computers

MDPI

Article

On the Optimization of Self-Organization and Self-Management Hardware Resource Allocation for Heterogeneous Clouds

Konstantinos M. Giannoutakis [1,2,*], Christos K. Filelis-Papadopoulos [3] and George A. Gravvanis [4] and Dimitrios Tzovaras [1]

[1] Information Technologies Institute, Centre for Research and Technology Hellas, 57001 Thessaloniki, Greece; Dimitrios.Tzovaras@iti.gr
[2] Department of Applied Informatics, School of Information Sciences, University of Macedonia, 54636 Thessaloniki, Greece
[3] Department of Computer Science, University College Cork, T12 XF62 Cork, Ireland; christos.papadopoulos-filelis@cs.ucc.ie
[4] Department of Electrical and Computer Engineering, Democritus University of Thrace, 67100 Xanthi, Greece; ggravvan@ee.duth.gr
* Correspondence: kgiannou@iti.gr or kgiannou@uom.edu.gr

Abstract: There is a tendency, during the last years, to migrate from the traditional homogeneous clouds and centralized provisioning of resources to heterogeneous clouds with specialized hardware governed in a distributed and autonomous manner. The CloudLightning architecture proposed recently introduced a dynamic way to provision heterogeneous cloud resources, by shifting the selection of underlying resources from the end-user to the system in an efficient way. In this work, an optimized Suitability Index and assessment function are proposed, along with their theoretical analysis, for improving the computational efficiency, energy consumption, service delivery and scalability of the distributed orchestration. The effectiveness of the proposed scheme is being evaluated with the use of simulation, by comparing the optimized methods with the original approach and the traditional centralized resource management, on real and synthetic High Performance Computing applications. Finally, numerical results are presented and discussed regarding the improvements over the defined evaluation criteria.

Keywords: simulation; optimization; high performance computing; heterogeneity; cloud computing

Citation: Giannoutakis, K.M.; Filelis-Papadopoulos, C.K.; Gravvanis, G.A.; Tzovaras, D. On the Optimization of Self-Organization and Self-Management Hardware Resource Allocation for Heterogeneous Clouds. *Computers* **2021**, *10*, 147. https://doi.org/10.3390/computers10110147

Academic Editor: George K. Adam

Received: 30 September 2021
Accepted: 4 November 2021
Published: 9 November 2021

1. Introduction

Cloud Computing has evolved into the main computing paradigm nowadays, usually coupled with edge and fog layers formulating the Edge to Cloud Continuum (E2C) [1]. The need for edge, fog and cloud computing paradigms relies on the fact that the processing of big data produced from various application domains caused a complexity explosion and pushed existing infrastructures to the limits. The capabilities and characteristics of this distributed and heterogeneous paradigm enabled the efficient deployment of real-world applications, such as smart grids [2], predictive maintenance [3], connected and autonomous vehicles [4], augmented/virtual reality [5], smart cities [6] and smart factories [7], among others. Typically, data are being generated and filtered/pre-processed on the edge layer, while the fog layer aggregates and further processes data stemming from distributed edge devices. Then, the cloud layer is subject to performing computationally intensive operations and storing of large amount of aggregated data, for supporting such complex applications.

Due to the constantly increasing demand of computational power and, consequently, of computational resources, cloud data centers have integrated hardware accelerators capable of performing computations efficiently with reduced energy requirements. This

specialized hardware, originally stemming from High Performance Computing (HPC) infrastructures, may contain Graphics Processing Units (GPUs), Field Programmable Gate Arrays (FPGAs) and Many Integrated Cores (MICs) in a heterogeneous manner (at the level of computing racks). Nowadays, almost all cloud providers offer many types of accelerator-enabled hardware instances for the execution of computationally demanding applications.

The imposed heterogeneity in the hardware resources installed on large-scale cloud data centers adds an additional layer of complexity for the efficient management of the underlying resources, as well as reliability and security challenges [8]. The efficient management of the hardware resources is essential, since it determines how efficiently the resources are being utilized with respect to Quality of Service (QoS) and energy efficiency.

In the traditional centralized cloud management, the end-user is responsible for selecting the resources to be utilized for the execution of the application and the cloud broker offloads the executables on the available virtualized hardware. Several approaches have been proposed for the dynamic allocation of resources based on application characteristics, such as techniques based on machine learning or optimization. Autonomic computing approaches, offer promising solutions with the absence of external control or interference from outside the boundaries of the system [9–12].

Recently, through the CloudLightning (CL) EU project, a Self-Organization and Self-Management (SOSM) resource allocation scheme has been proposed, targeting improved service delivery, computational efficiency, power consumption, scalability and management of heterogeneous cloud resources [13]. This scheme allows the decentralized and dynamic provisioning of heterogeneous resources towards local or global goals. For achieving this, the Suitability Index (SI) term was introduced for characterising in real-time how suitable the hardware is for fulfilling the incoming task requests. SI has been formulated according to the local and global goals of each cloud infrastructure, that is, reduce energy consumption, improve computational efficiency and so forth [14].

In this work, an improved assessment function is proposed for formulating the Suitability Index of heterogeneous clouds, by taking into account the performance per unit of consumed power. The new function is expected to provision resources considering the efficiency of the hardware to perform computations with the lowest energy footprint in the hierarchical CloudLightning architecture. The theoretical formulation of the Assessment Function is described in detail, enabling the definition of a new Suitability Index for guiding the task over the most efficient, in terms of energy resources. For evaluating the efficiency of the new proposed schemes, a discrete-event simulation framework is utilized that allows the efficient experimentation at large-scale. The evaluation framework is being presented, and numerical results, using real-world application traces and synthetic data, indicate that the improved SOSM allocation scheme can provide better energy efficiency by retaining the same levels of computational efficiency, service delivery and scalability.

The remainder of this paper is organised as follows. Section 2 presents the architectural overview of the heterogeneous cloud system considered and the simulation framework utilized for the evaluation of the proposed schemes at a large-scale. Section 3 introduces the theoretical background for the derivation of the improved assessment function and Suitability Index, while Section 4 presents the evaluation framework and results of the proposed schemes. This includes numerical results from real-world applications along with synthetic ones at a large scale. Concluding remarks and discussion are then presented in Section 5.

2. Background

This Section delineates the research area of resource management in cloud computing and presents the CloudLightning architecture along with the Self-Organization and Self-Management resource allocation schemes. The discrete-event simulation framework utilized for the large-scale evaluation of the proposed schemes is also briefly presented.

2.1. Resource Management in Cloud Computing

The categorization of resource allocation techniques has been the main research effort of many studies [15–17]. Different taxonomies have been proposed, with respect to scheduling criteria, specifically designed for specific purposes such as cost, energy consumption, utilization and SLA violations. In general, resource allocation and the scheduling of tasks and hardware resources can be considered an optimization problem, whose complexity increases as the number of cloud nodes increases. By also adopting heterogeneity and specialized hardware (accelerators) for supporting computations, the efficient scheduling of resources is very important. Resource allocation research has been mainly based on simulation frameworks that provide artificial infrastructures to assess resource management scenarios, while also enabling resource modelling.

The research effort for addressing the challenges of resource management of cloud computing is increasing. Agent-based solutions have been proposed in the literature [18], that aim to develop an explicit resource allocation model to adaptively distribute workflow jobs to the different data centers in the cloud environment, based on network delay and workload evaluation criteria. The approach of [19] proposes an Autonomous Agent Based Load Balancing Algorithm (A2LB) to address the common problem of load balancing in cloud computing data centers, by dynamically reallocating the virtual machines from one data center to another when overloading is detected.

Machine learning algorithms have also been proposed to model and analyze the multi-dimensional resource allocation problems that deal with multiple resource types including multi-processor, memory and storage [20]. A Reinforcement Learning (RL) based allocation framework has been proposed [21] to develop an optimal allocation policy and employ Fuzzy Logic for the design of a decision support methodology based on the criterion of energy efficiency in the cloud infrastructure. Deep Reinforcement Learning (DRL) has been proposed for developing DERP [22], a resource provisioning system that combines the capability of Deep Learning (DL) to learn multi-dimensional representations of the states of the cloud computing system, with the dynamic adaptation capability of RL to workload demand changes, with the aim of finding the optimal resource management policy. A hierarchical DRL approach [23] is used that deals with the minimization of power consumption while controlling performance degradation by finding an effective dynamic power management policy. The work of [24] investigates DRL-based resource allocation combining the Deep Deterministic Policy Gradient (DDPG) method with Long Short-Term memory (LSTM) network units, towards maximizing the cloud provider's profit and evaluate the methodology on real-world datasets addressing both resource allocation and pricing on the online user arrival time-variant patterns.

Other research works focused mainly on applying optimization based techniques for addressing the resource allocation challenges. Such techniques include the use of non dominated sorting genetic algorithms [25] and artificial bee colonies with a crossover mutation genetic algorithm [26], amongst others. The list of research works is not exhaustive, but are characteristic of how optimization algorithms can support this challenge. A review of optimization-based resource scheduling algorithms is given by [27].

The research works presented consider the centralized management of resources, where an entity, usually called a broker, is responsible for allocating the resources according to the corresponding technique implemented. The limitations of traditional centralized management lead to increased deployment and discovery times due to the increased search space of the hardware resources, especially for large-scale hardware deployments. Decentralization on hardware orchestration has recently been introduced [13] through a hierarchical architecture that enables the distributed management of the underlying resources. Each decentralized entity does not need to obtain knowledge about the status of the whole data center, and instead utilizes aggregate information from other resources. This substantially reduces the amount of transmitted information about the state of the resources, and allows the cloud provider to set global and local goals to the underlying

resources. This framework has been designed and evaluated through simulation at a large-scale and is briefly discussed in Sections 2.2 and 2.3 respectively.

2.2. Resource Management with Self-Organization and Self-Management Strategies

Self-Organization and Self-Management (SOSM) strategies, targeting the optimal hardware resource allocation taking into consideration the computational efficiency, energy consumption, service delivery and scalability, have recently been proposed [13,28]. The developed framework is applicable to cloud architectures that follow the Warehouse Scale Computer (WSC) architectural model [29]. In this model, the cloud computing nodes are grouped into cells and are managed by the cell brokers. Each cell can contain various homogeneous racks of computational servers. Heterogeneity is realized in the rack level, where cells may contain racks of different CPU types or CPU–accelerators pairs [29,30]. The abstract architecture of the Warehouse Scale Computer model is depicted in Figure 1.

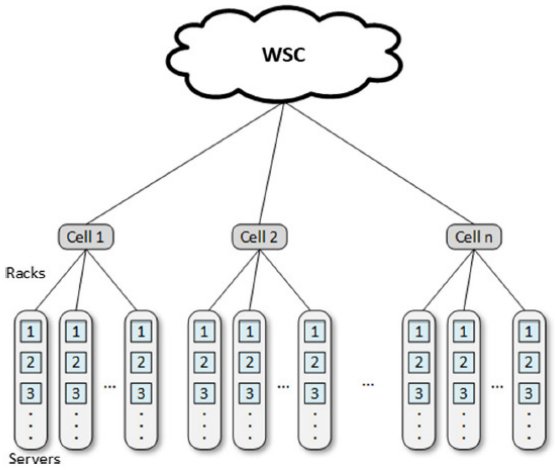

Figure 1. Warehouse scale computer abstract architecture [30].

The SOSM framework was developed and deployed on the CloudLightning (CL) system, as illustrated in Figure 2, and consists of three virtual hierarchical levels for managing the underlying resources.

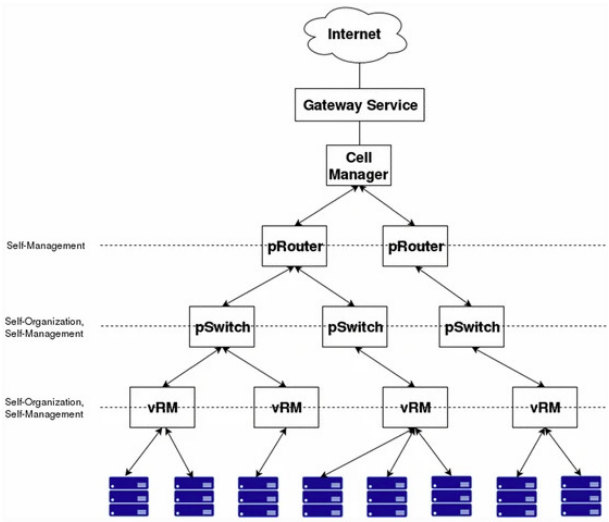

Figure 2. Abstract architecture of the CloudLightning system [31,32].

This tree-structured virtual consideration of the architecture enables the efficient management of the heterogeneous resources. The root management entity of this architecture

is the cell manager, which is responsible for the management of the pRouters. There are different pRouters according to the distinct hardware resource type. pSwitches partition the Virtual Rack Managers (vRMs), managing the same resource type into groups. The number of pSwitches per pRouter and the size of the vRMs managed by each pSwitch can be changed over time. In order to minimise potential administrative overheads and to simplify the coalition formation process for serving tasks, vRMs cannot span pSwitches. As the system evolves, the number of pSwitches connected to a pRouter and the number of vRMs connected to a pSwitch will change and will converge to some optimal number according to the global or local goals defined by the Suitability Index (SI).

The assessment functions defined in the CL architecture mathematically express the state of the cloud components by considering functional and non-functional characteristics, namely: task throughput, energy efficiency, computational efficiency, resource management efficiency and memory consumption. The formulation and description of the assessment functions have been proposed in [31]. Additionally, any component in the architecture is associated with a Suitability Index, a metric for expressing how suitable the component is for contributing to the global goal of the system. SI can be expressed as a weighted sum of the assessment functions, with weights defined by the cloud providers representing the global goals.

2.3. Simulation Framework

For examining the effectiveness of the traditional centralized and SOSM clouds at a large scale, a software simulation has been adopted. Despite the fact that a variety of simulation frameworks already exists [33,34], their scalability and support for heteroneneous hardware is limited [32]. For this reason, the discrete-time CloudLightning Simulation Framework has been designed and developed, allowing the experimentation of dynamic resource allocation schemes along with the traditional centralized management of resources.

The main difference between the two provisioning methods is that in the traditional cloud the end-user is responsible for selecting the application and its implementation that will be deployed and executed, while in the SOSM cloud, the user is responsible for selecting the type of application and then the SOSM mechanism selects the most appropriate implementation according to the status of the system. This was simulated for the traditional cloud by randomly generating application implementations that serve as user inputs, and for the SOSM cloud by randomly generating applications. In both cases, the uniform random distribution was used for the generation of simulation inputs. For the traditional cloud, a sequential search is being performed for the first available resources that can serve an incoming application implementation, while for the SOSM cloud, the selection of resources for serving each incoming application depend on the Suitability Index of each Cell per time step. These algorithmic procedures have been proposed in [30,32].

The simulation framework has been implemented in C++, coupled with the Message Passing Interface (MPI) and OpenMP, for the parallelization of the workload in multi-computer and multi-processor modern computing infrastructures. The framework is extensible, allowing a variety of resource allocation schemes and models to be adopted. It has been experimentally proven that it can achieve the same level of accuracy as most of the well-known simulation frameworks in the literature, while achieving high levels of scalability by simulating millions of cloud nodes [32]. The CloudLightning simulation platform is open-sourced at https://bitbucket.org/cloudlightning/cloudlightning-simulator (accessed on 3 November 2021).

3. Improved Assessment Function and Suitability Index

The motivation for proposing an improved assessment function was derived from the need to reflect the capability of the cloud resources to perform computations efficiently with the minimum energy consumption. This section presents the mathematical formulation of the new assessment function that leads to a new Suitability Index for the allocation of

heterogeneous cloud resources. Table 1 summarizes the mathematical notations utilized in this section.

Table 1. Mathematical notations utilized for the formulation of the improved Assessment Function and SI.

Notation	Description
$C_{pRouter}$	Total MIPS of a pRouter
C_{vCore}	Computational capability of a computational virtual core in MIPS
N_{vCores}	Total number of virtual cores of all servers hosted under a pRouter
C_{acc}	The computational capability of an accelerator in MIPS
N_{acc}	The number of accelerators of all hosted under a pRouter
$P_{pRouter}$	The power consumption of all servers hosted by a pRouter
P_{vCores}^{idle}	The idle power consumption of a virtual core
P_{vCores}^{max}	The maximum power consumption of a virtual core
P_{acc}^{idle}	The idle power consumption of an accelerator
P_{acc}^{max}	The maximum power consumption of an accelerator
N_{vCores}^{avail}	The available virtual cores
N_{acc}^{avail}	The available accelerators
f_1	The assessment function describing performance per Watt
SI	The Suitability Index
$\delta N_{vCores}^{avail}$	The number of virtual cores required by an incoming task
δN_{acc}^{avail}	The number of accelerators required by an incoming task
T_{cpu}	The performance of a CPU based implementation
T_{acc}	The performance of the accelerator based implementation
E_{cpu}	The energy consumption of a CPU based implementation
E_{acc}	The energy consumption of an accelerator based implementation
$P_{cpu}(t)$	The power consumption of the CPU based implementation at time t
$P_{acc}(t)$	The power consumption of the accelerator based implementation at time t
$\hat{P}_{cpu}$	The average power consumption of $P_{cpu}(t)$
$\hat{P}_{acc}$	The average power consumption of $P_{acc}(t)$
ψ	The benefit on the execution time of application by utilizing accelerators
N_{Cores}^{r}	The total number of requested cores
N_{acc}^{r}	The total number of requested accelerators

The five assessment functions presented in Section 2 and in [31], can be replaced by a single assessment function depicting the performance per unit of consumed power, for example, Million Instructions per Second (MIPS) per Watt [14]. This can be achieved by calculating the potential performance of all the hardware hosted under a pRouter. The total MIPS $C_{pRouter}$ of a pRouter hosting CPU based servers is computed as follows:

$$C_{pRouter} = C_{vCore} N_{vCores}, \tag{1}$$

where C_{vCore} is the computational capability of a computational core in MIPS and N_{vCores} is the total number of virtual cores of all servers hosted under a pRouter. In the case of pRouters composed of CPU-Accelerator pair based servers, the total MIPS $C_{pRouter}$ of a pRouter are defined as follows:

$$C_{pRouter}(N_{vCores}, N_{acc}) = C_{vCores} N_{vCores} + C_{acc} N_{acc}, \tag{2}$$

where C_{acc} is the computational capability of an accelerator and N_{acc} is the number of accelerators of all hosted by the pRouter. Similarly, the power consumption of all servers hosted by a pRouter can be estimated using a linear model [35], for all the CPU or CPU-Accelerator pair based servers:

$$P_{pRouter}(N_{vCores}, N_{acc}) = P^{idle}_{vCores} N_{vCores} + \left(P^{max}_{vCores} - P^{idle}_{vCores}\right)\left(N_{vCores} - N^{avail}_{vCores}\right) + P^{idle}_{acc} N_{acc} + \left(P^{max}_{acc} - P^{idle}_{acc}\right)\left(N_{acc} - N^{avail}_{acc}\right), \quad (3)$$

where P^{idle}_{vCores}, P^{max}_{vCores} is the idle and maximum power consumption of a virtual core, respectively. These metrics can be computed by dividing the idle and maximum power consumption of a CPU by the total number of virtual cores it can host. The P^{idle}_{acc}, P^{max}_{acc} is the idle and maximum power consumption of an accelerator. The superscript *avail* denotes the available resources.

The assessment function describing performance per Watt is then defined as follows:

$$f_1\left(N^{avail}_{vCores}, N^{avail}_{acc}\right) = \frac{C_{pRouter}}{P_{pRouter}\left(N^{avail}_{vCores}, N^{avail}_{acc}\right)}. \quad (4)$$

The function f_1 is hyperbolic and strictly monotonically decreasing for both input arguments. The Suitability Index has the same value as the assessment function, since $SI = f_1\left(N^{avail}_{vCores}, N^{avail}_{acc}\right)$. In order to compute the change in the Suitability Index when a task passes from a logical component of the hierarchy such as a pRouter, a pSwitch or a vRM the multi-variate Taylor expansion is required, since the assessment functions are always evaluated at the vRM level. The Taylor expansion of Equation (4) is as follows:

$$SI = f_1\left(N^{avail}_{vCores} \pm \delta N^{avail}_{vCores}, N^{avail}_{acc} \pm \delta N^{avail}_{acc}\right) = f_1\left(N^{avail}_{vCores}, N^{avail}_{acc}\right) \pm \delta N^{avail}_{vCores} \frac{\partial f_1\left(N^{avail}_{vCores}, N^{avail}_{acc}\right)}{\partial N^{avail}_{vCores}} \pm \delta N^{avail}_{acc} \frac{\partial f_1\left(N^{avail}_{vCores}, N^{avail}_{acc}\right)}{\partial N^{avail}_{acc}} + \mathcal{O}\left(\left(\delta N^{avail}_{vCores}\right)^2 + \left(\delta N^{avail}_{acc}\right)^2\right), \quad (5)$$

where $\delta N^{avail}_{vCores}$, δN^{avail}_{acc} are the number of virtual cores and accelerators required by a task.

In order to further improve guidance of tasks through the system, the characteristics of the incoming tasks can be taken into account. This is required in cases where the improvement in performance, from the use of accelerators, is less than the ratio of energy consumed. Let us consider the performance of a CPU based implementation T_{cpu} and the performance of the accelerator based implementation T_{acc}, then the energy consumption of the two implementations can be computed as follows:

$$E_{cpu} = \int_0^{T_{cpu}} P_{cpu}(t)dt \approx \hat{P}_{cpu} T_{cpu}, \quad (6)$$

$$E_{acc} = \int_0^{T_{acc}} P_{acc}(t)dt \approx \hat{P}_{acc} T_{acc}, \quad (7)$$

where $P_{cpu}(t)$ is the power consumption corresponding to the CPU based implementation at time t and $P_{acc}(t)$ is the power consumption corresponding to the accelerator based implementation at time t. The notations $\hat{P}_{cpu}$ and $\hat{P}_{acc}$ denote the average of $P_{cpu}(t)$ and $P_{acc}(t)$, respectively. The efficient execution on accelerators requires:

$$E_{acc} \leq E_{cpu} \Leftrightarrow \frac{\hat{P}_{acc}}{\hat{P}_{cpu}} \leq \frac{T_{cpu}}{T_{acc}}. \quad (8)$$

Equation (8) implies that the performance (execution time) is coupled with the relative average power consumption of available implementations of an application. Thus, scheduling applications, favorably, to accelerators as implied from Equation (4), might result in increased power consumption, especially if the improvement in performance does not satisfy Equation (8). The choice of hardware is performed on the Cell Manager level, where the appropriate pRouter is chosen based on the Suitability Index. The Suitability Index corresponding to hardware hosting accelerators can be modified using task information. The SI of hardware hosting accelerators is reduced temporarily based on information derived from incoming tasks. After the reduction, the hardware type with the maximum SI is chosen. The benefit on the performance of application by utilizing accelerators, can be computed as follows:

$$\psi = \frac{C_{vCores} N^r_{Cores} + \rho C_{acc} N^r_{acc}}{C_{vCores} N^r_{Cores} + C_{acc} N^r_{acc}}, \quad (9)$$

where N^r_{cores} and N^r_{acc} are the total number of requested cores and the total number of required accelerators, respectively. The ρ parameter denotes the percentage of an application that is parallelized and executed on an accelerator [35,36]. When the full potential of an accelerator is used then $\rho = 1$, while when no accelerator is used, $\rho = 0$. The mean value of the parameter ρ can be computed as the ratio of the time that an accelerator is utilized over the total execution time of the application. Using the mean value of ρ, the energy consumption of the accelerator for an application can be computed ($E_{acc} = (t_2 - t_1) \cdot P_{acc}(\rho)$ for a time interval t_1 to t_2). Values of ψ close to 1 denote a high utilization of the accelerator resources, while values close to 0 lead to under-utilization of the accelerators without a significant reduction in processing. In the case where there are no accelerators, then $\psi = 1$. It should be noted that the formulation of Equation (9) does not consider the effect of ρ on CPUs, since the developed framework concerns scientific computing applications where CPUs are occupied by managing operations related to the accelerators, such as memory transfers, along with allocated computational work.

4. Numerical Results

For the evaluation of the proposed optimized scheme of assessment functions, an evaluation framework has been setup, utilizing the simulation framework of Section 2 in order to be able to scale up to millions of cloud nodes. The behavior of the new scheme over specific criteria has been evaluated against the traditional centralized resource allocation scheme and SOSM, by modeling three real-world HPC use cases and by synthetic HPC applications. It should be noted that the evaluation of the SOSM resource allocation scheme over the traditional one has been extensively presented in [32].

4.1. Evaluation Framework and Setup

For the direct comparison of the new proposed scheme, the evaluation framework and setup utilized in [32] was also adopted in this study. This allows the identification of potential improvements and optimizations concerning the evaluation criteria defined.

4.2. Real-World HPC Applications

The first set of simulation results was obtained from the modeling of three real-world HPC applications: (i) Oil and Gas exploration, (ii) Ray Tracing, and (iii) Genomics, that were executed, except from conventional CPUs, on GPU, MIC and FPGA environments, respectively [37]. The characteristics of the simulated applications were extracted from real execution traces on the various supported hardware platforms, and can be summarized in Table 2. Each instance contains a CPU-only and a CPU-accelerator implementation, where in the case of SOSM, it is selected according to the corresponding resource allocation strategy and the available hardware resources [32].

In terms of the simulated hardware, it consisted of 13 data centers, each of them containing 100,000 servers equally distributed to the different hardware types supported (i.e., 25,000 servers per CPU, CPU + GPU, CPU + MIC or CPU + FPGA configuration). The

hardware that has been modeled is summarized in Table 3. The distribution of incoming simulated tasks was uniform and the simulation time was one day. Finally, the network bandwidth was set to 20 Gbps.

Table 2. Simulated applications characteristics.

Instance No.	Available Implementations	Supported vCPUs	Required Memory (GB)	Required 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

Table 3. Heterogeneous hardware characteristics simulated for the real-world HPC applications.

Hardware Type	Model	Characteristics	MIPS
CPU	Dell PowerEdge C4120 [38]	2 Intel Xeon E5-2630 v4 Processors (20 cores, 40 threads, 2.20 GHz), HT disabled, 128GB RAM, 1,024GB HDD	88,000.80 MIPS
GPU	Nvidia Tesla P100 [39]	3584 cores	587,505.34 MIPS
MIC	Intel Xeon Phi 5110P [40]	4 Xeon Phi cards	507,494.40 MIPS
FPGA	MPC-X [41]	8 MAX4 cards	295,959.30 MAOPS (similarly treated to MIPS for simplicity)

The simulation results of the improved SOSM for 13 Cells and for various numbers of incoming tasks are given in Table 4. These results contain the metrics extracted from the simulation framework considering the total number of submitted, accepted and rejected tasks, the average utilization of the main hardware components, and the total energy consumption for each execution. It should be noted that the utilization metrics are computed both for the whole simulated infrastructure and for the servers that are not idle (using the "over active servers" notation).

Compared to the corresponding results for traditional centralized and SOSM clouds [32], there is a small improvement in the total energy consumption and the number of accepted tasks, especially for the case of 640 incoming tasks per second. In this case, improved SOSM managed to serve all incoming tasks (against SOSM that rejected 35 tasks [32]), while the total energy consumption was slightly reduced by 3.76 MWh. The rest of the metrics were kept at the same levels.

The optimized version of the SOSM implementation was also tested while increasing the number of cloud nodes and the number of incoming tasks per second in order to stress the system and evaluate its behavior in extreme conditions. This allowed us to examine the behavior of the two resource allocation schemes at a large scale (up to 100 cells yielding total 10,000,000 simulated servers) and a large number of incoming tasks per second (up to 10,240 tasks per second for 100 cells). The results are depicted in Table 5, while the percentage of accepted tasks over SOSM is shown in Table 6 and the energy consumption

(in KWh) over the number of accepted tasks is shown in Table 7. The corresponding diagrams, showing the improvement over SOSM, are depicted in Figure 3.

Table 4. Simulation results with the improved SOSM index for 13 Cells.

	Maximum Tasks per Second						
	20	**40**	**60**	**80**	**160**	**320**	**640**
Total Energy Consumption (MWh)	12,702.60	12,726.03	12,749.24	12,772.45	12,866.64	13,054.60	13,429.17
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	27,650,718
Total Number of rejected Tasks	0	0	0	0	0	0	0
Average Processor Utilization over active servers	42%	43.30%	43.69%	43.95%	44.46%	44.86%	45.21%
Average Processor Utilization	4.27×10^{-3}%	8.57×10^{-3}%	1.28%	1.71%	3.43%	6.90%	13.84%
Average Memory Utilization over active servers	5.08%	5.28%	5.33%	5.36%	5.43%	5.48%	5.52%
Average Memory Utilization	5.17×10^{-4}%	1.05×10^{-3}%	1.57×10^{-3}%	2.09×10^{-3}%	4.20×10^{-3}%	8.43×10^{-3}%	1.69%
Average Network Utilization	2.68×10^{-11}%	5.36×10^{-11}%	8.03×10^{-11}%	1.07×10^{-10}%	2.14×10^{-10}%	4.30×10^{-10}%	8.65×10^{-10}%
Average Storage Utilization over active servers	1.47%	1.53%	1.54%	1.55%	1.57%	1.58%	1.59%
Average Storage Utilization	1.50×10^{-4}%	3.02×10^{-4}%	4.52×10^{-4}%	6.02×10^{-4}%	1.21×10^{-3}%	2.44×10^{-3}%	4.90×10^{-3}%
Average Accelerator Utilization over active servers	36.85%	38.13%	38.49%	38.73%	39.16%	36.63%	31.82%
Average Accelerator Utilization	3.74×10^{-3}%	7.55×10^{-3}%	1.13%	1.50%	3.02%	5.63%	9.73%
Energy consumption (KWh) over number of accepted tasks	14.686	7.344	4.915	3.703	1.862	0.944	0.486

Table 5. Large scale simulation results with the improved SOSM.

	Number of Cells				
	13	**20**	**30**	**50**	**100**
Number of incoming tasks per second	640	1280	2560	5120	10,240
Total Energy Consumption (MWh)	13,429.17	21,006.24	32,271.12	54,679.52	109,369.86
Total Number of submitted Tasks	27,650,718	55,319,241	111,095,039	221,534,173	443,504,461
Total Number of accepted Tasks	27,650,718	55,319,241	111,045,068	217,671,730	435,682,990
Total Number of rejected Tasks	0	0	49,971	3,862,443	7,821,471
Average Processor Utilization over active servers	45.21%	45.33%	45.51%	47.13%	47.15%
Average Processor Utilization	13.84%	18.01%	24.09%	27.03%	27.05%
Average Memory Utilization over active servers	5.52%	5.53%	5.55%	5.88%	5.89%

Table 5. *Cont.*

	Number of Cells				
	13	**20**	**30**	**50**	**100**
Average Memory Utilization	1.69%	2.20%	2.94%	3.38%	3.38%
Average Network Utilization	8.65×10^{-10}%	1.13×10^{-9}%	1.51×10^{-9}%	1.62×10^{-9}%	1.62×10^{-9}%
Average Storage Utilization over active servers	1.59%	1.60%	1.60%	1.67%	1.67%
Average Storage Utilization	4.90×10^{-3}%	6.39×10^{-3}%	8.56×10^{-3}%	9.66×10^{-3}%	9.67×10^{-3}%
Average Accelerator Utilization over active servers	31.82%	30.71%	29.84%	31.80%	31.83%
Average Accelerator Utilization	9.73%	12.19%	15.78%	18.23%	18.25%

Table 6. The percentage of accepted tasks for SOSM and Improved SOSM clouds while increasing the number of cells.

	Number of Cells				
Resource Allocation	**13**	**20**	**30**	**50**	**100**
SOSM	100.000%	97.922%	94.655%	92.563%	92.604%
Improved SOSM	100.000%	100.000%	99.955%	98.257%	98.236%

Table 7. The energy consumption (KWh) over the number of accepted tasks for SOSM and Improved SOSM clouds while increasing the number of cells.

	Number of Cells				
Resource Allocation	**13**	**20**	**30**	**50**	**100**
SOSM	0.486	0.387	0.307	0.266	0.266
Improved SOSM	0.486	0.380	0.291	0.251	0.251

The numerical results indicate that the improved SOSM implementation further boosted the performance of SOSM. The percentage of accepted tasks as the number of cells and incoming tasks increases, is being maintained close to 100% for the improved SOSM, while for SOSM, tasks start to be rejected even from 20 cells. Since the new Suitability Index specifies the performance per unit of power consumption, it is expected that energy efficiency will be improved. Indeed, the total energy consumption over the total number of accepted tasks (Table 7) shows an improvement on the average required KWh of a task, as the number of cells and incoming tasks increases. By comparing the ratio of total spent KWh over the number of accepted tasks, improved SOSM delivers a better energy footprint per task than SOSM (0.251 versus 0.266 in the case of 100 cells).

Additionally, from the metrics extracted, it can be observed that, with the use of improved SOSM, utilization of the hardware resources (i.e., CPU processors, memory, network and storage) has been increased, while the use of accelerators has been reduced. Despite using fewer accelerators, this Suitability Index demonstrated a more efficient solution for the cloud hardware resource allocation problem.

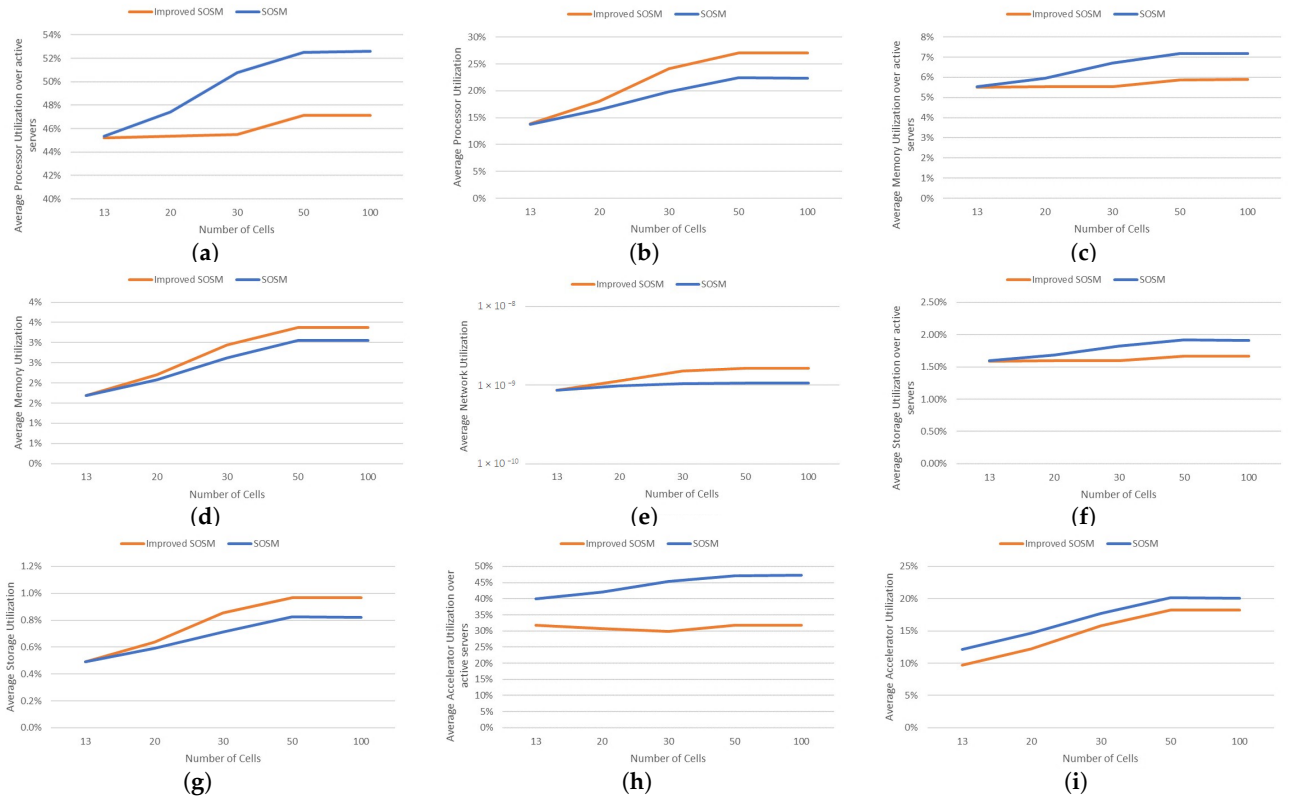

Figure 3. Scalability of the Improved SOSM over SOSM by increasing the number of cells and the number of incoming tasks: (**a**) Average Processor Utilization over active servers, (**b**) Average Processor Utilization, (**c**) Average Memory Utilization over active servers, (**d**) Average Memory Utilization, (**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.

4.3. Synthetic Applications

Supplementary results are given for estimating the behavior of the traditional centralized and Self-Organization and Self-Management mechanisms, using synthetic inputs. Due to the limited CloudLightning use cases [13] and the low parallel performance of the Oil and Gas use case on the accelerators [37], the synthetic set of inputs tries to define a broader set of HPC applications that could be potentially executed on a cloud.

The requirements of the synthetic application tasks that were chosen to be used as inputs in the supplementary experiments, were chosen arbitrarily to implement various types of applications (i.e., CPU intensive applications, network intensive applications, etc.). The selected applications' characteristics are presented in Table 8. It should be noted that $\rho \in [0, 1]$ denotes the percentage of the application that is parallelized on an accelerator.

The distribution of the supplementary synthetic input tasks, as well as the distribution of requirements for each task, were chosen to be uniform. The distribution of the tasks, for an input size of 100,000 tasks, is depicted in Figure 4a. The distribution of Millions of Instructions, vCPUs, Memory, storage and network with respect to input tasks is given in Figure 4b–f, respectively.

The simulation results for the traditional, SOSM and improved SOSM clouds, using the synthetic data presented, are given in Tables 9–11, respectively. The corresponding percentages of accepted tasks for the three allocation mechanisms are given in Table 12, while in Table 13 the energy consumption over the number of accepted tasks is presented. The corresponding diagrams, showing the three allocation mechanisms, are depicted in Figure 5. The total energy consumption for the three approaches are given in Figure 6, while in Figure 7 the total number of accepted and rejected tasks for the whole simulation time are depicted in Figure 7.

Table 8. Requirements for the synthetic application tasks.

Application	Implementation	Min–Max MIPS	Min–Max VMs	vCPUs per VM	Util. vCPU	Min–Max Mem per VM (GB)	Util. Mem	Min–Max Storage per VM (GB)	Min–Max Network per VM (GBps)	Acc. per VM	ρ
1	**CPU**	1,386,228,336.48–5,544,913,345.92	1–16	4–8	1	4–8	1	20–40	0.0025–0.005	0	0
	CPU–GPU	1,386,228,336.48–5,544,913,345.92	1–16	4–8	0.5	4–8	1	20–40	0.0025–0.005	1	0.7
	CPU–MIC	1,386,228,336.48–5,544,913,345.92	1–16	4–8	0.5	4–8	1	20–40	0.0025–0.005	1	0.7
2	**CPU**	462,076,112.16–2,772,456,672.96	1–8	8–16	1	6–10	1	10–20	0.0005–0.001	0	0
	CPU–FPGA	462,076,112.16–2,772,456,672.96	1–8	8–16	0.6	6–10	1	10–20	0.0005–0.001	1	0.8
	CPU–GPU	462,076,112.16–2,772,456,672.96	1–8	8–16	0.6	6–10	1	10–20	0.0005–0.001	1	0.8
3	**CPU**	693,114,168.24–4,158,685,009.44	1–4	4–8	1	4–8	1	4–8	0.0025–0.005	0	0
	CPU–FPGA	693,114,168.24–4,158,685,009.44	1–4	4–8	0.7	4–8	1	4–8	0.0025–0.005	1	0.65
	CPU–MIC	693,114,168.24–4,158,685,009.44	1–4	4–8	0.7	4–8	1	4–8	0.0025–0.005	1	0.65
	CPU -GPU	693,114,168.24–4,158,685,009.44	1–4	4–8	0.7	4–8	1	4–8	0.0025–0.005	1	0.65
4	**CPU**	2,237,788.20–392,963,271.76	1–8	12–12	1	0.5–8	0.47	10–10	4×10^{-12}	0	0
	CPU–GPU	2,237,788.20–392,963,271.76	1–8	12–12	0.51	0.5–8	0.47	10–10	4×10^{-12}	1	0.8
5	**CPU**	14,878,487.26–119,027,898.08	1–8	1–8	1	8–8	0.81	10–10	0	0	0
	CPU–FPGA	14,878,487.26–119,027,898.08	1–8	1–8	1	8–8	0.81	10–10	0	1	0.85
6	**CPU**	2,202,965.68–17,623,725.44	1–8	2–8	1	0.5–1	0.04	10–10	0	0	0
	CPU–MIC	2,202,965.68–17,623,725.44	1–8	2–8	0.046	0.5–1	0.04	10–10	0	1	0.55
7	**CPU**	3,125,137.36–25,001,098.88	1–8	2–8	1	0.5–1	0.4652	10–10	0	0	0
	CPU–MIC	3,125,137.36–25,001,098.88	1–8	2–8	0.056	0.5–1	0.4652	10–10	0	1	0.6
	CPU–FPGA	3,125,137.36–25,001,098.88	1–8	2–8	0.056	0.5–1	0.4652	10–10	0	1	0.6

Table 8. *Cont.*

Application	Implementation	Min–Max MIPS	Min–Max VMs	vCPUs per VM	Util. vCPU	Min–Max Mem per VM (GB)	Util. Mem	Min–Max Storage per VM (GB)	Min–Max Network per VM (GBps)	Acc. per VM	ρ
8	**CPU**	18,545,897.12–148,367,176.96	1–8	2–8	1	0.8–1.5	0.5858	10–10	0	0	0
	CPU–GPU	18,545,897.12–148,367,176.96	1–8	2–8	1	0.8–1.5	0.5858	10–10	0	1	0.8
	CPU–FPGA	18,545,897.12–148,367,176.96	1–8	2–8	1	0.8–1.5	0.5858	10–10	0	1	0.8
	CPU–MIC	18,545,897.12–148,367,176.96	1–8	2–8	1	0.8–1.5	0.5858	10–10	0	1	0.8
9	**CPU**	17,091,794.24–370,917,942.4	8–10	4–8	1	8–10	0.4	10–10	0.0008–0.003	0	0
	CPU–MIC	17,091,794.24–370,917,942.4	8–10	4–8	1	8–10	0.4	10–10	0.0008–0.003	1	0.7
10	**CPU**	693,114,168.24–4,990,422,011.32	8–12	2–6	1	4–8	1	10–20	0.0005–0.001	0	0
	CPU–GPU	693,114,168.24–4,990,422,011.32	8–12	2–6	0.6	4–8	1	10–20	0.0005–0.001	1	0.7
	CPU–FPGA	693,114,168.24–4,990,422,011.32	8–12	2–6	0.6	4–8	1	10–20	0.0005–0.001	1	0.7

It is noted that the simulation parameters, except for the application configurations, were the same as in Section 4.2.

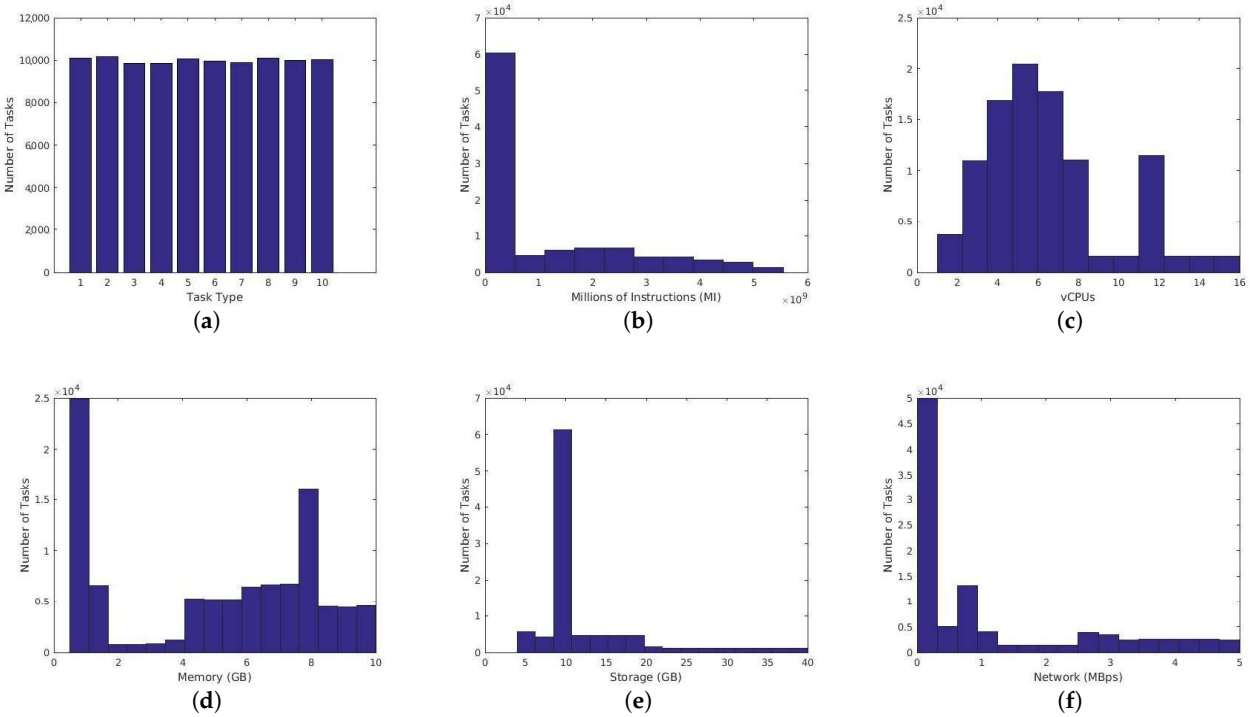

Figure 4. Distribution of task characteristics for the synthetic applications: (**a**) Distribution of tasks with respect to their application type, (**b**) Distribution of the number of MIs required by tasks, (**c**) Distribution of the number of vCPUs required by tasks, (**d**) Distribution of the required Memory with respect to tasks, (**e**) Distribution of the required Storage with respect to tasks, (**f**) Distribution of the required Network with respect to tasks.

Table 9. Large scale simulation results with the traditional cloud on synthetic inputs.

	Number of Cells				
	13	**20**	**30**	**50**	**100**
Number of incoming tasks per second	640	1280	2560	5120	10,240
Total Energy Consumption (MWh)	13,618.62	21,036.18	31,722.35	53,096.19	106,198.41
Total Number of submitted Tasks	27,657,065	55,285,417	110,788,761	222,018,105	441,713,374
Total Number of accepted Tasks	15,819,349	30,673,819	59,717,240	116,985,184	232,979,319
Total Number of rejected Tasks	11,837,716	24,611,598	51,071,521	105,032,921	208,734,055
Average Processor Utilization over active servers	62.92%	63.83%	65.48%	65.43%	65.32%
Average Processor Utilization	14.42%	15.71%	17.57%	18.78%	18.73%
Average Memory Utilization over active servers	11.48%	11.03%	10.56%	10.16%	10.16%
Average Memory Utilization	2.63%	2.72%	2.84%	2.92%	2.92%
Average Network Utilization	97.76%	97.78%	97.71%	97.68%	97.69%
Average Storage Utilization over active servers	5.96%	5.69%	5.41%	5.18%	5.18%
Average Storage Utilization	1.39%	1.42%	1.47%	1.51%	1.51%
Average Accelerator Utilization over active servers	11.76%	11.08%	10%	9.30%	9.33%
Average Accelerator Utilization	2.69%	2.74%	2.70%	2.69%	2.70%

Table 10. Large scale simulation results with the SOSM cloud on synthetic inputs.

	Number of Cells				
	13	**20**	**30**	**50**	**100**
Number of incoming tasks per second	640	1280	2560	5120	10,240
Total Energy Consumption (MWh)	16,721.63	25,748.30	39,017.29	64,757.92	129,530.38
Total Number of submitted Tasks	27,592,651	55,308,306	110,558,115	221,602,170	443,542,128
Total Number of accepted Tasks	20,351,196	38,014,662	71,577,179	137,561,406	275,404,788
Total Number of rejected Tasks	7,241,455	17,293,644	38,980,936	84,040,764	168,137,340
Average Processor Utilization over active servers	49.70%	48.53%	45.42%	44.29%	44.22%
Average Processor Utilization	8.25%	8.75%	9.53%	10.16%	10.20%
Average Memory Utilization over active servers	10.37%	9.94%	8.88%	8.35%	8.33%
Average Memory Utilization	1.72%	1.79%	1.86%	1.91%	1.92%
Average Network Utilization	93.73%	94.05%	93.62%	93.16%	93.11%
Average Storage Utilization over active servers	6.56%	6.17%	5.50%	5.17%	5.15%
Average Storage Utilization	1.10%	1.13%	1.17%	1.20%	1.20%
Average Accelerator Utilization over active servers	51.32%	49.26%	43.35%	40.37%	40.26%
Average Accelerator Utilization	8.53%	8.89%	9.10%	9.26%	9.28%

Table 11. Large scale simulation results with the Improved SOSM cloud on synthetic inputs.

	Number of Cells				
	13	**20**	**30**	**50**	**100**
Number of incoming tasks per second	640	1280	2560	5120	10,240
Total Energy Consumption (MWh)	14,712.09	22,792.32	34,427.73	57,691.08	115,366.41
Total Number of submitted Tasks	27,662,402	55,364,650	110,774,898	220,868,089	440,603,473
Total Number of accepted Tasks	18,950,384	35,895,402	67,665,969	130,788,600	261,030,054
Total Number of rejected Tasks	8,712,018	19,469,248	43,108,929	90,079,489	179,573,419
Average Processor Utilization over active servers	47.51%	46.90%	47.03%	47.22%	47.25%
Average Processor Utilization	7.97%	8.24%	8.59%	8.86%	8.85%
Average Memory Utilization over active servers	10.98%	10.65%	10.30%	10.06%	10.07%
Average Memory Utilization	1.84%	1.87%	1.88%	1.89%	1.89%
Average Network Utilization	95.65%	95.12%	94.55%	94.21%	94.21%
Average Storage Utilization over active servers	6.65%	6.38%	6.16%	6.02%	6.03%
Average Storage Utilization	1.13%	1.13%	1.14%	1.14%	1.14%
Average Accelerator Utilization over active servers	54.31%	53.05%	52.14%	51.59%	51.64%
Average Accelerator Utilization	9.12%	9.33%	9.52%	9.68%	9.68%

Table 12. The percentage of accepted tasks for traditional, SOSM and Improved SOSM clouds for the synthetic inputs.

	Number of Cells				
Resource Allocation	**13**	**20**	**30**	**50**	**100**
Traditional	57.1982%	55.4827%	53.9019%	52.6917%	52.7445%
SOSM	73.7559%	68.7323%	64.7417%	62.0758%	62.0921%
Improved SOSM	68.5059%	64.8345%	61.0842%	59.2157%	59.2438%

Table 13. The energy consumption (KWh) over the number of accepted tasks for SOSM and Improved SOSM clouds while increasing the number of cells for the synthetic inputs.

	Number of Cells				
Resource Allocation	**13**	**20**	**30**	**50**	**100**
Traditional	0.861	0.686	0.531	0.454	0.456
SOSM	0.822	0.677	0.545	0.471	0.470
Improved SOSM	0.776	0.635	0.509	0.441	0.442

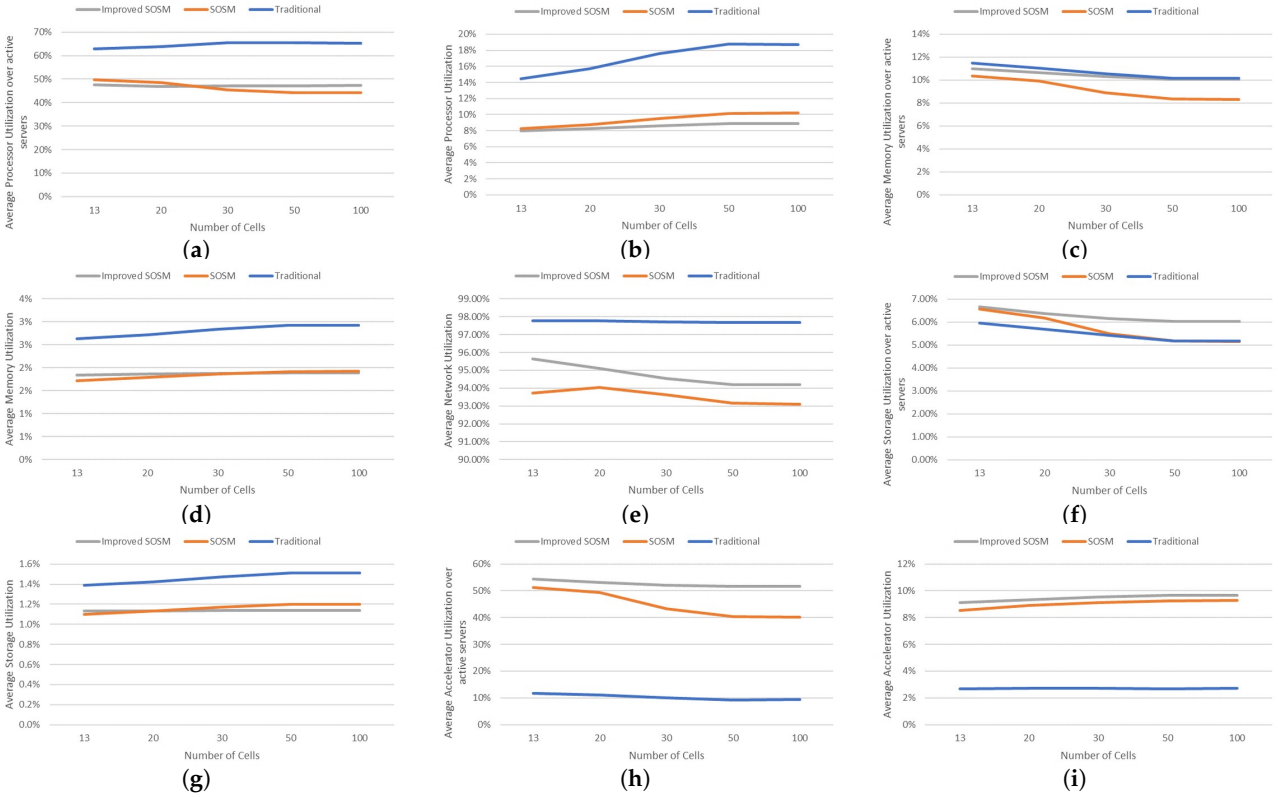

Figure 5. Scalability of the three allocation mechanisms by increasing the number of cells and number of incoming tasks, for the synthetic inputs: (**a**) Average Processor Utilization over active servers, (**b**) Average Processor Utilization, (**c**) Average Memory Utilization over active servers, (**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.

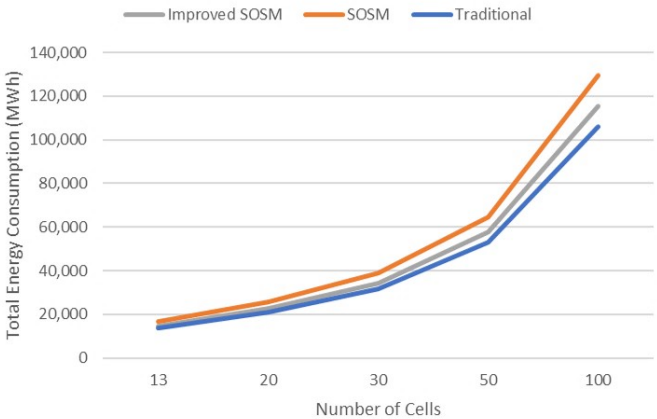

Figure 6. Total energy consumption in MWh for traditional centralized and SOSM clouds, for the synthetic inputs.

The results confirm the improvement of the SOSM variants over the traditional resource allocation system in all sizes of the cloud. This can be derived by the fact that both SOSM clouds managed to serve more tasks for the different sizes of clouds and the number of incoming tasks per second. The improved SOSM variant rejected slightly more tasks than the SOSM cloud, but managed to keep the consumed energy at lower levels. Thus, the ratio of total energy consumption over the number of accepted tasks of improved SOSM (Table 13) was 0.442 (versus 0.470 of SOSM), validating the fact that the proposed Assessment Function and Suitability Index can achieve an improved management of energy consumption required for each task to be executed. Finally, it can be observed that this was achieved due to the higher utilization of the energy efficient accelerators. The utilization of the rest of the hardware resources remained at the same level.

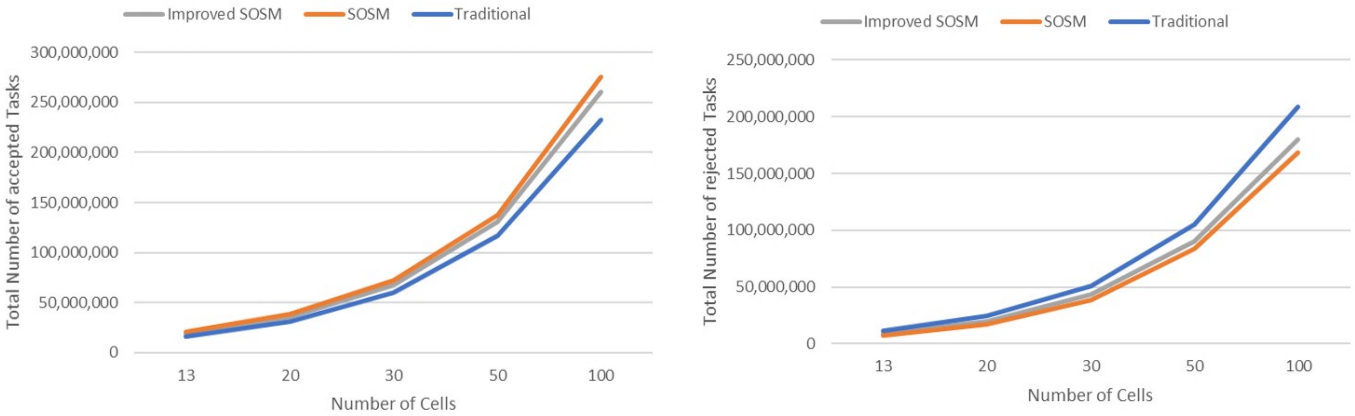

Figure 7. The number of accepted and rejected tasks for traditional centralized and SOSM clouds, for the synthetic inputs: (**a**) Total Number of accepted Tasks, (**b**) Total Number of rejected Tasks.

5. Discussion and Conclusions

The recently proposed Self-Organization and Self-Management resource allocation scheme for heterogeneous HPC clouds was recently proposed and evaluated at a large scale through simulation. This scheme was proven to be more efficient in terms of computational efficiency, energy consumption, service delivery and scalability, compared to the traditional centralized cloud orchestration. The numerical results indicated that, as the number of simulated cloud nodes and incoming tasks increase, the improvement is further extended.

In this work, a new Assessment Function and Suitability Index are being proposed for further optimizing the behavior of the SOSM system. The proposed schemes target improved energy consumption as a global goal of the cloud infrastructure, by reflecting the capability of the hardware resources to perform computations with the minimum energy consumption. This is expected to further utilize resources with a higher performance per unit of consumed power, thus achieving an efficient decentralized resource allocation scheme with a lower energy footprint.

The evaluation of the improved SOSM was performed through simulation, and more specifically with the use of the efficient CloudLightning discrete-time simulator, which is capable of scaling up to millions of cloud nodes. The simulated applications arose from traces of three real world HPC applications, while extended benchmarking was performed by generating synthetic HPC applications. The numerical results indicated that the Improved SOSM cloud performs equivalently to the SOSM cloud in terms of computational efficiency, service delivery and scalability, while outperforming on the required average energy consumption used by the served tasks. Both SOSM variants outperform the traditional centralised recourse allocation scheme in all cases and metrics utilized. The use of synthetic application inputs confirmed the tendency of the SOSM clouds.

Future work will include the adaptation of distributed machine learning techniques for enhancing the framework with predictive capabilities that will guide the incoming tasks more efficiently onto the hardware resources. Additionally, new assessment functions and the Suitability Index will be proposed for improving the utilization of the resources.

Author Contributions: Conceptualization, C.K.F.-P. and K.M.G.; methodology, all authors; software, C.K.F.-P.; validation, K.M.G.; formal analysis, C.K.F.-P.; investigation, C.K.F.-P.; resources, G.A.G.; data curation, K.M.G.; writing original draft preparation, all authors; writing – review & editing, all authors; visualization, K.M.G. and C.K.F.-P.; supervision, G.A.G. and D.T.; funding acquisition, G.A.G. and D.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by the European Union's Horizon 2020 Research and Innovation Programme through CloudLightning project under Grant Agreement No. 643946.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author, as the number of data entries is large to fit in this section.

Acknowledgments: The authors would like to thank Antonios Makaratzis for collecting simulation execution data. Additionally, 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.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

A2LB	Agent Based Load Balancing Algorithm
CL	CloudLightning
CPU	Central Processing Unit
DDPG	Deep Deterministic Policy Gradient
DRL	Deep Reinforcement Learning
E2C	Edge to Cloud Continuum
FPGAs	Field Programmable Gate Arrays
GPUs	Graphics Processing Units
HPC	High Performance Computing
LSTM	Long Short-Term memory
MICs	Many Integrated Cores
MIPS	Million Instructions per Second
MPI	Message Passing Interface
QoS	Quality of Service
RL	Reinforcement Learning
SI	Suitability Index
SLA	Service Level Agreement
SOSM	Self-Organization and Self-Management
WSC	Warehouse Scale Computer

References

1. Bittencourt, L.; Immich, R.; Sakellariou, R.; Fonseca, N.; Madeira, E.; Curado, M.; Villas, L.; DaSilva, L.; Lee, C.; Rana, O. The Internet of Things, Fog and Cloud continuum: Integration and challenges. *Internet Things* **2018**, *3–4*, 134–155. [CrossRef]
2. Feng, C.; Wang, Y.; Chen, Q.; Ding, Y.; Strbac, G.; Kang, C. Smart grid encounters edge computing: Opportunities and applications. *Adv. Appl. Energy* **2021**, *1*, 100006. [CrossRef]
3. Hafeez, T.; Xu, L.; Mcardle, G. Edge Intelligence for Data Handling and Predictive Maintenance in IIOT. *IEEE Access* **2021**, *9*, 49355–49371. [CrossRef]

4. Mahmood, A.; Zhang, W.E.; Sheng, Q.Z. Software-Defined Heterogeneous Vehicular Networking: The Architectural Design and Open Challenges. *Future Internet* **2019**, *11*. [CrossRef]
5. Liu, Y.; Peng, M.; Shou, G.; Chen, Y.; Chen, S. Toward Edge Intelligence: Multiaccess Edge Computing for 5G and Internet of Things. *IEEE Internet Things J.* **2020**, *7*, 6722–6747. [CrossRef]
6. Wu, H.; Zhang, Z.; Guan, C.; Wolter, K.; Xu, M. Collaborate Edge and Cloud Computing with Distributed Deep Learning for Smart City Internet of Things. *IEEE Internet Things J.* **2020**, *7*, 8099–8110. [CrossRef]
7. Wang, J.; Li, D. Adaptive Computing Optimization in Software-Defined Network-Based Industrial Internet of Things with Fog Computing. *Sensors* **2018**, *18*, 2509. [CrossRef] [PubMed]
8. Marinescu, D.C. *Complex Systems and Clouds: A Self-Organization and Self-Management Perspective*; Morgan Kaufmann: Burlington, MA, USA, 2016.
9. Puviani, M.; Frei, R. Self-management for cloud computing. In Proceedings of the 2013 Science and Information Conference, London, UK, 7–9 October 2013; pp. 940–946.
10. Puviani, M.; Cabri, G.; Zambonelli, F. A Taxonomy of Architectural Patterns for Self-Adaptive Systems. In Proceedings of the International C* Conference on Computer Science and Software Engineering, C3S2E '13, Porto, Portugal, 10–12 July 2013; Association for Computing Machinery: New York, NY, USA, 2013; pp. 77–85. [CrossRef]
11. Tomar, R.; Khanna, A.; Bansal, A.; Fore, V. An Architectural View Towards Autonomic Cloud Computing. In *Data Engineering and Intelligent Computing*; Satapathy, S.C., Bhateja, V., Raju, K.S., Janakiramaiah, B., Eds.; Springer: Singapore, 2018; pp. 573–582.
12. Singh, S.; Chana, I. QoS-Aware Autonomic Resource Management in Cloud Computing: A Systematic Review. *ACM Comput. Surv.* **2015**, *48*, 1–46. [CrossRef]
13. Lynn, T.; Xiong, H.; Dong, D.; Momani, B.; Gravvanis, G.; Filelis-Papadopoulos, C.; Elster, A.; Khan, M.M.Z.M.; Tzovaras, D.; Giannoutakis, K.; et al. CLOUDLIGHTNING: A Framework for a Self-organising and Self-managing Heterogeneous Cloud. In Proceedings of the 6th International Conference on Cloud Computing and Services Science, Rome, Italy, 23–25 April 2016; Volume 1, pp. 333–338. [CrossRef]
14. Xiong, H.; Filelis-Papadopoulos, C.; Dong, D.; Castañé, G.G.; Meyer, S.; Morrison, J.P. Energy-Efficient Servers and Cloud. In *Hardware Accelerators in Data Centers*; Springer International Publishing: Cham, Switzerland, 2019; pp. 163–180. [CrossRef]
15. Madni, S.H.; Latiff, M.S.; Coulibaly, Y.; Abdulhamid, S.M. Recent Advancements in Resource Allocation Techniques for Cloud Computing Environment: A Systematic Review. *Clust. Comput.* **2017**, *20*, 2489–2533. [CrossRef]
16. Singh, S.; Chana, I. A survey on resource scheduling in cloud computing: Issues and challenges. *J. Grid Comput.* **2016**, *14*, 217–264. [CrossRef]
17. Yousafzai, A.; Gani, A.; Noor, R.M.; Sookhak, M.; Talebian, H.; Shiraz, M.; Khan, M.K. Cloud resource allocation schemes: Review, taxonomy, and opportunities. *Knowl. Inf. Syst.* **2017**, *50*, 347–381. [CrossRef]
18. Jung, G.; Sim, K.M. Agent-Based Adaptive Resource Allocation on the Cloud Computing Environment. In Proceedings of the 2011 40th International Conference on Parallel Processing Workshops, Taipei City, Taiwan, 13–16 September 2011; pp. 345–351. [CrossRef]
19. Singh, A.; Juneja, D.; Malhotra, M. Autonomous Agent Based Load Balancing Algorithm in Cloud Computing. *Procedia Comput. Sci.* **2015**, *45*, 832–841. [CrossRef]
20. Zhang, J.; Xie, N.; Zhang, X.; Yue, K.; Li, W.; Kumar, D. Machine Learning Based Resource Allocation of Cloud Computing in Auction. *Comput. Mater. Contin.* **2018**, *56*, 123–135. [CrossRef]
21. Thein, T.; Myo, M.M.; Parvin, S.; Gawanmeh, A. Reinforcement learning based methodology for energy-efficient resource allocation in cloud data centers. *J. King Saud Univ. Comput. Inf. Sci.* **2020**, *32*, 1127–1139. [CrossRef]
22. Bitsakos, C.; Konstantinou, I.; Koziris, N. DERP: A Deep Reinforcement Learning Cloud System for Elastic Resource Provisioning. In Proceedings of the 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Nicosia, Cyprus, 10–13 December 2018; pp. 21–29. [CrossRef]
23. Liu, N.; Li, Z.; Xu, J.; Xu, Z.; Lin, S.; Qiu, Q.; Tang, J.; Wang, Y. A Hierarchical Framework of Cloud Resource Allocation and Power Management Using Deep Reinforcement Learning. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 372–382. [CrossRef]
24. Du, B.; Wu, C.; Huang, Z. Learning Resource Allocation and Pricing for Cloud Profit Maximization. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19), Honolulu, HI, USA, 27 January–1 February 2019; 33, pp. 7570–7577. [CrossRef]
25. Miriam, A.J.; Saminathan, R.; Chakaravarthi, S. Non-dominated Sorting Genetic Algorithm (NSGA-III) for effective resource allocation in cloud. *Evol. Intell.* **2021**, *14*, 759–765. [CrossRef]
26. Chandran, R.; Kumar, S.R.; Gayathri, N. Genetic algorithm-based tabu search for optimal energy-aware allocation of data center resources. *Soft Comput.* **2020**, *24*, 16705–16718. [CrossRef]
27. Abrol, P.; Guupta, S.; Singh, S. Nature-inspired metaheuristics in cloud: A review. In *ICT Systems and Sustainability*; Springer: Singapore, 2020; pp. 13–34.
28. Xiong, H.; Filelis-Papadopoulos, C.; Castañe, G.G.; Dong, D.; Morrison, J.P. Self-Organising, Self-Managing Frameworks and Strategies. In *Heterogeneity, High Performance Computing, Self-Organization and the Cloud*; Palgrave Macmillan: Cham, Switzerland, 2018; pp. 63–88.

29. Barroso, L.A.; Clidaras, J.; Holzle, U. The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second edition. *Synth. Lect. Comput. Archit.* **2013**, *8*, 1–154. [CrossRef]
30. Filelis-Papadopoulos, C.K.; Giannoutakis, K.M.; Gravvanis, G.A.; Tzovaras, D. Large-scale simulation of a self-organizing self-management cloud computing framework. *J. Supercomput.* **2018**, *74*, 530–550. [CrossRef]
31. Filelis-Papadopoulos, C.; Xiong, H.; Spătaru, A.; Castañé, G.G.; Dong, D.; Gravvanis, G.A.; Morrison, J.P. A Generic Framework Supporting Self-Organisation and Self-Management in Hierarchical Systems. In Proceedings of the 2017 16th International Symposium on Parallel and Distributed Computing (ISPDC), Innsbruck, Austria, 3–6 July 2017; pp. 149–156. [CrossRef]
32. Giannoutakis, K.M.; Filelis-Papadopoulos, C.K.; Gravvanis, G.A.; Tzovaras, D. Evaluation of self-organizing and self-managing heterogeneous high performance computing clouds through discrete-time simulation. *Concurr. Comput. Pract. Exp.* **2021**, *33*, e6326. [CrossRef]
33. Senyo, P.K.; Addae, E.; Boateng, R. Cloud computing research: A review of research themes, frameworks, methods and future research directions. *Int. J. Inf. Manag.* **2018**, *38*, 128–139. [CrossRef]
34. Lynn, T.; Gourinovitch, A.; Byrne, J.; Byrne, P.J.; Svorobej, S.; Giannoutakis, K.; Kenny, D.; Morrison, J. A Preliminary Systematic Review of Computer Science Literature on Cloud Computing Research using Open Source Simulation Platforms. In Proceedings of the 7th International Conference on Cloud Computing and Services Science—CLOSER, Porto, Portugal, 24–26 April 2017; pp. 565–573. [CrossRef]
35. Giannoutakis, K.M.; Makaratzis, A.T.; Tzovaras, D.; Filelis-Papadopoulos, C.K.; Gravvanis, G.A. On the Power Consumption Modeling for the Simulation of Heterogeneous HPC Clouds. In Proceedings of the 1st International Workshop on Next Generation of Cloud Architectures, Belgrade, Serbia, 23–26 April 2017; Association for Computing Machinery: New York, NY, USA, 2017; [CrossRef]
36. Makaratzis, A.T.; Khan, M.M.; Giannoutakis, K.M.; Elster, A.C.; Tzovaras, D. GPU Power Modeling of HPC Applications for the Simulation of Heterogeneous Clouds. In *Parallel Processing and Applied Mathematics*; Wyrzykowski, R., Dongarra, J., Deelman, E., Karczewski, K., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 91–101.
37. Khan, M.; Becker, T.; Kuppuudaiyar, P.; Elster, A.C. Container-Based Virtualization for Heterogeneous HPC Clouds: Insights from the EU H2020 CloudLightning Project. In Proceedings of the 2018 IEEE International Conference on Cloud Engineering (IC2E), Orlando, FL, USA, 17–20 April 2018; pp. 392–397. [CrossRef]
38. Dell PowerEdge C4130 Data Sheet. Available online: https://i.dell.com/sites/doccontent/shared-content/data-sheets/en/Documents/Dell-PowerEdge-C4130-Spec-Sheet.pdf (accessed on 3 November 2021).
39. Nvidia Tesla P100 Data Sheet. Available online: https://images.nvidia.com/content/tesla/pdf/nvidia-tesla-p100-PCIe-datasheet.pdf (accessed on 3 November 2021).
40. Intel Xeon Phi 5110P Data Sheet. Available online: https://www.intel.com/content/dam/www/public/us/en/documents/datasheets/xeon-phi-coprocessor-datasheet.pdf (accessed on 3 November 2021).
41. MPC-X Data Sheet. Available online: https://www.maxeler.com/products/mpc-xseries/ (accessed on 3 November 2021).