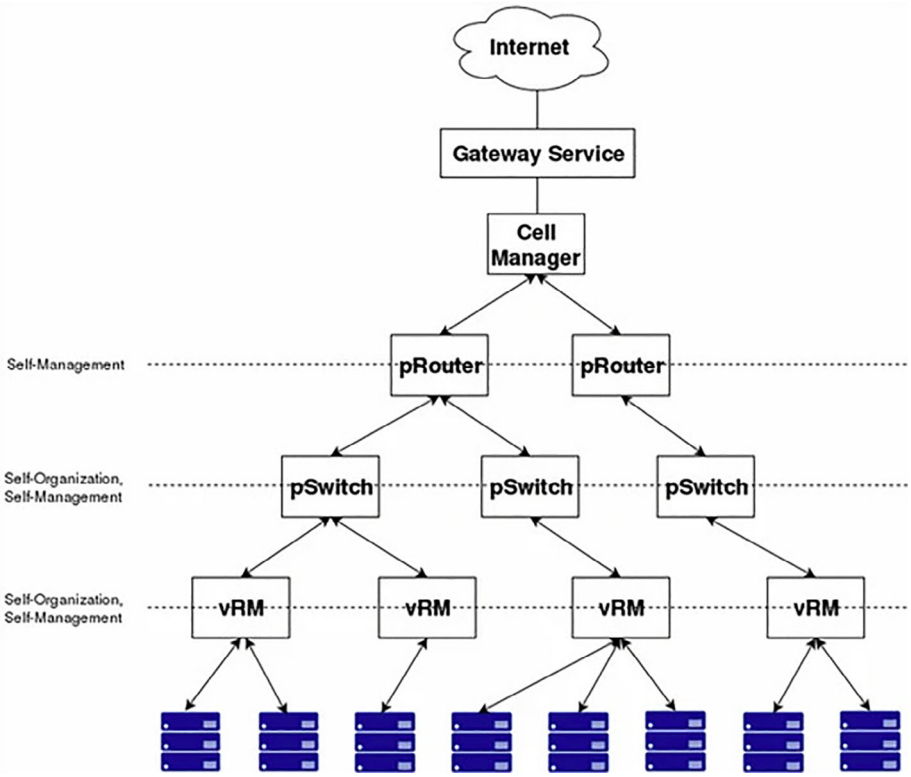


FIGURE 1 Abstract architecture of the CloudLightning system[7]

TABLE 1 Hardware characteristics of the computing nodes

Server type	Dell PowerEdge R730
CPU	Intel Xeon E5-2699 v4 2.20 GHz (44 cores)
MIPS	385,064
RAM	128 GB
Storage	40 TB
CPU utilization bins (even intervals)	[0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100] (%)
CPU utilization bins (uneven intervals)	[0, 10, 19.9, 30, 40.1, 50, 60, 69.6, 79.7, 90.4, 99.4] (%)
CPU power consumption (W)	[44.9, 88.4, 105, 122, 138, 152, 166, 182, 205, 236, 269]
CPU sleep power consumption	28 W

simulation framework was selected, since it is capable of simulating, except from SOSM clouds, traditional centralized clouds by utilizing the same models.[9,10]

The CloudLightning simulation framework was initially designed and developed in References 9 and 10, in order to support the simulation of SOSM strategies. The motivation behind its development, despite the fact that there is a plethora of available cloud simulation frameworks,[12] was to propose a framework that can:

- simulate dynamic resource allocation schemes,
- support heterogeneous cloud resources,
- simulate in parallel hyper-scale (i.e., millions of cloud nodes) cloud environments.

To achieve all these, a discrete-time principle was adopted, with the use of a time-advancing loop, without the need for pre-computation and storage of the events. With the discrete-time approach, several disadvantages of the existing discrete event simulators are eliminated. These include:

- the execution time for hyper-scale simulations (>1,000,000 servers)
- the support for dynamic components, such as SOSM, where changes in the configuration of the system may occur even if no events are happening on the simulated system
- hybrid distributed memory parallel systems cannot be efficiently used to accelerate the simulation processes, since the timeline of events needs to be separated into independent subsets, which is a difficult task when the number of events are in the order of millions

The framework was written in C++ and parallelized using message passing interface and OpenMP. It is available under an open source license at https://bitbucket.org/cloudlightning/cloudlightning-simulator. The framework is coupled with a web based user interface for allowing the visual formulation of the simulation input data, as well as visual representation of output data. The source code is located at https://bitbucket.org/cloudlightning/cl-simulatorvisualization. Sample files, in JSON document format, corresponding to the CloudLightning simulator input and output data are provided along with documentation, for enabling the easy setup of simulation experiments.

3 | COMPARISON AGAINST OTHER CLOUD SIMULATION PLATFORMS

In order to ensure that the developed simulation framework can provide accurate predictions, experimentation has been conducted against other open source cloud simulation frameworks. The same input data and the centralized resource allocation scheme were used for performing the comparison experimentation. Since the number of the open source cloud simulation platforms is large,[12] only the most used and well known were considered during this validation,[13,14] namely: CloudSim, DCSim, GreenCloud, and iCanCloud.

Experiments were conducted for the traditional resource allocation, for a data center with one cell, consisting of CPU-only servers (no accelerators), and for different numbers of hosts and VMs. The cloud nodes of the simulated data center were arbitrarily selected to be based upon a "Intel Xeon E5-2699 v4" 2.20 GHz processor with 44 cores and 385,064 MIPS.* The hardware characteristics are given in Table 1.

The time period of the simulation was set equal to 1 h (3600 s) and the gateway service (broker) update interval equal to 1 s. The characteristics of the application that was used during the simulations are presented in Table 2. The number of MIs was set equal to 1,386,228,316, such that each

*http://spec.org/power_ssj2008/results/res2016q2/power_ssj2008-20160328-00719.html

TABLE 2 Application characteristics during simulations

Instructions per app	1,386,228,316 MIs
VMs per app	1
vCPUs per VM	44
Memory per VM	4 GB
Storage per VM	20 GB
Network per VM	2.5 Mbps
Actual vCPU utilization	100%
Actual memory utilization	100%
Actual network utilization	100%

application requires 3600 s for its execution ((1,386,228,316 MIs)/(385,064 MIPS) = ,3599.9946 s). The bandwidth of the underlying network was set to 30 Gbps.

It is noted that for the estimation of the energy consumption of the CPU servers on the CloudLightning and DCSim simulators, the linear interpolation method on even intervals was used, unless otherwise stated. The GreenCloud and iCanCloud simulators are designed to compute the energy consumption of CPU, memory, storage, and network separately. Since no accurate data were available for the energy consumption of the CPU, the memory, the storage, and the network of Dell PowerEdge R730, the energy was estimated by the linear power consumption model,[14] using the maximum and the minimum power consumption of the server.

3.1 | Experiment 1: 100% utilization

The goal of this experiment is to measure the computed overall energy consumption with the use of various simulation platforms, by submitting a constant number of tasks. During this, the CPU utilization and the total number of accepted tasks are being calculated, as depicted in Table 3.

It can be observed that the total energy consumption in all cases is equivalent. Moreover, the processor utilization is close to 100% in all cases.

3.2 | Experiment 2: 50% utilization

Additional experiments were performed in order to test the outputs of the simulators when the number of instructions of each application was reduced by half (693,114,158 MIs). The processor utilization in all simulators was close to 100% for the first 1800 s, and 0% for the rest 1800 s of the simulation, thus the average CPU utilization was close to 50%. It is noted that CloudSim, GreenCloud, and iCanCloud simulators terminated

TABLE 3 CloudLightning, CloudSim, DCSim, GreenCloud, and iCanCloud energy consumption, CPU utilization, and number of the accepted tasks for different number of hosts and VMs (1,386,228,316 MIs per application, 100% vCPU utilization)

Hosts, VMs		50	100	200	500	1000
Number of submitted tasks		50	100	200	500	1000
Number of accepted tasks		50	100	200	500	1000
Total energy consumption (average processor utilization)	CloudLightning simulator	13.45 kWh (100%)	26.9 kWh (100%)	53.8 kWh (100%)	134.5 kWh (100%)	269.0 kWh (100%)
	CloudSim	13.44 kWh (100%)	26.89 kWh (100%)	53.77 kWh (100%)	134.43 kWh (100%)	268.85 kWh (100%)
	DCSim	13.263 kWh (98.332%)	26.526 kWh (98.33%)	53.053 kWh (98.333%)	132.632 kWh (98.333%)	265.264 kWh (98.333%)
	GreenCloud	13.1680 kWh (97.9%)	26.5965 kWh (98.8%)	53.4576 kWh (99.3%)	134.0442 kWh (99.6%)	268.3561 kWh (99.7%)
	iCanCloud	13.44992 kWh (100%)	26.89983 kWh (100%)	53.79967 kWh (100%)	134.49917 kWh (100%)	268.99833 kWh (100%)

the simulation after 1800 s, since all tasks had finished their execution, thus energy estimations were performed only for 1800 s. CloudLightning simulator continued the simulation after the tasks had finished their execution until the maximum simulation time was reached, thus during the time period that no tasks were executed on the data center, each server was consuming its sleep power consumption. Similarly to CloudLightning simulator, DCSim continued its execution until the simulation limit was reached. During the time period that no tasks were executed on DCSim simulator, each server was consuming its idle power consumption.

The experiments on the CloudLightning simulator were executed twice, once by setting the sleep power consumption of the servers equal to 0 W, and once equal to 28 W. Also, energy measurements on the DCSim simulator were collected once for the first 1800 s of the simulation (until all tasks had finished their execution, thus during the second half of the simulation time the power consumption of servers was considered equal to 0 W) and once for the total simulation time (during the second half of the simulation time the idle power consumption of servers was consumed). In Table 4, the results obtained by the simulators are presented.

The energy consumption estimation of CloudLightning and DCSim simulators, when the sleep power consumption of servers was considered equal to 0 W, is very close to the energy consumption estimation of CloudSim, GreenCloud, and iCanCloud simulators. Additionally, the energy estimations of CloudLightning and DCSim simulators for the total simulation time are very close.

3.3 | Experiment 3: 65% utilization

Experiments for different number of hosts and VMs were performed on the simulators by setting the vCPU utilization of the applications equal to 65%. The hardware and application characteristics of the configuration that was used, are those presented in Tables 1 and 2, respectively. In all simulators, during the simulation time (3600 s), the 65% of each CPU server was utilized. The energy consumption estimations of the experiments were equivalent in all simulators, as presented in Table 5. The estimations of CloudLightning, CloudSim, and DCSim simulators are calculated twice, once by using the linear interpolation and once by using the linear power model.[14]

3.4 | Experiment 4: CPU power models

The configuration of "Experiment 3" (1,386,228,316 MIs per application, 65% vCPU utilization) was used for the comparison of the various power models that have been implemented in CloudLightning and CloudSim simulators, since CloudSim is extensively used by the research community.[12,13] The various CPU power models that each of the two simulators support are summarized in Table 6.[14]

The estimations of the total energy consumption of CloudLightning and CloudSim, by using their supported CPU power models, are given in Tables 7 and 8, respectively. It can be observed that CloudLightning and CloudSim estimations when using the same CPU power models (i.e., linear, square, cubic, and linear interpolation on even intervals) are very close. Moreover, the interpolation methods (linear interpolation on even and

TABLE 4 CloudLightning, CloudSim, DCSim, GreenCloud, and iCanCloud energy consumption, CPU utilization and number of the accepted tasks for different number of hosts and VMs (693,114,158 MIs per application, 100% vCPU utilization)

Hosts, VMs		**50**	**100**	**200**	**500**	**1000**
Number of submitted tasks		**50**	**100**	**200**	**500**	**1000**
Number of accepted tasks		**50**	**100**	**200**	**500**	**1000**
Total energy consumption (average processor utilization)	CloudLightning simulator (0 W sleep power)	6.725 kWh (50%)	13.45 kWh (50%)	26.9 kWh (50%)	67.25 kWh (50%)	134.5 kWh (50%)
	CloudLightning simulator (28 W sleep power)	7.425 kWh (50%)	14.85 kWh (50%)	29.7 kWh (50%)	74.25 kWh (50%)	148.5 kWh (50%)
	CloudSim	6.72 kWh (50%)	13.44 kWh (50%)	26.88 kWh (50%)	67.20 kWh (50%)	134.39 kWh (50%)
	DCSim (0 W sleep power)	6.534 kWh (48.31%)	13.069 kWh (48.31%)	26.138 kWh (48.31%)	65.344 kWh (48.31%)	130.689 kWh (48.31%)
	DCSim (44.9 W idle power)	7.477 kWh (48.685%)	14.954 kWh (48.685%)	29.908 kWh (48.685%)	74.77 kWh (48.685%)	149.54 kWh (48.685%)
	GreenCloud	6.5747 kWh (48.85%)	13.2794 kWh (49.35%)	26.6909 kWh (49.6%)	66.9271 kWh (49.75%)	133.9878 kWh (49.8%)
	iCanCloud	6.724958 kWh (50%)	13.44991 kWh (50%)	26.89983 kWh (50%)	67.24958 kWh (50%)	134.49917 kWh (50%)

TABLE 5 CloudLightning, CloudSim, DCSim, GreenCloud, and iCanCloud energy consumption, CPU utilization, and number of the accepted tasks, for different number of hosts and VMs (1,386,228,316 MIs per application, 65% vCPU utilization)

Hosts, VMs		50	100	200	500	1000
Number of submitted tasks		50	100	200	500	1000
Number of accepted tasks		50	100	200	500	1000
Total energy consumption (average processor utilization)	CloudLightning simulator (linear interpolation model)	8.7 kWh (65%)	17.4 kWh (65%)	34.8 kWh (65%)	87.0 kWh (65%)	174.0 kWh (65%)
	CloudLightning simulator (linear model)	9.52825 kWh (65%)	19.0565 kWh (65%)	38.113 kWh (65%)	95.2825 kWh (65%)	190.565 kWh (65%)
	CloudSim (linear interpolation model)	8.70 kWh (65%)	17.39 kWh (65%)	34.78 kWh (65%)	86.95 kWh (65%)	173.90 kWh (65%)
	CloudSim (linear model)	9.52 kWh (65%)	19.05 kWh (65%)	38.09 kWh (65%)	95.23 kWh (65%)	190.46 kWh (65%)
	DCSim (linear interpolation model)	8.536 kWh (63.214%)	17.072 kWh (63.214%)	34.145 kWh (63.214%)	85.362 kWh (63.214%)	170.724 kWh (63.214%)
	DCSim (linear model)	9.328 kWh (63.214%)	18.656 kWh (63.214%)	37.313 kWh (63.214%)	93.282 kWh (63.214%)	186.563 kWh (63.214%)
	GreenCloud (linear model)	8.7304 kWh (63.9%)	17.4608 kWh (64.9%)	34,921.6 kWh (64.9%)	87.3041 kWh (64.9%)	174.6081 kWh (64.9%)
	iCanCloud (linear model)	8.55904 kWh (63.64%)	17.11808 kWh (63.64%)	34.23617 kWh (63.64%)	85.59042 kWh (63.64%)	171.18083 kWh (63.64%)

TABLE 6 CPU power models implemented in CloudLightning and CloudSim simulators

Power model	Description	CloudLightning	CloudSim
Linear	$P(u) = P_{min} + (P_{max} - P_{min})u$	✓	✓
Square	$P(u) = P_{min} + (P_{max} - P_{min})u^2$	✓	✓
Cubic	$P(u) = P_{min} + (P_{max} - P_{min})u^3$	✓	✓
Square root	$P(u) = P_{min} + (P_{max} - P_{min})\sqrt{u}$	✘	✓
Third degree polynomial (1)	Polynomial of Reference 9 with $P_{mid} = P_{min} + \frac{P_{max} - P_{min}}{2}$	✓	✘
Third degree polynomial (2)	Polynomial of Reference 9 with $P_{mid} = 5P_{max}/9$	✓	✘
Linear interpolation (even intervals)	–	✓	✓
Linear interpolation (uneven intervals)	–	✓	✘
Cubic spine interpolation (uneven intervals)	–	✓	✘

TABLE 7 Estimations of the total energy consumption in kWh, by using the CloudLightning CPU power models

Hosts, VMs, tasks	Linear	Square	Cubic	Third degree polynomial (1)	Third degree polynomial (2)	Linear interpolation (even intervals)	Linear interpolation (uneven intervals)	Cubic spine interpolation (uneven intervals)
50	9.52825	6.97911	5.32217	8.61074	8.40584	8.7	8.71667	8.69629
100	19.0565	13.9582	10.6443	17.2215	16.8117	17.4	17.4333	17.3926
200	38.113	27.9165	21.2887	34.443	33.6234	34.8	34.8667	34.7851
500	95.2825	69.7911	53.2217	86.1074	84.0584	87.0	87.1667	86.9629
1000	190.565	139.582	106.443	172.215	168.117	174.0	174.333	173.926

uneven intervals, cubic spline interpolation on uneven intervals) delivered very similar results. Since interpolation methods use real power measurements compared to the global methods (linear, square, cubic, square root, third degree polynomial), their estimations can be considered as more accurate. It can be also noticed that the estimations of the third degree polynomial methods yielded similar results with those that were obtained from the Interpolation methods.

TABLE 8 Estimations of the total energy consumption in kWh, by using the CloudSim CPU power models

Hosts, VMs, tasks	Linear	Square	Cubic	Square root	Linear interpolation (even intervals)
50	9.52	6.98	5.32	11.27	8.70
100	19.05	13.95	10.64	22.54	17.39
200	38.09	27.90	21.28	45.09	34.78
500	95.23	69.75	53.19	112.72	86.95
1000	190.46	139.50	106.38	225.45	173.90

3.5 | Experiment 5: Execution times

A performance experiment was conducted in order to compare the execution times of CloudLightning and CloudSim simulation frameworks. It is noted that the network requirements of the applications were set equal to zero for experiments larger than 10,000 servers, in order to avoid task rejection due to bandwidth limitations.

Three CPU servers were simulated/used in this experiment, as depicted in Table 9.

The produced results regarding the execution times are given in Table 10. It can be observed that the CloudLightning simulator performed significantly more efficient in terms of execution time, thus allowing the experimentation of millions of cloud nodes. CloudSim could not scale up to more than 100,000 servers in all cases due to memory exceptions, even though the JVM heap size was set to 44 GB, 28 GB, and 124 GB for the corresponding simulated cloud servers.

TABLE 9 Server characteristics for comparing the execution times of CloudLightning and CloudSim simulators

Server	CPU	No. of CPUs	No. of cores	CPU frequency	RAM
Dell PowerEdge T320	Intel Xeon E5-2420 v2	1	6	2.20 GHz (2.70 GHz max)	48 GB
Dell PowerEdge R730	Intel E5-2609 v4	2	8	1.70 GHz	32 GB
Dell PowerEdge C4130	Intel Xeon E5-2630 v4	2	10	2.20 GHz (3.10 GHz max)	128 GB

TABLE 10 Execution times of CloudLightning and CloudSim simulators for various number of hosts and VMs

Hosts, VMs	Number of accepted tasks	CloudLightning simulator			CloudSim		
		Dell PowerEdge T320	Dell PowerEdge R730	Dell PowerEdge C4130	Dell PowerEdge T320	Dell PowerEdge R730	Dell PowerEdge C4130
50	50	0 min 0.564 s	0 min 1.515 s	0 min 1.023 s	0 min 23.189 s	0 min 27.768 s	0 min 16.233 s
100	100	0 min 0.635 s	0 min 1.489 s	0 min 1.066 s	0 min 45.475 s	0 min 55.683 s	0 min 31.940 s
200	200	0 min 0.669 s	0 min 1.576 s	0 min 1.126 s	1 min 28.702 s	1 min 51.824 s	1 min 5.437 s
500	500	0 min 0.944 s	0 min 1.933 s	0 min 1.450 s	3 min 44.762 s	4 min 37.164 s	2 min 49.704 s
1000	1000	0 min 1.638 s	0 min 2.593 s	0 min 1.733 s	7 min 47.091 s	9 min 21.141 s	5 min 28.968 s
5000	5000	0 min 5.327 s	0 min 7.883 s	0 min 5.128 s	40 min 16.372 s	46 min 37.560 s	27 min 47.370 s
10,000	10,000	0 min 11.490 s	0 min 16.144 s	0 min 10.374 s	77 min 17.132 s	98 min 48.255 s	57 min 44.170 s
20,000	20,000	0 min 29.232 s	0 min 38.384 s	0 min 25.421 s	159 min 48.133 s	195 min 35.164 s	113 min 56.694 s
30,000	30,000	0 min 55.546 s	1 min 11.186 s	0 min 46.303 s	310 min 17.181 s	443 min 16.943 s	179 min 50.812 s
40,000	40,000	1 min 28.569 s	1 min 49.121 s	1 min 14.469 s	Out of mem.	Out of mem.	245 min 11.393 s
50,000	50,000	2 min 6.565 s	2 min 37.752 s	1 min 43.866 s	Out of mem.	Out of mem.	316 min 44.243 s
100,000	100,000	6 min 28.333 s	8 min 25.141 s	5 min 58.817 s	Out of mem.	Out of mem.	1245 min 25.131 s
200,000	200,000	33 min 41.884 s	28 min 55.639 s	18 min 22.936 s	Out of mem.	Out of mem.	Out of mem.

Concluding, CloudLightning simulation framework was found to be in accordance with well-known open source cloud simulation frameworks with respect to the estimated energy consumption. Additionally, due to its design and architecture, it outperformed CloudSim in total execution time and the ability to simulate large clouds.

4 | EVALUATION SETUP

In this section, the evaluation criteria, the simulation inputs, as well as the basic parameters for the cloud environments are discussed. The requirements and the size of the instances are based on use case trace data obtained on a small scale testbed.

4.1 | Evaluation criteria

The evaluation criteria have been defined in Reference 15, and consider measuring, with the use of simulation, the computational efficiency, energy consumption, service delivery, and scalability.

The CloudLightning simulation framework has been designed and implemented toward extracting metrics related to these criteria. More specifically, computational efficiency can be evaluated by comparing the cloud resources utilization considering as input a fixed number of incoming tasks. Energy consumption is measured with the use of proposed energy models[16,17] by calculating the total energy consumption used for the fulfillment of a fixed number of incoming tasks, while service delivery by the number of accepted and rejected tasks over the whole simulation time. Finally, scalability is evaluated by measuring the number of accepted/rejected tasks and utilization of resources.

4.2 | Use cases

Three HPC use cases were considered (oil and gas, ray tracing, genomics), as described in Reference 8. Traces from the use cases are analyzed and categorized according to their requirements to predefined resource configuration. These instances are given in Table 11. Each instance is considered to contain a CPU-only and a CPU-accelerator implementation, that the underlying system will decide which one to utilize for execution, according to the corresponding resource allocation strategy and the available hardware resources.

4.3 | Inputs of the model system

In the conducted experiments, each data center is composed of 100,000 servers, while these servers will be grouped in four different categories corresponding to four different types of cloud nodes:

- CPU only: 25,000 servers,
- CPU–GPU pair: 25,000 servers,
- CPU–MIC pair: 25,000 servers,
- CPU–DFE pair: 25,000 servers.

The number of simulated data centers is 13, reaching a total of 1,200,000 servers. The simulated time is chosen as 1 day or 86,400 s, since the distribution of tasks is uniform. The over-commitment ratios for all servers will be set to 1.0 for both CPUs and memory.

TABLE 11 Instance sizes and characteristics

Instance no.	Available implementations	vCPUs	Memory (GB)	Storage (GB)
1	CPU/CPU+GPU	1, 2, 4, 8, 12	0.5	10
2	CPU/CPU+FPGA	1, 2, 4, 8, 12	1.0	10
3	CPU/CPU+MIC	1, 2, 4, 8, 12	2.0	10
4	CPU/CPU+MIC	1, 2, 4, 8, 12	4.0	10
5	CPU/CPU+MIC	1, 2, 4, 8, 12	8.0	10

The interconnection speed of a cloud environment, according to experiments conducted by SPEC[18] is 10–20 Gbps. The network bandwidth in the experiments is set to 20 Gbps.

4.3.1 | Cloud nodes

The CPU based cloud nodes are based on the Dell PowerEdge C4120.[19] Each cloud node has two Intel Xeon E5-2630 v4 Processors with a total of 20 cores (40 threads) available, clocked at 2.20 GHz. The computational capability is $20 \times 4400.04 = 88,000.80$ MIPS. Hyper-threading is disabled. The memory for each node is 128 GB and storage capacity is 1024 GB. The chosen server type is able to support up to 4 GPUs or Intel Xeon Phi accelerators.

The Nvidia Tesla P100 is used to form CPU–GPU pair based servers.[20] The peak performance for the Nvidia Tesla P100 is 4.7 TeraFLOPS. In order to compute the MIPS of the accelerator, the peak theoretical performance of the corresponding CPU are required. This number is computed as: (base frequency) $\times$ (number of physical cores per processor) $\times$ (double precision FLOPs per cycle) $\times$ (number of processors) $= 2.20 \times 10 \times 16 \times 2 = 704$ GFLOPS. Thus, the MIPS of the Nvidia Tesla P100 is estimated to be $4700/704 * 88,000.80 = 587,505.34$. The minimum power consumption is approximately 32 W and maximum power consumption is approximately 250 W.[20] The power consumption of the accelerators is accumulated with the power consumption of the cloud node using the ρ parameter, which denotes the percentage of the application running on the accelerator taking into account delays induced by transferring data.[14] Each CPU–GPU pair node has 4 GPUs.

The Intel Xeon Phi 5110P is used to form the CPU–MIC pair cloud nodes.[21] The chosen CPU cloud node is able to support up to 4 Xeon Phi cards. The computational capability of each card is 507,494.40 MIPS. The minimum power consumption is 176.8 W,[22] while the maximum power consumption is 225 W.[21]

The CPU–DFE pair operates differently than the previous nodes. Each MPC-X node with eight MAX4 cards is directly connected to a CPU cloud node. The compute capability of each MAX4 card is 295,959.30 MAOPS.[†] The idle and the maximum power consumption for each MAX4 card is 17.3 and 25.1 W, respectively.

This section defined the inputs of the simulations, based on the three use cases. Trace data obtained from the small scale testbed were used for extracting the appropriate information that will feed the simulation.

4.4 | Traditional cloud environments

The traditional cloud environments are using a centralized approach where a broker is responsible for discovering appropriate resources to deploy incoming tasks as well as retaining state information of the underlying resources. The broker allocates the hardware resources corresponding to incoming tasks based on a search and deploy algorithmic procedure initially proposed in Reference 10 and extended in order to support accelerators. The incoming tasks are being selected randomly (using a uniform distribution) from the pool of available use case implementations of Table 11.

For the implementation of the broker search and deployment of tasks, a set of vectors of size N_{server} retains the available resources of each cell, namely virtual CPU cores (A_{vCores}), accelerators (A_{acc}), memory (A_{mem}), storage (A_{sto}), network (A_{net}), and network storage (A_{netsto}). An incoming task with n_{VM} VMs with n_{vCores} processing units per VM, n_{acc} accelerators per VM, n_{mem} memory per VM, n_{sto} storage capacity per VM, n_{net} network bandwidth, and n_{netsto} per available implementation is deployed or rejected by Algorithm 1.

With this approach, a sequential search across all the resources (or chunks of them in case of parallel execution) within a cell is performed, for retrieving the resources that fulfill the input tasks requirements. The interval for updating state information in the OpenStack cloud environments is 60 s.[‡] This update interval is used to conduct the simulation of a traditional cloud environment.

4.5 | SOSM cloud environments

The SOSM cloud environment is composed of 100 servers per vRM and 10 vRMs per pSwitch. The search and deployment algorithmic procedure for the incoming tasks for the SOSM cloud has been presented in Reference 9. The selection of input tasks is performed randomly at the level of use cases (in contrast with the traditional approach where input tasks feed the broker with the use case implementations to be executed), since SOSM shifts the selection of the use cases implementations from the user to the underlying system. The interval for updating state information at each level is chosen to be 60 s. The five assessment functions used, have been described in,[6] while the weights vector is chosen as $\vec{w} = [1\ 1\ 1\ 1\ 0]$, thus targeting maximum task throughput, and maximum energy, computational and management efficiency. The compaction VM placement strategy was used in all cases.[10]

[†] For simplicity, DFE MAOPS are treated similar to MIPS.
[‡] http://openstack.org

Algorithm 1. Sequential search and deploy

Initialize vector ID of size n_{VM} to -1 retaining the IDs of the resources to deploy task
if $n_{net} \leq A_{net}$ **then**
  A_{net}=A_{net}-n_{net}
  A_{netsto}=A_{netsto}-n_{netsto}
  for $v = 0 \rightarrow n_{vm}$ **do**
    for $r = 0 \rightarrow N_{server} - 1$ **do**
      if $n_{vCores} \leq A_{vCores}$ **then**
        d=probe(r, n_{vCores}, n_{acc} , n_{mem} , n_{sto})
        if d=true **then**
          ID_v=r
          $(A_{vCores})_r = (A_{vCores})_r - n_{vCores}$
          $(A_{acc})_r = (A_{acc})_r - n_{acc}$
          $(A_{mem})_r = (A_{mem})_r - n_{mem}$
          $(A_{sto})_r = (A_{sto})_r - n_{sto}$
          break
        end if
      end if
    end for
    if ID_v = -1 **then**
      A_{net}=A_{net}+n_{net}
      A_{netsto}=A_{netsto}+n_{netsto}
      for $r = 0 \rightarrow v - 1$ **do**
        $(A_{vCores})_r = (A_{vCores})_r + n_{vCores}$
        $(A_{acc})_r = (A_{acc})_r + n_{acc}$
        $(A_{mem})_r = (A_{mem})_r + n_{mem}$
        $(A_{sto})_r = (A_{sto})_r + n_{sto}$
      end for
      Reject the task and **return**
    end if
  end for
  for $v = 0 \rightarrow n_{VM}$ **do**
    Deploy VM to resource with ID ID_v
  end for
  Attach ID vector to the task
  Enqueue task to the task queue of the Broker and **return**
else
  Reject the task and **return**
end if

5 | NUMERICAL RESULTS

The simulation experiments were performed on the ARIS HPC infrastructure.[§] Each compute node consists of 2x Ivy Bridge - Intel Xeon E5-2680v2 (10 cores), with 64 GB of RAM. Each compute node was assigned to perform the simulation of one cell, while one node was performing the simulation of the gateway service.

The number of incoming tasks was sampled by a uniform random distribution over the intervals [0 … 20], [0 … 40], [0 … 60], [0 … 80], [0 … 160], [0 … 320], and [0 … 640]. The choice of uniform distribution was performed for assessing both the traditional and the SOSM approaches without the need for taking into account particularities or disadvantages induced by the choice of input tasks distribution.

[§]Through access granted from the Greek Research and Technology Network (GRNET) High Performance Computing Infrastructure https://hpc.grnet.gr/en/ under projects PR004033-ScaleSciCompII and PR006053-ScaleSciCompIII.

The simulation results for the traditional cloud environment for various numbers of incoming tasks are given in Table 12, while the results for the SOSM cloud environment are given in Table 13. Another important issue is the effect of self-organization to the SOSM system. Self-organization is triggered when a task, reaching a vRM, cannot be accommodated due to lack of available resources. The vRM triggers a self-organization strategy where resources of vRMs under the same pSwitch are sought. The effects of SOSM are depicted by performing a series of experiments with large number of incoming tasks.

TABLE 12 Simulation results of the traditional cloud environment

Maximum tasks per second	20	40	60	80	160	320	640
Total energy consumption (MWh)	12,717.16	12,754.66	12,792.30	12,830.04	12,980.22	13,215.15	13,422.20
Total number of submitted tasks	863,927	1,727,546	2,593,862	3,464,579	6,923,827	13,838,235	27,564,857
Total number of accepted tasks	863,927	1,727,546	2,593,862	3,464,579	6,923,827	12,709,415	19,873,184
Total number of rejected tasks	0	0	0	0	0	1632	420,154
Average processor utilization over active servers	73.87%	74.70%	74.98%	75.12%	75.75%	74.62%	69.26%
Average processor utilization	1.70%	3.40%	5.11%	6.84%	13.68%	23.43%	27.44%
Average memory utilization over active servers	13.21%	13.34%	13.38%	13.41%	13.51%	13.35%	12.42%
Average memory utilization	0.31%	0.62%	0.93%	1.24%	2.47%	4.23%	4.94%
Average network utilization	3.428E-13%	6.89E-13%	1.04E-12%	1.40E-12%	2.83E-12%	5.10E-12%	7.07E-12%
Average storage utilization over active servers	3.02%	3.05%	3.06%	3.07%	3.10%	3.06%	2.87%
Average storage utilization	0.07%	0.14%	0.21%	0.28%	0.57%	0.97%	1.15%
Average accelerator utilization over active servers	8.77%	8.85%	8.90%	8.89%	8.94%	10.11%	15.54%
Average accelerator utilization	0.19%	0.39%	0.58%	0.77%	1.54%	3.08%	6.13%

TABLE 13 Simulation results of the self-organizing self-managing cloud environment

Maximum tasks per second	20	40	60	80	160	320	640
Total energy consumption (MWh)	12,702.59	12,726	12,749.30	12,772.48	12,866.74	13,055.36	13,432.93
Total number of submitted tasks	864,970	1,732,851	2,593,975	3,448,771	6,911,275	13,832,798	27,650,718
Total number of accepted tasks	864,970	1,732,851	2,593,975	3,448,771	6,911,275	13,832,798	27,650,683
Total number of rejected tasks	0	0	0	0	0	0	35
Average processor utilization over active servers	41.93%	43.31%	43.64%	43.93%	44.49%	44.93%	45.34%
Average processor utilization	0.43%	0.86%	1.28%	1.71%	3.44%	6.89%	13.78%
Average memory utilization over active servers	5.07%	5.28%	5.32%	5.36%	5.44%	5.49%	5.54%
Average memory utilization	0.05%	0.10%	0.16%	0.21%	0.42%	0.84%	1.69%
Average network utilization	2.68E-11%	5.36E-11%	8.03E-11%	1.07E-10%	2.14E-10%	4.28E-10%	8.56E-10%
Average storage utilization over active servers	1.47%	1.53%	1.54%	1.55%	1.57%	1.58%	1.60%
Average storage utilization	0.01%	0.03%	0.05%	0.06%	0.12%	0.24%	0.49%
Average accelerator utilization over active servers	36.78%	38.14%	38.44%	38.70%	39.19%	39.59%	39.94%
Average accelerator utilization	0.37%	0.75%	1.13%	1.51%	3.03%	6.07%	12.14%

5.1 | Computational efficiency

Evaluating the computational efficiency for the two approaches, includes direct comparison using the metrics of cloud hardware utilization (CPU, accelerators, memory, network, and storage). The results have been presented in Tables 12 and 13, while a graphical representation of the hardware utilization is depicted. More specifically, in Figure 2, the utilization of the cloud hardware components is given for both the traditional centralized and SOSM cloud delivery models.

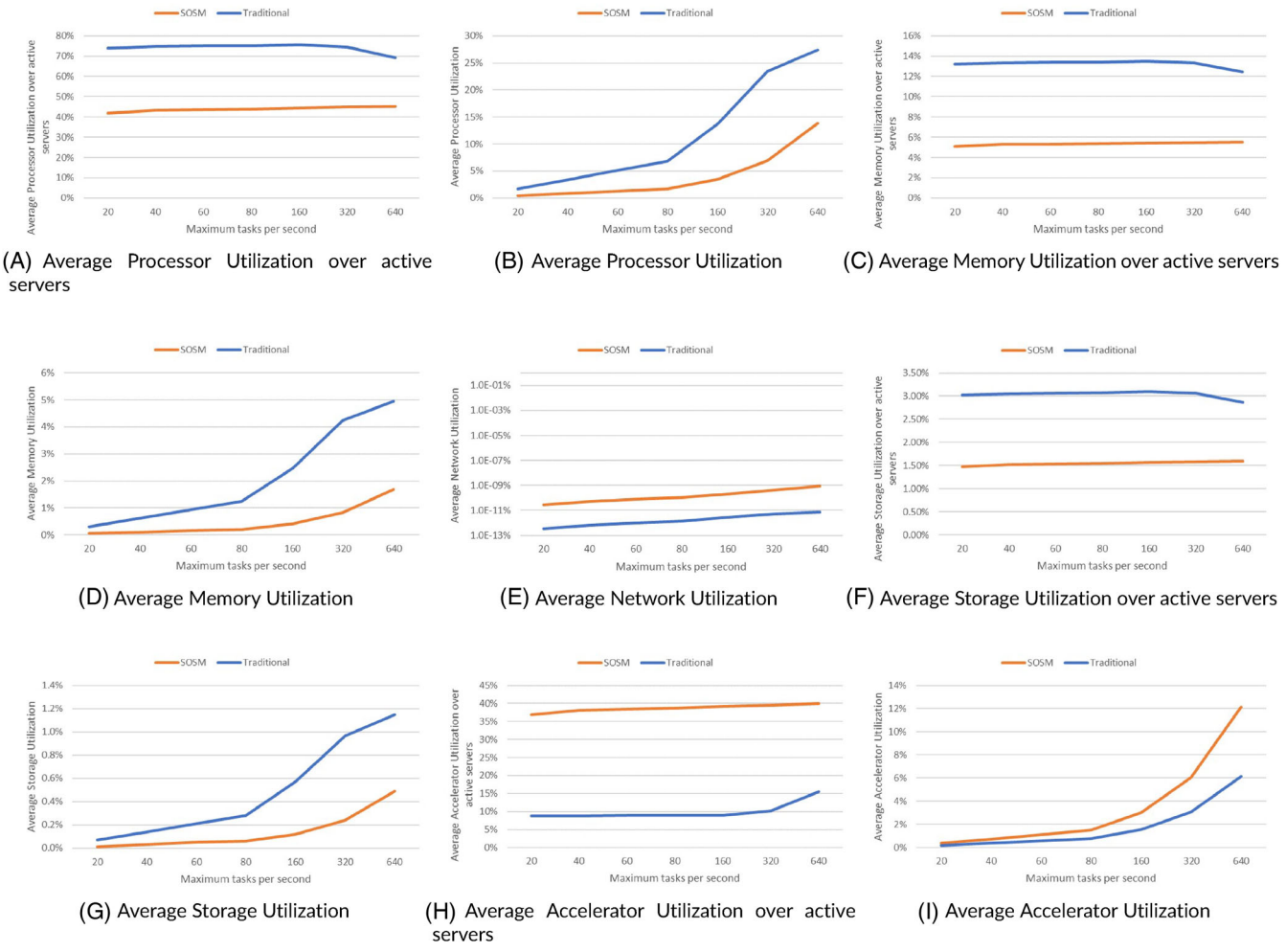
It can be observed that the utilization of processors, memory, network, and storage for the traditional delivery system is higher than SOSM for all numbers of incoming simulated tasks. This is due to the increased utilization of accelerators with SOSM resource allocation mechanism, that is, level of 40% instead of 10% over active servers. Allocating accelerators for tasks, not only reduces power consumption but also releases hardware resources (memory, network, and storage) since tasks are executed in a shorter time frame.

5.2 | Energy consumption

The total energy consumption for the two approaches has been given in Tables 12 and 13, and is depicted in Figure 3.

It is observed that the difference between the two allocation schemes is small, indicating that the two approaches behave almost the same in terms of energy consumption. By taking into account the total number of accepted tasks in each case, the SOSM mechanism is capable of executing almost 39% more tasks with the same energy consumed (due to the higher accelerator utilization), leading to the conclusion that SOSM is more energy efficient than the traditional centralized cloud resource allocation system.

Table 14 indicates that for all incoming tasks, the average energy consumption of each task using SOSM is less than the traditional centralized cloud delivery approach.



(A) Average Processor Utilization over active servers (B) Average Processor Utilization (C) Average Memory Utilization over active servers

(D) Average Memory Utilization (E) Average Network Utilization (F) Average Storage Utilization over active servers

(G) Average Storage Utilization (H) Average Accelerator Utilization over active servers (I) Average Accelerator Utilization

FIGURE 2 Hardware utilization for traditional centralized and SOSM clouds

FIGURE 3 Total energy consumption in MWh for traditional centralized and SOSM clouds

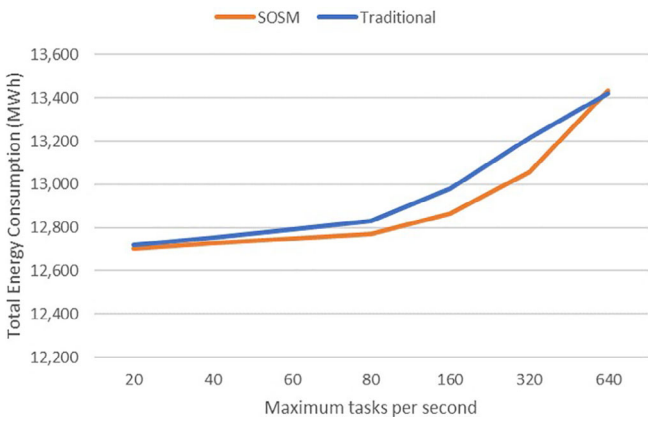


TABLE 14 Energy consumption (kWh) over number of accepted tasks

Maximum tasks per second	20	40	60	80	160	320	640
Traditional cloud	14.720	7.383	4.932	3.703	1.875	1.040	0.675
SOSM cloud	14.686	7.344	4.915	3.703	1.862	0.944	0.486

5.3 | Service delivery

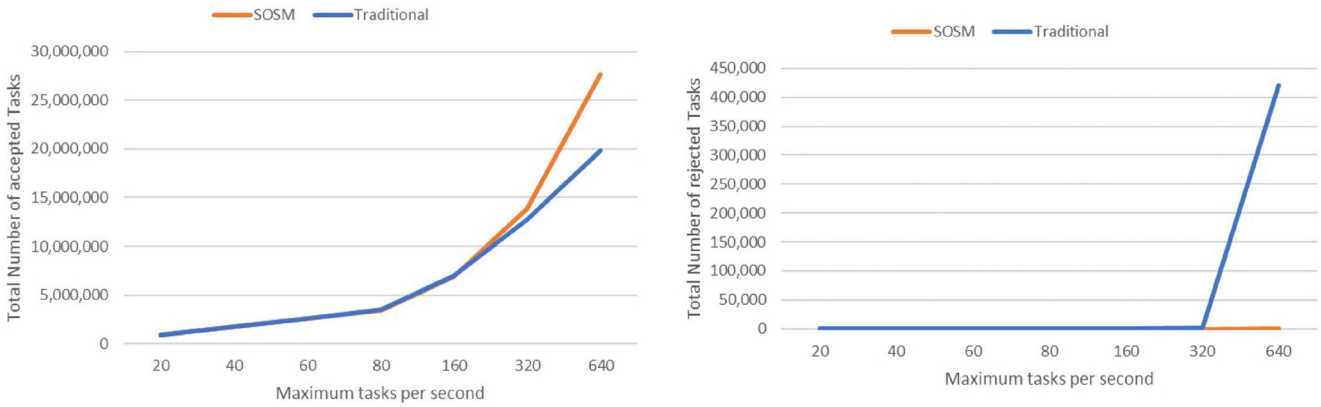
Service delivery is compared in terms of the number of accepted and rejected tasks over the whole simulation time. These numbers have been given in Tables 12 and 13, while in Table 15 the percentages of the accepted tasks are presented. Moreover, the total number of accepted and rejected tasks are depicted in Figure 4.

Both traditional and SOSM schemes satisfy all incoming tasks for up to 160 tasks per second, for the cloud hardware configuration simulated. Then, the traditional cloud starts rejecting tasks due to its inability to allocate more resources to those tasks. This behavior is more apparent in the case of 640 tasks per second, where the percentage of accepted tasks is about 72% while SOSM rejects only 35 tasks out of 27,650,718.

The service delivery of the traditional and SOSM schemes, in terms of search operations, was also compared in order to assess the delay during the process of searching resources. The average number of searching operations per incoming task for the two systems is given in Tables 16 and 17.

TABLE 15 The percentage of accepted tasks over the whole simulation time for traditional centralized and SOSM clouds

	Number of incoming tasks per second						
	20	40	60	80	160	320	640
Traditional cloud	100.000%	100.000%	100.000%	100.000%	100.000%	91.843%	72.096%
SOSM cloud	100.000%	100.000%	100.000%	100.000%	100.000%	100.000%	99.999%



(A) Total Number of accepted Tasks

(B) Total Number of rejected Tasks

FIGURE 4 The number of accepted and rejected tasks for traditional centralized and SOSM clouds

TABLE 16 The average number of searching operations per hardware type and number of incoming tasks per second for the traditional cloud environment

	Hardware type			
Tasks per second	CPU only	CPU–GPU	CPU–MIC	CPU–DFE
20	4210.6	832.8	133.7	303.4
40	8393.8	1606.2	222.4	522.9
60	12,554.0	2412.1	308.2	757.3
80	16,773.0	3194.1	398.1	997.5
160	33,269.0	6330.4	618.1	1879.0
320	54,823.0	12,595.0	1106.9	3680.9
640	54,794.0	25,002.0	2124.4	7338.6

TABLE 17 The average number of searching operations per hardware type and number of incoming tasks per second for the SOSM cloud environment

	Hardware type			
Tasks per second	CPU only	CPU–GPU	CPU–MIC	CPU–DFE
20	0.0	35.0	28.9	29.7
40	0.0	37.8	28.9	30.5
60	0.0	45.2	28.9	32.6
80	0.0	51.0	28.9	34.6
160	0.0	76.5	30.0	42.6
320	0.0	126.3	34.7	58.1
640	0.0	225.4	42.8	88.3

The number of search operations per incoming task is substantially reduced for the SOSM cloud. This is expected since the search space of a vRM, a pSwitch or a pRouter is smaller than that of all servers in a traditional cloud environment. This reduced number of search operations leads to substantially improved service delivery, since reduced time for resource discovery per task is required.

5.4 | Scalability

Scalability in terms of number of incoming tasks by preserving the cloud hardware constant is depicted in Tables 12 and 13. As the number of tasks increases, SOSM has a better performance in task acceptance rate, utilization of resources, and energy efficiency.

Increasing the hardware resources, that is, number of cells, by keeping constant the number of incoming tasks per second, that is, 640 tasks per second, enables the acceptance of all tasks since a large number of hardware resources can satisfy all requests. Tables 18 and 19 depict these results, where the SOSM allocation mechanism remains advantageous while the number of hardware resources increases.

TABLE 18 Scalability of the traditional cloud by increasing the number of cells with constant number of incoming tasks

Number of cells	20	30	50	100	175
Total energy consumption (MWh)	20,432.63	30,421.59	49,970.23	98,731.43	171,882.83
Total number of submitted tasks	27,684,155	27,647,049	27,737,523	27,569,356	27,577,305
Total number of accepted tasks	22,970,631	26,953,542	27,737,523	27,569,356	27,577,305
Total number of rejected tasks	148,637	10	0	0	0
Average processor utilization over active servers	72.82%	75.89%	76.03%	75.55%	74.98%
Average processor utilization	24.80%	22.64%	14.22%	7.06%	4.03%
Average memory utilization over active servers	13.08%	13.56%	13.55%	13.48%	13.40%
Average memory utilization	4.49%	4.10%	2.57%	1.28%	7.30E-03%
Average network utilization	5.70E-12%	4.76E-10%	2.94E-10%	1.45E-10%	8.16E-11%
Average storage utilization over active servers	3%	3.11%	3.11%	3.09%	3.07%
Average storage utilization	1.03%	9.41E-03%	5.88E-03%	2.91E-03%	1.66E-03%
Average accelerator utilization over active servers	11.92%	9.29%	8.98%	8.93%	8.86%
Average accelerator utilization	4%	2.66%	1.60%	7.97E-03%	4.55E-03%
Energy consumption (kWh) over number of accepted tasks	0.890	1.129	1.802	3.581	6.233

TABLE 19 Scalability of the SOSM cloud by increasing the number of cells with constant number of incoming tasks

Number of cells	20	30	50	100	175
Total energy consumption (MWh)	20,261.10	30,010.91	49,517.50	98,282.09	171,425.92
Total number of submitted tasks	27,736,626	27,633,592	27,690,758	27,685,106	27,545,719
Total number of accepted tasks	27,736,626	27,633,592	27,690,758	27,685,106	27,545,719
Total number of rejected tasks	0	0	0	0	0
Average processor utilization over active servers	45.10%	44.85%	44.51%	44%	43.32%
Average processor utilization	8.97%	5.96%	3.58%	1.79%	1.01%
Average memory utilization over active servers	5.51%	5.48%	5.44%	5.37%	5.29%
Average memory utilization	1.10%	7.29E-03%	4.37E-03%	2.18E-03%	1.23E-03%
Average network utilization	5.58E-12%	3.71E-10%	2.23E-10%	1.11E-10%	6.33E-11%
Average storage utilization over active servers	1.59%	1.58%	1.57%	1.55%	1.53%
Average storage utilization	3.17E-05%	2.11E-03%	1.26E-03%	6.29E-04%	3.56E-04%
Average accelerator utilization over active servers	39.73%	39.51%	39.21%	38.75%	38.16%
Average accelerator utilization	7.91%	5.25%	3.15%	1.57%	8.90E-03%
Energy consumption (kWh) over number of accepted tasks	0.730	1.086	1.788	3.550	6.223

By keeping the number of incoming tasks constant while increasing the number of hardware resources, the cloud capacity significantly increases and is able to serve a larger number of tasks. For this reason, in both cases, the number of rejected tasks is equal to zero for more than 50 cells in the traditional cloud, and in all cases of the SOSM cloud. Even in these cases, the SOSM allocation mechanism utilizes more energy efficient accelerators, leading to lower power consumption.

Further experiments were conducted with increased the number of incoming tasks as the number of cells increases. The results are depicted in Tables 20 and 21, and graphically presented in Figure 5. The percentage of the accepted tasks are given in Table 22.

TABLE 20 Scalability of the traditional cloud by increasing the number of cells and number of incoming tasks

Number of cells	13	20	30	50	100
Number of incoming tasks per second	640	1280	2560	5120	10,240
Total energy consumption (MWh)	13,422.20	20,826.41	31,620.36	53,211.10	106,423.89
Total number of submitted tasks	27,564,857	55,274,081	110,512,844	221,484,764	443,100,234
Total number of accepted tasks	19,873,184	37,076,011	69,492,075	134,515,623	269,139,714
Total number of rejected tasks	7,691,673	18,198,070	41,020,769	86,969,141	173,960,520
Average processor utilization over active servers	69.26%	66.76%	63.89%	62.02%	62.01%
Average processor utilization	27.44%	29.44%	32.29%	34.57%	34.58%
Average memory utilization over active servers	12.42%	11.83%	11.13%	10.64%	10.64%
Average memory utilization	4.94%	5.23%	5.64%	5.94%	5.94%
Average network utilization	7.07E-12%	8.35E-10%	1.02E-09%	1.17E-09%	1.17E-09%
Average storage utilization over active servers	2.87%	2.76%	2.63%	2.54%	2.54%
Average storage utilization	1.15%	1.23%	1.35%	1.44%	1.44%
Average accelerator utilization over active servers	15.54%	18.13%	21.06%	22.94%	22.95%
Average accelerator utilization	6.13%	7.98%	10.64%	12.79%	12.79%
Energy consumption (kWh) over number of accepted tasks	0.675	0.562	0.455	0.396	0.395

TABLE 21 Scalability of the SOSM cloud by increasing the number of cells and number of incoming tasks

Number of cells	13	20	30	50	100
Number of incoming tasks per second	640	1280	2560	5120	10,240
Total energy consumption (MWh)	13,432.93	20,983.74	32,141.70	54,462.77	108,880.32
Total number of submitted tasks	27,650,718	55,352,597	110,647,861	221,536,903	441,190,138
Total number of accepted tasks	27,650,683	54,202,458	104,733,213	205,061,108	408,559,835
Total number of rejected tasks	35	1,150,139	5,914,648	16,475,795	32,630,303
Average processor utilization over active servers	45.34%	47.41%	50.80%	52.52%	52.60%
Average processor utilization	13.78%	16.52%	19.80%	22.40%	22.34%
Average memory utilization over active servers	5.54%	5.96%	6.71%	7.17%	7.17%
Average memory utilization	1.69%	2.08%	2.62%	3.06%	3.05%
Average network utilization	8.56E-10%	9.79E-10%	1.04E-09%	1.06E-09%	1.06E-09%
Average storage utilization over active servers	1.60%	1.68%	1.82%	1.92%	1.91%
Average storage utilization	0.49%	5.89E-03%	7.13E-03%	8.25E-03%	8.20E-03%
Average accelerator utilization over active servers	39.94%	42.02%	45.42%	47.19%	47.26%
Average accelerator utilization	12.14%	14.65%	17.70%	20.12%	20.08%
Energy consumption (kWh) over number of accepted tasks	0.486	0.387	0.307	0.266	0.266

SOSM proved to be more efficient in all metrics and able to handle hardware resources and incoming tasks more efficiently. With the same energy, SOSM is able to serve 52.46% (for the case of 100 cells) more tasks than the traditional resource allocation system. This is mainly due to fact that 56.99% (for the case of 100 cells) more accelerators have been used during the SOSM simulation.

6 | CONCLUSIONS

A newly introduced resource allocation scheme, namely SOSM, targeting hyper-scale cloud infrastructures was recently proposed. For examining its efficiency, in various levels, an evaluation framework was set up. This included the development of a new parallel cloud simulation framework and use cases that can exploit the new architecture and demonstrate the framework's efficiency.

This article was focused on the set up of this evaluation framework including: (a) the validation of the cloud simulation framework itself, (b) the definition of the evaluation criteria and the use cases input parameters, and (c) the derivation of evaluation results by direct comparison of SOSM with the traditional centralized cloud resource allocation schemes with the use of (a) and (b). The obtained numerical results demonstrate the dominance of SOSM in all evaluation criteria, and especially in large number of simulated cloud nodes. The simulations were able to scale up to 10,000,000 heterogeneous cloud nodes, validating the scalability not only of the simulation platform but also of the SOSM resource allocation scheme. SOSM was proved to be more efficient in all defined criteria, and able to handle hardware resources and incoming tasks more efficiently. By utilizing the same energy, SOSM was able to serve about 52% (for the case of 100 cells) more tasks than the traditional resource allocation system. The efficiency, with respect to all evaluation criteria, of the SOSM against the centralized cloud is increasing as the number of simulated heterogeneous nodes increases. This is due to the higher allocation of accelerators for fulfilling the incoming tasks that not only reduces power consumption but also releases peripheral resources (such as memory, network, and storage) since tasks are executed in a shorter time frame. Further work includes the comparison with more sophisticated resource allocation schemes for the traditional centralized cloud, as well as experimentation with various application characteristics and different SOSM assessment functions.

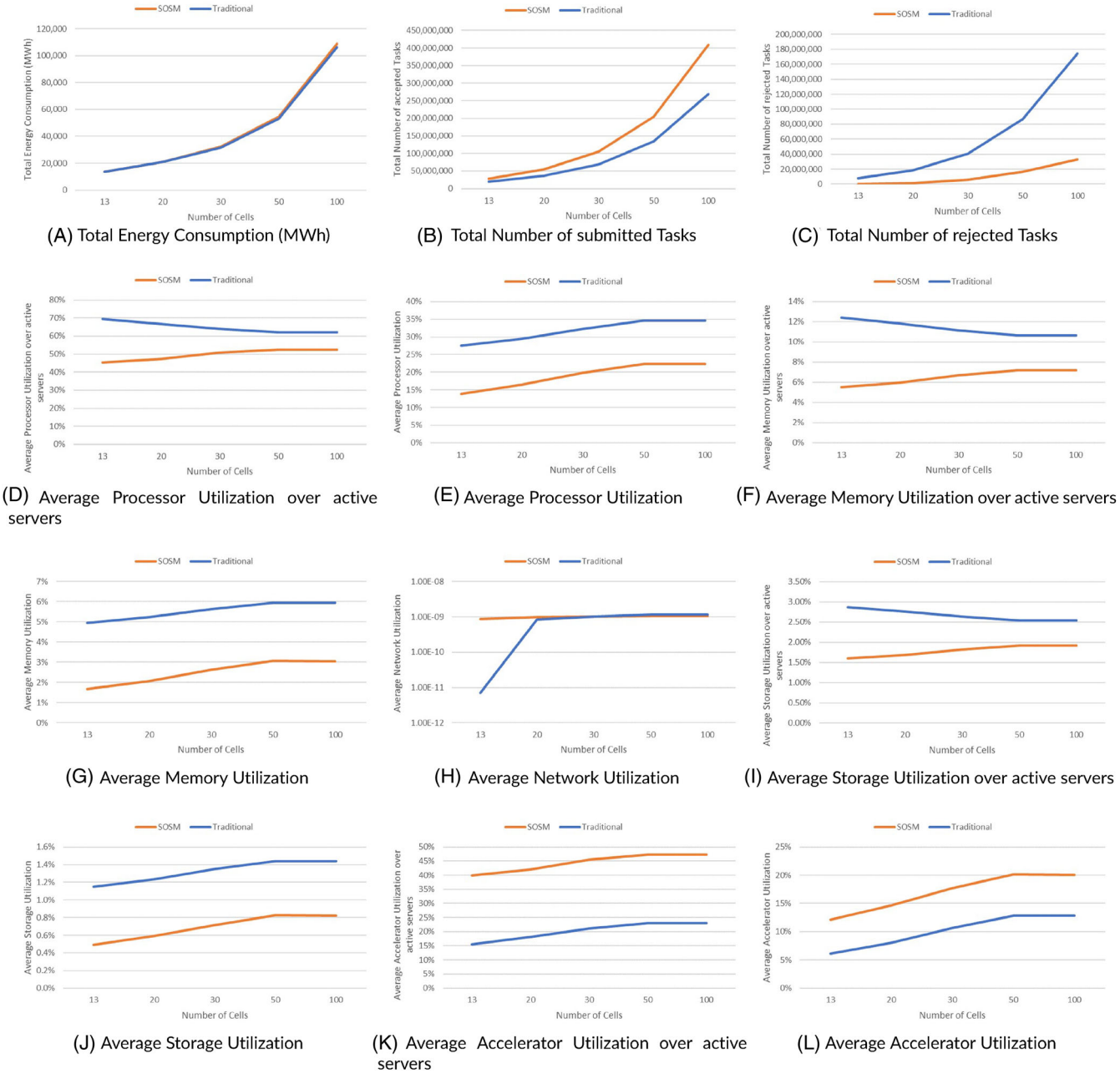


FIGURE 5 Scalability of the traditional and SOSM cloud by increasing the number of cells and number of incoming tasks

TABLE 22 The percentage of accepted tasks for traditional centralized and SOSM clouds while increasing the number of cells.

	Number of cells				
Resource allocation	**13**	**20**	**30**	**50**	**100**
Traditional	72.096%	67.077%	62.881%	60.734%	60.740%
SOSM	100.000%	97.922%	94.655%	92.563%	92.604%

ACKNOWLEDGMENTS

This work has been partially funded by the European Union's Horizon 2020 Research and Innovation Programme through CloudLightning project (http://www.cloudlightning.eu) under Grant Agreement No. 643946. The authors would like to thank Perumal Kuppuudaiyar (Intel, Leixlip, Ireland), Tobias Becker (Maxeler Technologies, London, UK), and Malik Khan (Department of Computer Science at Norwegian University of Science and Technology, Trondheim, Norway) for providing execution traces for the application use cases on the heterogeneous hardware, and Antonios Makaratzis

for collecting simulation execution data. Finally, the authors acknowledge the Greek Research and Technology Network (GRNET) for the provision of the National HPC facility ARIS under Project PR004033-ScaleSciCompII and PR006053-ScaleSciCompIII.

ORCID

Konstantinos M. Giannoutakis https://orcid.org/0000-0001-8939-4912

REFERENCES

1. Panda DK, Lu X. HPC meets cloud: building efficient clouds for HPC, big data, and deep learning middleware and applications. Paper presented at: Proceedings of the 10th International Conference on Utility and Cloud Computing UCC '17; 2017:189-190, Austin, Texas; Association for Computing Machinery.
2. Netto MAS, Calheiros RN, Rodrigues ER, Cunha RLF, Buyya R. HPC cloud for scientific and business applications: taxonomy, vision, and research challenges. *ACM Comput Surv*. 2018;51(1):1–29. https://doi.org/10.1145/3150224.
3. Malla S, Christensen K. HPC in the cloud: performance comparison of function as a service (FaaS) vs infrastructure as a service (IaaS). *Internet Technol Lett*. 2020;3(1):e137. https://doi.org/10.1002/itl2.137.
4. Lynn T. Addressing the complexity of HPC in the cloud: emergence, self-organisation, self-management, and the separation of concerns. *Heterogeneity, High Performance Computing, Self-Organization and the Cloud*. Cham, Switzerland: Springer International Publishing; 2018:1-30.
5. Zhang Q, Cheng L, Boutaba R. Cloud computing: state-of-the-art and research challenges. *J Internet Serv Appl*. 2010;1(1):7-18.
6. Xiong H, Filelis-Papadopoulos C, Castañe GG, Dong D, Morrison JP. Self-organising, self-managing frameworks and strategies. *Heterogeneity, High Performance Computing, Self-Organization and the Cloud 2018*. Cham, Switzerland: Palgrave Macmillan; 2018:63-88.
7. Filelis-Papadopoulos C, Xiong H, Spataru A, et al. A generic framework supporting self-organisation and self-management in hierarchical systems. Paper presented at: Proceedings of the 2017 16th International Symposium on Parallel and Distributed Computing (ISPDC 2017), Innsbruck, Austria: IEEE; July 3, 2017:149-156.
8. Khan M, Becker T, Kuppuudaiyar P, Elster AC. Container-based virtualization for heterogeneous HPC clouds: insights from the EU H2020 CloudLightning project. Paper presented at: Proceedings of the 2018 IEEE International Conference on Cloud Engineering (IC2E 2018), Orlando, Florida: IEEE; April 17, 2018:392-397.
9. Filelis-Papadopoulos CK, Giannoutakis KM, Gravvanis GA, Tzovaras D. Large-scale simulation of a self-organizing self-management cloud computing framework. *J Supercomput*. 2018;74(2):530-550. https://doi.org/10.1007/s11227-017-2143-2.
10. Filelis-Papadopoulos CK, Gravvanis GA, Kyziropoulos PE. A framework for simulating large scale cloud infrastructures. *Future Generat Comput Syst*. 2018;79:703-714. https://doi.org/10.1016/j.future.2017.06.017.
11. Barroso LA, Clidaras J, Holzle U. The datacenter as a computer: an introduction to the design of warehouse-scale machines. *Synth Lect Comput Archit*. 2013;8(3):1-154.
12. Byrne J, Svorobej S, Giannoutakis KM, et al. A review of cloud computing simulation platforms and related environments. Paper presented at: Proceedings of the 7th International Conference on Cloud Computing and Services Science, Porto, Portugal; 2017:651-663.
13. Lynn T, Gourinovitch A, Byrne J, et al. A preliminary systematic review of computer science literature on cloud computing research using open source simulation platforms. Paper presented at: Proceedings of the 7th International Conference on Cloud Computing and Services Science, Porto, Portugal; 2017:537-545.
14. Makaratzis AT, Giannoutakis KM, Tzovaras D. Energy modeling in cloud simulation frameworks. *Future Generat Comput Syst*. 2017;79:715–725. https://doi.org/10.1016/j.future.2017.06.016.
15. Filelis-Papadopoulos CK, Gravvanis GA, Morrison JP. CloudLightning simulation and evaluation roadmap. *CloudNG '17*. New York, NY: ACM; 2017:2:1-2:6.
16. Giannoutakis KM, Makaratzis AT, Tzovaras D, Filelis-Papadopoulos CK, Gravvanis GA. On the power consumption modeling for the simulation of heterogeneous HPC clouds. *CloudNG '17*. New York, NY: ACM; 2017:1:1-1:6.
17. Makaratzis AT, Khan MM, Giannoutakis KM, Elster AC, Tzovaras D. GPU power modeling of HPC applications for the simulation of heterogeneous clouds. In: Wyrzykowski R, Dongarra J, Deelman E, Karczewski K, eds. *Parallel Processing and Applied Mathematics*. Cham, Switzerland: Springer International Publishing; 2018:91-101.
18. SPEC Standard performance evaluation corporation, server power and performance characteristics; 2008. http://www.spec.org/power_ssj2008/. Accessed date 08, April 2021.
19. Dell PowerEdge C4130 data sheet. https://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Dell-PowerEdge-C4130-Spec-Sheet.pdf. Accessed date 08, April 2021.
20. Nvidia Tesla P100 data sheet. https://images.nvidia.com/content/tesla/pdf/nvidia-tesla-p100-PCIe-datasheet.pdf. Accessed date 08, April 2021.
21. Intel Xeon Phi 5110P data sheet. https://www.intel.com/content/dam/www/public/us/en/documents/datasheets/xeon-phi-coprocessor-datasheet.pdf. Accessed date 08, April 2021.
22. Alfieri R, Brambilla M, De Pietri R, et al. *The COKA Project. INFN-CNAF Annual Report ISSN 2283-5490*. Italy: Universita di Parma and INFN-Gruppo collegato di Parma; 2014.

How to cite this article: Giannoutakis KM, Filelis-Papadopoulos CK, Gravvanis GA, Tzovaras D. Evaluation of self-organizing and self-managing heterogeneous high performance computing clouds through discrete-time simulation. *Concurrency Computat Pract Exper*. 2021;e6326. https://doi.org/10.1002/cpe.6326