

Article

Improving the Eco-Efficiency of High Performance Computing Clusters Using EECluster

Alberto Cocaña-Fernández *, Luciano Sánchez † and José Ranilla †

Departamento de Informática, Universidad de Oviedo, 33204 Gijón, Spain; luciano@uniovi.es (L.S.); ranilla@uniovi.es (J.R.)

* Correspondence: cocanaalberto@gmail.com; Tel.: +34-985-182-121 or +34-985-182-130; Fax: +34-985-181-986

† These authors contributed equally to this work.

Academic Editor: Enrico Pontelli

Received: 31 December 2015; Accepted: 7 March 2016; Published: 14 March 2016

Abstract: As data and supercomputing centres increase their performance to improve service quality and target more ambitious challenges every day, their carbon footprint also continues to grow, and has already reached the magnitude of the aviation industry. Also, high power consumptions are building up to a remarkable bottleneck for the expansion of these infrastructures in economic terms due to the unavailability of sufficient energy sources. A substantial part of the problem is caused by current energy consumptions of High Performance Computing (HPC) clusters. To alleviate this situation, we present in this work EECluster, a tool that integrates with multiple open-source Resource Management Systems to significantly reduce the carbon footprint of clusters by improving their energy efficiency. EECluster implements a dynamic power management mechanism based on Computational Intelligence techniques by learning a set of rules through multi-criteria evolutionary algorithms. This approach enables cluster operators to find the optimal balance between a reduction in the cluster energy consumptions, service quality, and number of reconfigurations. Experimental studies using both synthetic and actual workloads from a real world cluster support the adoption of this tool to reduce the carbon footprint of HPC clusters.

Keywords: energy-efficient cluster computing; multi-criteria decision making; evolutionary algorithms

1. Introduction

Data and supercomputing centres are an essential element in modern society, as the vast majority of IT services are supported by them, profiting from the consolidation and centralization of high performance processors and networks. Targeting both academic and industrial communities, they provide the key infrastructure for web and application servers, e-commerce platforms, corporate databases, network storage, data mining, or the high performance computing resources required to address fundamental problems in science and engineering, to name a few examples.

The versatility of these computing facilities, coupled with the ever-increasing demand for IT services and their substantial power consumption, makes data and supercomputing centres one of the fastest-growing users of electricity in developed countries [1]. According to [1,2], electricity consumptions in the U.S. alone escalated from 61 billion kilowatt-hours (kWh) in 2006 to 91 billion kWh in 2013, and is projected to increase to 140 billion kWh in 2020. However, it should be noted that these large energy demands not only produce a significant economical impact for IT services providers [3,4], but also a carbon footprint equivalent to the aviation industry [5], which is expected to reach 340 million metric tons of CO₂ by 2020 worldwide [6].

Because of this, there is an unyielding need to improve the energy efficiency of data and supercomputing centres to reduce their environmental impact, operation costs and to improve the reliability of their components.

Abundant research has been conducted over the last years on the improvement of cluster computing efficiency, following multiple approaches that could be taxonomically classified in two categories: static and dynamic power management [7]. Static approaches focus on the development of low power CPUs seeking maximum efficiency, such as the IBM PowerPC A2 processors [8,9], as well as using GPUs or Intel Xeon Phi coprocessors as the main computing resources, given that this type of hardware is designed for an optimal FLOPS/watt relation instead of just raw performance. Dynamic techniques focus on the reconfiguration of the compute resources to best suit current workloads, saving energy when the cluster is underused. Among these techniques is the Dynamic Voltage and Frequency Scaling (DVFS) [10–17], which adjusts CPU frequency and voltage to match current demand, energy-efficient job schedulers that implement algorithms capable of reducing intercommunication-related power consumptions [18,19], thermal-aware methods which take into account the cooling efficiency of each area of the cluster [20,21], or software frameworks to assist in the development of energy-efficient applications [22–26]. Lastly, the adaptive resource cluster technique consists of the automatic reconfiguration of the cluster resources to fit the workload at every moment by switching on or off its compute nodes, thus saving energy whenever these are idle. This technique has been applied to Load-Balancing clusters in [27–32] and in VMware vSphere [33] and Citrix XenServer [34] hypervisors. Recently, various software tools implementing this technique in HPC clusters have also been developed [35–37].

However, previous adaptive resource solutions for High Performance Computing (HPC) clusters have limited practical applications for two fundamental reasons. Firstly, as shown in [38,39], closed sets of expert-defined rules are not optimal to every scenario, leading to a lack of flexibility when it comes to complying with the preferences and tolerances of real-world cluster administrators in terms of impact in service quality and node reliability. Secondly, these solutions require its expert system to be tuned by hand, what is a complex task and is likely to conduce to incorrectly-configured systems that can cause substantial interferences with the cluster operation, such as node thrashing or reduction of its productivity, as demonstrated in [40].

Because of this, the tool EECluster is presented that overcomes these limitations. EECluster can improve the energy efficiency of HPC clusters by dynamically adapting their resources to the changing workloads. This is done using a Hybrid Genetic Fuzzy System as the decision-making mechanism and is tuned by means of multi-objective evolutionary algorithms in a machine learning approach to achieve good compliance with the administrator preferences.

The remainder of the paper is as follows. Section 2 explains the concept of eco-efficiency and details the modelling assumptions for the carbon footprint of an HPC. Section 3 explains the architecture of the EECluster tool. Section 4 explains the decision making-mechanism. Section 5 explains the learning algorithm used. Section 6 shows multiple experimental results in both synthetic and actual scenarios. Section 7 concludes the paper and discusses the future work.

2. Eco-Efficiency

The concept of eco-efficiency brings together economic and environmental factors for a more efficient use of resources and lower emissions [41]. Eco-efficiency is represented by the quotient between the service value and its environmental influence. In the particular case of HPCs, the service value is related to the Quality of Service (QoS), and the environmental influence affects both energy consumption and greenhouse gas emissions.

As mentioned in the introduction, the dependence between the energy consumption and the Quality of Service has been studied, and different strategies were proposed to improve their balance [35–40]. In this work, these studies are updated by including other sources of carbon dioxide emissions that are originated in the life cycle of a compute node. These additional sources are of

secondary importance but nonetheless represent a significant part of the emissions. According to [42], manufacturing a computer requires more than 1700 kWh of primary energy, and more than 380 kg of CO₂ are emitted in the process, accounting for a significant fraction of the greenhouse emissions during the whole life of the equipment. It must be noted that a standard factor of 220 kg CO₂/MWh was assumed for manufacturing-related consumptions, corresponding to an energy mix with a significant proportion of wind power. For operation-related consumptions, the 370 kg CO₂/MWh emission factor reported by the Ministry of Agriculture, Food and Environment from the Government of Spain (“Ministerio de Agricultura, Alimentación y Medio Ambiente”) [43] was used. This factor must be altered accordingly for clusters operating under different energy mixes.

As a consequence of this, in this paper it is proposed that three different aspects are taken into account in the model of the emissions of an HPC:

1. Dependence between QoS and primary consumption of energy. The primary savings are about 370 g of CO₂ for each kWh of electrical energy that is saved during the operation of the HPC.
2. Dependence between the QoS and the lifespan of the equipment. According to our own experience, the average life of a compute node in an HPC cluster is between 4 and 5 years. The number of failures of a given node during its whole lifetime is typically two or three. A rough estimation of the average number of failures of a single node during a year is 0.5 failures/year (thus 0.5 failures/year * 5 years = 2.5 failures). Both the life extent and the quantity of failures depend on the number of power-on and power-off cycles. Heavily loaded nodes might suffer from 0.75 failures/year and a shorter lifespan of 3 years. Assuming that the most common failures are power supplies, motherboards, and disk drives, the typical cost of a reparation can be estimated in 5% of the acquisition cost, *i.e.*, about 20 kg of CO₂ are saved for each failure that is prevented. Each additional year of use of a compute node saves more than 80 kg of CO₂ (approx. 22% of the total manufacturing emissions if the life is between 4 and 5 years, as mentioned). This includes the primary energy used for manufacturing a new node and the recycling costs of the discarded equipment.
3. Dependence between the QoS and the lifespan of the support equipment. An additional 1% was added for each saved kWh (2.2 g CO₂) and 1 g CO₂ for each saved power cycle. In the first case, this models the improved failure rate of support equipment such as cooling fans and air conditioning. The second case models different failures in the datacenter that may be caused by current surges when a large number of compute nodes are powered on or off at the same time.

The emissions model described in this section will be applied in the experiments of Section 6 to estimate global energy savings and carbon footprint reductions as a result of adopting of the proposed system.

3. Architecture

Computing clusters are a type of computer system consisting of multiple computers interconnected that, together, work as a single computing resource [44]. High Performance Computing (HPC) clusters are a particular type of cluster whose main purpose is to address complex and computationally-demanding problems, such as new material, semiconductors, or drugs design, cardiovascular engineering, new combustion systems, cancer detection and therapies, CO₂ sequestration, *etc.* [45].

HPC clusters typically combine a master node and several compute nodes. The master node is the only one accessible by the users and is tasked with the cluster management using various software components, including the Resource Management System (RMS) and monitoring tools (such as Ganglia, Nagios, Zabbix), among others. The RMS is a software layer which abstracts users from the cluster underlying hardware by providing a mechanism where they can submit resource requests to run any supplied software program (hereafter denoted as *jobs*). It is worth noting that cluster resources are represented logically by a number of slots which, depending on the RMS configuration, can depict

form a single CPU core to a whole compute node. The RMS working cycle consists of (1) gathering job submissions in an internal queue; (2) running a job scheduling algorithm to find the best possible matching between the resources available in the compute nodes and the slots requested by each job and (3) assigning slots and dispatching the job to the compute nodes (see Figure 1). Data and results are passed between the master and the compute nodes through a shared network storage space by means of a network file system or a storage area network.

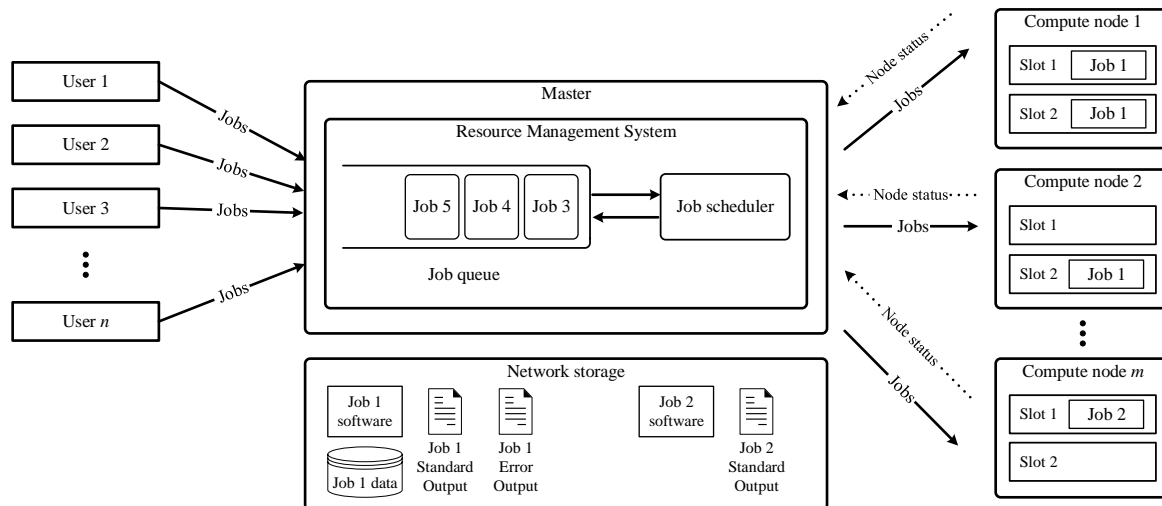


Figure 1. Resource Management System (RMS) components.

The EECluster tool is a solution which can reduce the carbon footprint of ordinary HPC clusters running open-source RMS, such as OGE (Oracle Grid Engine, Open Grid Engine/SGE (Sun Grid Engine, Son of Grid Engine) and PBS (Portable Batch System)/TORQUE (Terascale Open-source Resource and QUEUE Manager) by implementing an intelligent mechanism to adapt the cluster resources to the current workload, saving energy when the cluster is underused. Specifically, the prototype of EECluster features only two out-of-the-box connectors for OGE/SGE and PBS/TORQUE, as these are two of the most used RMS worldwide in HPC infrastructures. As mentioned in reference [46], the OGE/SGE family (including its multiple branches of products and projects, such as Sun Grid Engine, Oracle Grid Engine, Open Grid Engine, and Son of Grid Engine) is a suitable choice for small and medium sites because it is easy to deploy and operate, what has led to a very substantial expansion over the last decade in HPC centres. On the other hand, TORQUE (Terascale Open-source Resource and QUEUE Manager) is the open-source RMS based on the original PBS project (Portable Batch System), and it is arguably the most widely-used batch system nowadays in HPC grid infrastructures and also in small and medium site [46]. It is noteworthy that EECluster can be potentially integrated with any RMS as long as it provides a suitable interface in the form of either a series of command-line utilities or an API (Application Programming Interface) that allows EECluster to obtain the required information for its operation (detailed below).

EECluster is composed of a service (EEClusterd) and a learning algorithm, coupled with a Database Management System (DBMS) as the persistence system, and a web-based administration dashboard. The EEClusterd service periodically updates an internal set of cluster status records by retrieving information from multiple command-line applications. This information, which is stored in a Database Management System, is used by the EECluster decision-making mechanism to dynamically reconfigure the cluster resources by issuing power-on or shutdown commands to the compute nodes using the Power Management module. The learning algorithm mission is to find a set of optimal configurations for the decision-making mechanism from which the administrator can

choose one according to its preferences in terms of impact in the service quality, energy savings, and node reconfigurations.

A functional prototype of EECluster can be downloaded via web [47,48], where can also be found a brief description of the software, quick start guides, contact address and acknowledgements.

Figure 2 provides a high-level overview of the system components. A detailed description of this architecture is out of the scope of this paper and can be found in reference [49].

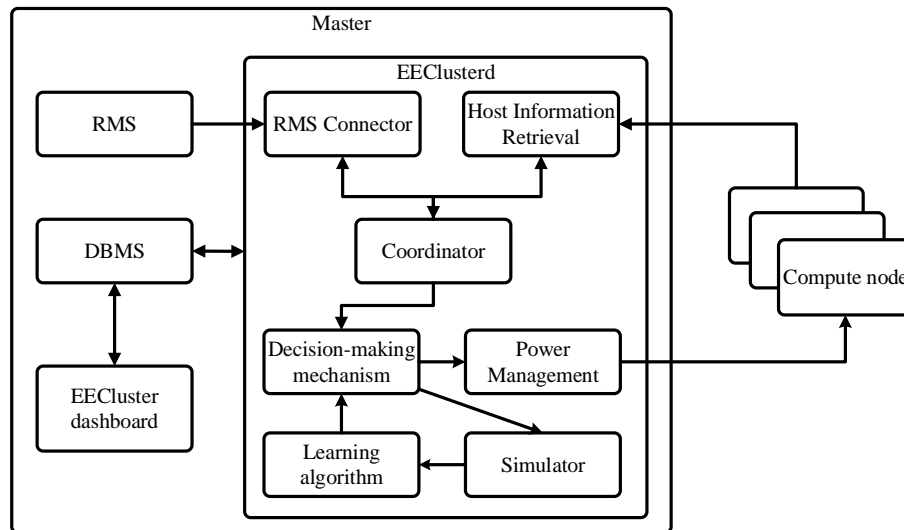


Figure 2. EECluster Tool: System components overview. DBMS: Database Management System.

4. Decision-Making Mechanism

The essential component in an adaptive resource cluster solution is the decision-making mechanism, for it is the one that determines the amount of nodes that will be available at every moment. As mentioned earlier, multiple approaches have been proposed previously based on sets of expert-defined rules, such as “if the node i has been idle for more than a t time threshold, it must be powered off”. Closed expert systems like these have the advantage of coping better with unforeseen workload scenarios over systems learnt automatically in a machine learning approach. This is because machine-learnt systems are more likely to overtrain, especially in scenarios with great changes in the pattern of job arrivals. However, this advantage of simple expert systems comes at the price of low flexibility to adapt to both the sharp changes inherent to ordinary HPC clusters due to the large granularity of its workload (low number of concurrent jobs and multiple resources requested by each job), and the complex set of constraints implicit in the preferences of the cluster administrator. The first limitation leads to worse results than the ones obtainable with a pure machine learning approach. The second limitation leads to a potentially inadmissible solution for the cluster administrator if it does not comply with his or her tolerance of negative impacts on service quality and node reliability.

In order to avoid overtraining and achieve good generalization and flexibility capabilities, EECluster decision-making mechanism is a Hybrid Genetic Fuzzy System (HGFS) composed of both a set of crisp expert-defined rules and a set of fuzzy rules elicited automatically from previous workload records in a machine learning approach. The first set of human-generated expert rules was adapted from reference [36], and can be defined as follows:

- If the current number of resources are insufficient to run every queued job in a sequential manner, then keep powered on at least the highest number of slots requested by any queued job, as long as that amount does not exceed the total number of slots in the cluster.
- If the average waiting time for the queued jobs is higher than a given threshold t_{max} or if the number or queued jobs is higher than a given threshold n_{max} , then power on one slot.

- If the average waiting time for the queued jobs is lower than a given threshold t_{min} or if the number of queued jobs is lower than a given threshold n_{min} , then power off one slot.

The mission of this rule set is to assure that minimum working conditions for the cluster are met, avoiding undesired behaviours in unforeseen scenarios, which may lead to a dramatic impact in the service quality or to a node thrashing effect reducing node reliability and causing early damages in the hardware equipment.

The purpose of the second set of computer-generated rules is the progressive shutdown of idle nodes when the cluster load decreases. Each rule in this set defines the degree of truth of the assertion “the i -node must be powered off”. This degree of truth is computed using a zero-order Tagaki–Sugeno–Kang fuzzy model [50,51] with N triangular fuzzy subsets on the domain of the nodes idle times and N weights between 0 and 1. For instance, if $N = 3$ then the computer-generated rules would consider three linguistic terms regarding the total amount of time that the i -node has been at idle state, with N_1 being “SHORT”, N_2 being “MEDIUM” and N_3 being “LARGE”. In this case, the degree of truth of the aforementioned assertion would be computed as:

$$\text{off}(\text{node } i) = \widetilde{\text{SHORT}}(\text{idle}_i) \cdot w_1 + \widetilde{\text{MEDIUM}}(\text{idle}_i) \cdot w_2 + \widetilde{\text{LARGE}}(\text{idle}_i) \cdot w_3 \quad (1)$$

idle_i is the amount of time that the i -node has been idle, $\widetilde{\text{SHORT}}$, $\widetilde{\text{MEDIUM}}$, and $\widetilde{\text{LARGE}}$ are fuzzy sets with triangular memberships [50], and w_1, w_2, w_3 are the N weights taking values between 0 and 1.

Once each rule has been computed, results are combined so that the number of nodes to be powered off is the sum of the values off for each node. As can be seen, this second set of fuzzy rules does not require nodes to reach a certain crisp value before they can be selected for shutting down, but rather is applied to the cluster as a whole, as opposed to the rules proposed previously to power off idle nodes, such as in reference [36]. This approach allows the system to respond to smaller changes in the cluster load more frequently, thus progressively adapting its resources to better match workload valleys produced when jobs release their resources upon completion. Further information on the Hybrid Genetic Fuzzy System can be found in references [38,39].

Once a decision is made by the HGFS determining the number of slots that must be powered on/off, then this decision is translated to a set of physical compute nodes that will be chosen for reconfiguration. This is done considering the state of each node (only idle nodes can be shutdown and only powered-off nodes are candidates for power-on) plus two additional values: the node efficiency, measured as $\frac{\text{performance}}{\text{power consumption}}$, and the timestamp of its last failure. The latter is used to determine how likely it is for a given node to fail upon a reconfiguration request, in such a way that if a node was recently issued a power on/off command and it failed to comply with it, then the same problem is expected also to occur in the near future. However, if the node failed a long time ago it is more likely to have been repaired. The target nodes to be reconfigured are first split into two groups depending on whether they failed or not to comply with the last issued command. The first group, consisting of the nodes which worked correctly, are sorted according to their efficiency so that the least efficient ones are chosen to be powered off and, conversely, the most efficient ones are chosen to be powered on. If the nodes in the previous group are not enough to match the slots in the reconfiguration decision, the remaining nodes are chosen from the second group, which are sorted according to the timestamps of their failures, choosing first the nodes with the earliest values. That is, when a node is chosen from the second group it is because its last failure occurred before the failure of any other node. If this chosen node fails again, the timestamp of the last failure would be updated, thus the next time nodes are sorted, then it will be the last one. The idea behind this design is to prevent the system from selected systematically the same malfunctioning node if others are available for reconfiguration.

5. Learning Algorithm

The Hybrid GFS described is flexible enough to behave as desired in order to suit the cluster administrator preferences. However, this requires every HGFS parameter to be properly tuned, which

is a complex task due to the presence of multiple conflicting objectives and to the huge amount of combinations that renders infeasible an extensive search. In particular, every instance or configuration for the previous HGFS is the combination of the following parameters:

$$(t_{min}, t_{max}, n_{min}, n_{max}, w_1, \dots, w_N) \quad (2)$$

To address this problem, the EECluster learning algorithm uses multi-objective evolutionary algorithms (MOEAs) to find the parameters defining the HGFS by optimizing a fitness function consisting in three conflicting criteria: the quality of service (QoS), the energy saved, and the number of node reconfigurations. Specifically, EECluster uses the MOEA Framework [52] implementation of the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [53] to obtain a Pareto Efficient Frontier from which the administrator can choose a suitable HGFS configuration. The Pareto Efficient Frontier is the set of configurations for the HGFS obtained in the experiment that are not worse than any other configuration in all components of the fitness function simultaneously. Every configuration in the Pareto Efficient Frontier is known to be “non-dominated”. As can be seen, the result of the learning algorithm is not a single optimal solution (configuration for the HGFS) but rather a set of optimal configurations from which an expert human can pick the one that is best given his or her preferences. The reason for this is that there is no optimal solution because the three objectives involved are in conflict and attempts to apply any form of weighted sorting of the solutions obtained would lead to an inaccurate model of the preferences of the administrator.

For a given set of n jobs, where the j -th job ($j = 1 \dots n$) is scheduled to start at time $tsch_j$, but effectively starts at time ton_j and stops at time $toff_j$, the quality of service in an HPC cluster reflects the amount of time that each job has to wait before it is assigned its requested resources. Once the job starts its execution, it will not be halted; thus, we focus only on its waiting time. Because jobs do not last the same amount of time, their waiting in the queue is better expressed as a ratio considering their execution time. It is noteworthy that the execution times of the job can differ greatly since they range from seconds to weeks or months. This can potentially lead to situations where very short jobs must wait over a hundred times their execution timespan, distorting the measurement of the quality of service and depicting inaccurately the cluster performance. Because of this, the 90 percentile is used instead of average:

$$QoS = \min \left\{ p : \left| \left\{ j \in 1 \dots n : \frac{ton_j - tsch_j}{toff_j - ton_j} \leq p \right\} \right| > 0.9n \right\} \quad (3)$$

where $|A|$ is the cardinality of the set A .

The energy saved is measured as the amount of watts-hour that were prevented from being wasted by shutting down idle nodes. Let c be the number of nodes, let $state(i, t)$ be 1 if the i -th node ($i = 1 \dots c$) is powered at time t , and 0 otherwise, let the time scale be the lapse between $tini = \min_j \{tsch_j\}$ and $tend = \max_j \{toff_j\}$. Lastly, let $power_{idle}(i)$ be the power consumption measured in watts of the i -th node when it is at idle state. Then,

$$\text{Energy saved} = \sum_{i=1}^c power_{idle}(i) \cdot (tend - tini) - \sum_{i=1}^c power_{idle}(i) \int_{tini}^{tend} state(i, t) dt. \quad (4)$$

The node reconfigurations is the number of times that a node has been powered on or off. Let $nd(i)$ be the number of discontinuities of the function $state(i, t)$ in the time interval $t \in (tini, tend)$:

$$\text{Reconfigured nodes} = \sum_{i=1}^c nd(i) \quad (5)$$

The mission of the NSGA-II algorithm is to obtain a set of non-dominated configurations for the HGFS, guided by the previous fitness function, whose values are calculated by running a cluster simulation with a given number of nodes, slots, and job records, as seen in Figure 3.

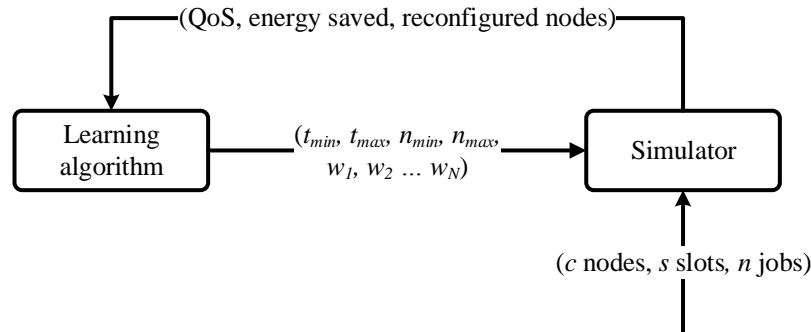


Figure 3. EECluster learning process. QoS: Quality of Service.

6. Experimental Results

In order to provide a sound answer on whether the decision-making mechanism has the required flexibility to perform correctly and suit any desired working mode, it must be tested in a range of cluster scenarios which together can build a significant representation of real-world clusters. To do so, a combination of synthetically-generated and actual cluster workloads from the Scientific Modelling Cluster (CMS) of the University of Oviedo [54] were used. Synthetic workloads represent four different scenarios with an increasing degree of fluctuation in terms of job arrival rates, each one spanning 24 months. Job arrivals in each scenario follow a Poisson process with the λ values shown in Table 1, and job run times are distributed exponentially with rate $\lambda = 10^{-5}$ s in all scenarios. As can be seen in the table, scenario 1 exhibits a cluster with a stable and sustained workload where all hours of the year have the same job arrival pattern. Scenario 2 adds a distinction between working, non-working, and weekend hours. Scenario 3 adds a substantial variation in the arrival rates depending on the week of the month, and scenario 4 increases this variation even more. On the other hand, the workloads from the CMS cluster consist of 2907 jobs spanned over 22 months. This real-world cluster, built from three independent computing clusters and five transversal queues using PBS as Resource Management System (RMS), can accurately show a very common activity pattern in most HPC clusters.

Table 1. Poisson process of job arrivals in each scenario.

Scenario	Day of Week	Hour Range	Week of Year	λ Value
1	All	All	All	2×10^{-4} s
2	Monday–Friday	8:00–20:00	All	2×10^{-4} s
	Saturday–Sunday	8:00–20:00	All	2×10^{-5} s
	Monday–Sunday	20:00–8:00	All	10^{-5} s
3	Monday–Friday	8:00–20:00	$w \% 5 = 0$	10^{-4} s
			$w \% 5 = 1$	2×10^{-4} s
			$w \% 5 = 2$	5×10^{-4} s
			$w \% 5 = 3$	5×10^{-4} s
			$w \% 5 = 4$	2×10^{-4} s
	Monday–Sunday	20:00–8:00	All	2×10^{-5} s
	Monday–Friday	8:00–20:00	All	10^{-5} s
4	Monday–Friday	8:00–20:00	$w \% 5 = 0$	10^{-4} s
			$w \% 5 = 1$	10^{-4} s
			$w \% 5 = 2$	5×10^{-4} s
			$w \% 5 = 3$	5×10^{-4} s
			$w \% 5 = 4$	10^{-4} s
	Monday–Sunday	20:00–8:00	All	2×10^{-5} s
	Monday–Friday	8:00–20:00	All	10^{-5} s

To assess the flexibility of the Hybrid GFS to suit the desired behaviour of the cluster administrator under different scenarios, a wide range of preferences must be tested. Given that these preferences are inherently subjective, a set of five different synthetic preferences were used in the experiments allowing us to measure to what extent the HGFS complies with them in each scenario as well to compare results across different scenarios. These synthetic preferences, denoted as “Hybrid GFS (QoS_{max}, reconfigurations_{max})”, determine that the HGFS configuration chosen in each case is the one with the maximum energy savings that has a service quality below QoS_{max} and a number of node reconfigurations not higher than reconfigurations_{max} over the timespan of the validation and test sets. It must be noted that values used for these parameters in the experiment differ slightly from one scenario to another in order to adapt to the different average loads and arrival rates in each scenario.

To measure the effect of applying each HGFS configuration, a cluster simulator has been developed for both training and testing, so that every model can be evaluated in the three criteria of the fitness function. The holdout method was used for validation, with a 50/25/25% split in training, validation, and test, respectively. Results obtained from the experiment using the test set of each workload are shown in the following tables. They are measured in terms of impact in the service quality and reduction in the cluster carbon footprint achieved according to the CO₂ emission factors for the current energy mix, and the number node reconfigurations. Observe that the “Energy saved” column depends on this energy mix that has been estimated according to the 370 kg CO₂/MWh emission factor reported by the Ministry of Agriculture, Food and Environment from the Government of Spain (“Ministerio de Agricultura, Alimentación y Medio Ambiente”) [43], but some of the greenhouse gas is not emitted at Spain; the primary energy consumption is assumed to be served by Spanish sources but secondary sources are located at the country where the compute nodes are manufactured. In this respect, Megawatt-hours of saved energy are understood as the amount of energy that would be saved according to the Spanish mix of generation; the actual amount of globally saved energy is possibly higher. Also, charts have been generated to depict the evolution over time of the active cluster resources and the requested slots by the queued and running jobs. In particular, results for

scenario 1 are displayed in Table 2 and in Figure 4, scenario 2 in Table 3 and in Figure 5, scenario 3 in Table 4 and in Figure 6, and scenario 4 in Table 5 and in Figure 7. Lastly, results obtained for the CMS cluster recorded workloads are displayed in Table 6 and in Figure 8.

Table 2. Experiment results for the test set of scenario 1.

Scenario 1 Test Set					
	QoS	Energy Saved (%)	Energy Saved (MWh)	Carbon Reduction (MtCO ₂)	Reconfigurations
Hybrid GFS (0.00, 1000)	0.00×10^0	28.62%	8.30	3.07	487
Hybrid GFS (0.005, 1250)	4.25×10^{-3}	33.19%	9.63	3.56	945
Hybrid GFS (0.01, 1500)	9.32×10^{-3}	35.63%	10.34	3.82	1412
Hybrid GFS (0.015, 2000)	1.35×10^{-2}	37.16%	10.78	3.99	1921
Hybrid GFS (0.02, 3000)	1.92×10^{-2}	38.50%	11.17	4.13	2835

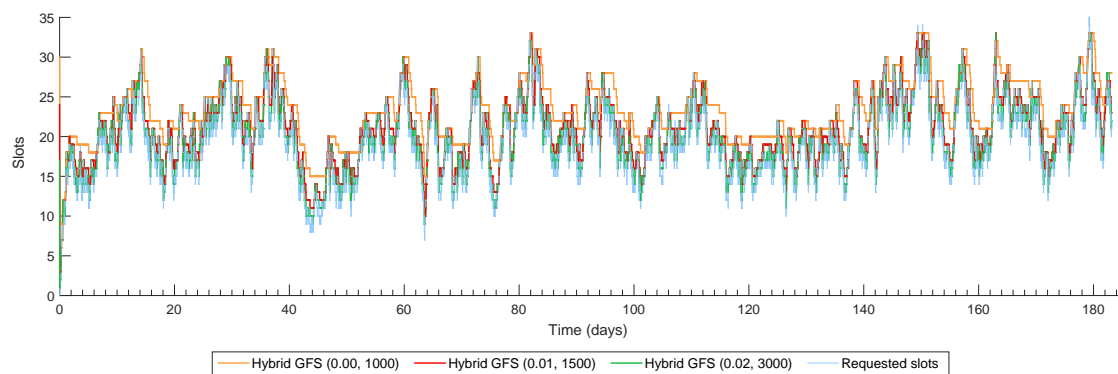


Figure 4. Cluster simulation trace for the test set of scenario 1. GFS: Genetic Fuzzy System.

Table 3. Experiment results for the test set of scenario 2.

Scenario 2 Test Set					
	QoS	Energy Saved (%)	Energy Saved (MWh)	Carbon Reduction (MtCO ₂)	Reconfigurations
Hybrid GFS (0.00, 500)	0.00×10^0	60.07%	17.43	6.45	218
Hybrid GFS (0.005, 500)	4.88×10^{-3}	65.46%	18.99	7.03	384
Hybrid GFS (0.01, 750)	9.45×10^{-3}	69.50%	20.17	7.46	635
Hybrid GFS (0.015, 1000)	1.43×10^{-2}	71.61%	20.78	7.69	838
Hybrid GFS (0.02, 1500)	1.84×10^{-2}	73.44%	21.31	7.88	1082

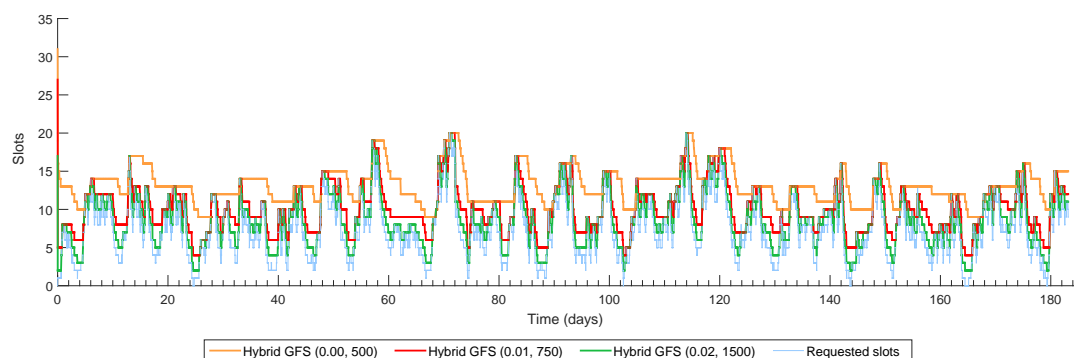


Figure 5. Cluster simulation trace for the test set of scenario 2.

Table 4. Experiment results for the test set of scenario 3.

Scenario 3 Test Set					
	QoS	Energy Saved (%)	Energy Saved (MWh)	Carbon Reduction (MtCO ₂)	Reconfigurations
Hybrid GFS (0.00, 500)	0.00×10^0	27.99%	8.12	3.00	194
Hybrid GFS (0.005, 500)	2.19×10^{-3}	41.53%	12.05	4.46	342
Hybrid GFS (0.01, 750)	8.12×10^{-3}	50.73%	14.72	5.45	671
Hybrid GFS (0.015, 1000)	1.17×10^{-2}	56.02%	16.25	6.01	1006
Hybrid GFS (0.02, 1500)	1.69×10^{-2}	59.44%	17.25	6.38	1334

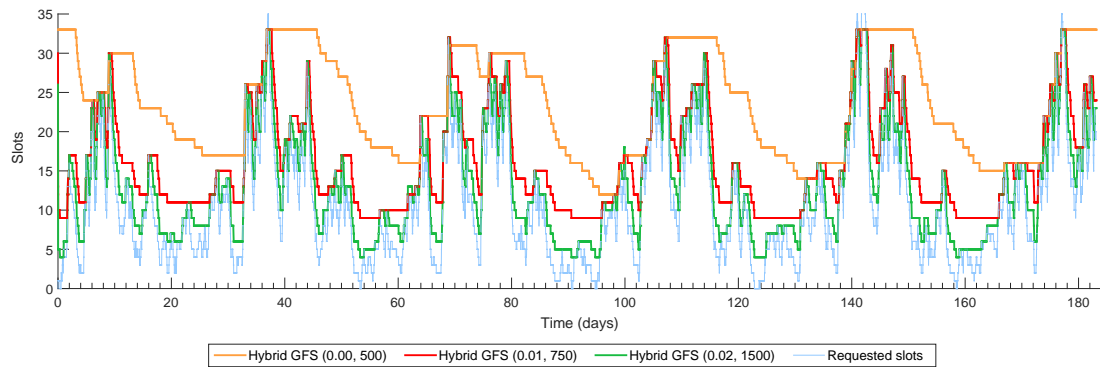


Figure 6. Cluster simulation trace for the test set of scenario 3.

Table 5. Experiment results for the test set of scenario 4.

Scenario 4 Test Set					
	QoS	Energy Saved (%)	Energy Saved (MWh)	Carbon Reduction (MtCO ₂)	Reconfigurations
Hybrid GFS (0.00, 500)	0.00×10^0	25.00%	7.25	2.68	173
Hybrid GFS (0.005, 500)	2.48×10^{-3}	46.09%	13.37	4.95	367
Hybrid GFS (0.01, 750)	4.23×10^{-3}	51.50%	14.94	5.53	473
Hybrid GFS (0.015, 1000)	1.43×10^{-2}	60.06%	17.42	6.45	1010
Hybrid GFS (0.02, 1500)	1.92×10^{-2}	63.51%	18.43	6.82	1253

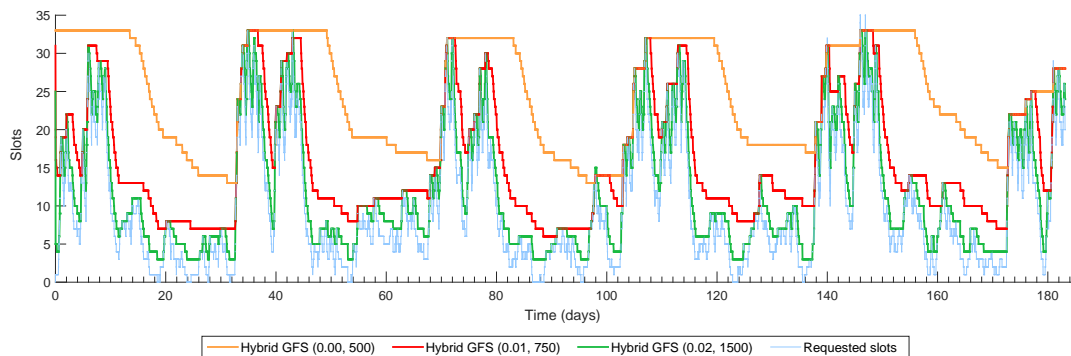


Figure 7. Cluster simulation trace for the test set of scenario 4.

Table 6. Experiment results for the test set of the Scientific Modelling Cluster (CMS) workload records.

CMS Cluster Test Set					
	QoS	Energy Saved (%)	Energy Saved (MWh)	Carbon Reduction (MtCO ₂)	Reconfigurations
Hybrid GFS (0.00, 100)	0.00×10^0	46.67%	13.38	4.95	42
Hybrid GFS (0.05, 500)	4.46×10^{-2}	64.26%	18.42	6.82	293
Hybrid GFS (0.10, 500)	6.82×10^{-2}	68.60%	19.67	7.28	361
Hybrid GFS (0.25, 750)	1.49×10^{-1}	70.34%	20.17	7.46	363
Hybrid GFS (0.50, 750)	1.95×10^{-1}	75.54%	21.66	8.01	590

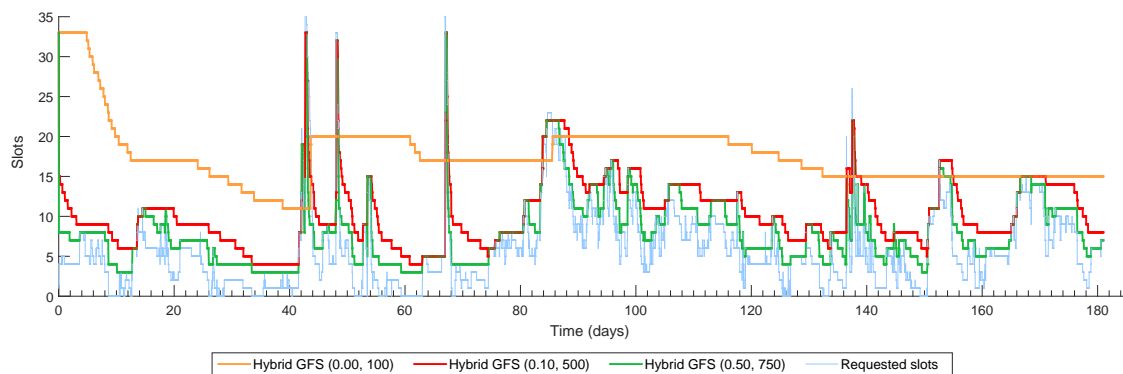


Figure 8. Cluster simulation trace for the test set of the CMS cluster workload records.

As can be seen from these results, in every cluster scenario used in the experiments, the learning algorithm found a configuration for the HGFS that achieves significant energy savings without any noticeable impact in service quality. Also, additional configurations were found that comply with the synthetic administrator preferences defined, increasing energy savings while strictly complying with the constraints set in the aforementioned preferences in terms of QoS and node reconfigurations. It should be noted that the five configurations displayed in the previous tables are only a small selection of the vast set obtained in the Pareto Efficient Frontier, and many other solutions are available that can save even more energy at the cost of a higher penalty in service quality.

The experiments also show that the results obtained differ significantly depending on the characteristics of the workload. In scenario 1, the regular job arrival rate depicts a workload where the distances between peaks are very short and valleys tend to be shallow. This leads to HGFS configurations with a higher average number of reconfigurations and a relatively low amount of saved energy. Also, as can be seen in Table 2, rising the degree of tolerance for impact in QoS from 0.0 to 0.02 and node reconfigurations up to 3000 only allows an increase in the overall energy savings of 9.88%. Results improve progressively as job arrival patterns vary over time and the workload becomes more irregular with deeper valleys and longer distances between peaks. For instance, scenario 2 allows an increase in energy savings of 13.37% between the HGFS configurations obtained for a QoS of 0.0 and a QoS of 0.02. In scenario 3, the difference grows up to 31.45%, and in scenario 4 reaches 38.51%.

These two last scenarios show important results since they represent the workload patterns most likely to occur in real-world HPC clusters. This can be verified by the results obtained using actual records from the CMS cluster of the University of Oviedo, where the workload is even sharper than in scenario 4, with energy savings between 46.67% and 75.54%, depending on the administrator preferences. These values can be translated to actual power savings ranging from 13.38 MWh to 21.66 MWh over the course of the test set, and figures of carbon reduction between 4.95 and 8.01 tonnes of CO₂.

7. Concluding Remarks

The EECluster tool has been designed to reduce the carbon footprint of HPC clusters by improving their energy efficiency. This software package implements the adaptive resource cluster technique in clusters running OGE/SGE or PBS/TORQUE as RMS, allowing for practical application in real-world scenarios, owing to the flexibility of its sophisticated machine-learned decision-making mechanism to comply with cluster administrator preferences. This mechanism, based on Computational Intelligence techniques, is learnt by means of multi-objective evolutionary algorithms to assure finding a suitable configuration that maximises energy savings within the tolerance region of the administrator in terms of service quality and node reliability.

Thorough experimental studies based on both synthetic and actual workloads from the Scientific Modelling Cluster of Oviedo University [54] provide empirical evidence of the ability of EECluster to deliver good results in multiple scenarios, supporting the adoption of EECluster to reduce the environmental impact of real world clusters.

Acknowledgments: This work has been partially supported by the Ministry of Economy and Competitiveness (“Ministerio de Economía y Competitividad”) from Spain/FEDER under grants TEC2012-38142-C04-04, TEC2015-67387-C4-3-R and TIN2014-56967-R and by the Regional Ministry of the Principality of Asturias under grant FC-15-GRUPIN14-073.

Author Contributions: All authors participated equally to this research work from its inception, with L.S. providing the design of the decision-making mechanism and learning algorithm, J.R. providing the architectural design of the solution for OGE/SGE and PBS/TORQUE clusters, and A.C.-F. carrying out software implementation and experimentation. All authors also prepared, reviewed and agreed the manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Acronyms

HPC	High Performance Computing
RMS	Resource Management System
OGE	Oracle Grid Engine/Open Grid Engine
SGE	Sun Grid Engine/Son of Grid Engine
PBS	Portable Batch System
TORQUE	Terascale Open-source Resource and QUEue Manager
DBMS	Database Management System
IPMI	Intelligent Platform Management Interface
WOL	Wake On Lan
SSH	Secure SHell
HGFS	Hybrid Genetic Fuzzy System
TSK	Tagaki-Sugeno-Kang
QoS	Quality of Service
NSGA-II	Non-dominated Sorting Genetic Algorithm-II
MOEAs	MultiObjective Evolutionary Algorithms

References

1. Delforge, P.; Whitney, J. *Issue Paper: Data Center Efficiency Assessment Scaling up Energy Efficiency across the Data Center Industry: Evaluating Key Drivers and Barriers*; Technical Report; Natural Resources Defense Council (NRDC): New York, NY, USA, 2014.
2. U.S. Environmental Protection Agency. *Report to Congress on Server and Data Center Energy Efficiency Public Law 109-431*; Technical Report; ENERGY STAR Program: Washington, DC, USA, 2007.
3. Ebberts, M.; Archibald, M.; Fonseca, C.F.F.D.; Griffel, M.; Para, V.; Searcy, M. *Smarter Data Centers: Achieving Greater Efficiency*; Technical Report; IBM Redpaper: Research Triangle Park, NC, USA, 2011.
4. The Economist Intelligence Unit. *IT and the Environment: A New Item on the CIO's Agenda?* Technical Report; The Economist: London, UK, 2007.
5. Gartner. *Gartner Estimates ICT Industry Accounts for 2 Percent of Global CO₂ Emissions*; Gartner: Stamford, CT, USA, 2007.
6. Forrest, W.; Kaplan, J.M.; Kindler, N. *Data Centers: How to Cut Carbon Emissions and Costs*; Technical Report; McKinsey & Company: New York, NY, USA, 2008.
7. Valentini, G.L.; Lassonde, W.; Khan, S.U.; Min-Allah, N.; Madani, S.A.; Li, J.; Zhang, L.; Wang, L.; Ghani, N.; Kolodziej, J.; *et al.* An overview of energy efficiency techniques in cluster computing systems. *Clust. Comput.* **2013**, *16*, 3–15.
8. Haring, R.; Ohmacht, M.; Fox, T.; Gschwind, M.; Satterfield, D.; Sugavanam, K.; Coteus, P.; Heidelberger, P.; Blumrich, M.; Wisniewski, R.; *et al.* The IBM blue gene/Q compute chip. *IEEE Micro* **2012**, *32*, 48–60.

9. IBM Systems and Technology Group. *IBM System Blue Gene/Q*; Technical Report; IBM: Somers, NY, USA, 2011.
10. Hsu, C.H.; Kremer, U. The design, implementation, and evaluation of a compiler algorithm for CPU energy reduction. *ACM SIGPLAN Not.* **2003**, *38*, 38–48.
11. Hsu, C.H.; Feng, W.C. A Power-Aware Run-Time System for High-Performance Computing. In Proceedings of the ACM/IEEE SC 2005 Conference (SC '05), Seattle, WA, USA, 12–18 November 2005; IEEE: Washington, DC, USA, 2005; p. 1.
12. Freeh, V.W.; Lowenthal, D.K.; Pan, F.; Kappiah, N.; Springer, R.; Rountree, B.L.; Femal, M.E. Analyzing the Energy-Time Trade-off in High-Performance Computing Applications. *IEEE Trans. Parallel Distrib. Syst.* **2007**, *18*, 835–848.
13. Lim, M.; Freeh, V.; Lowenthal, D. Adaptive, Transparent Frequency and Voltage Scaling of Communication Phases in MPI Programs. In Proceedings of the ACM/IEEE SC 2006 Conference (SC '06), Tampa, FL, USA, 11–17 November 2006; IEEE: Tampa, FL, USA, 2006; p. 14.
14. Chen, Y.; Zeng, Y. Automatic energy status controlling with dynamic voltage scaling in poweraware high performance computing cluster. In Proceedings of the Parallel and Distributed Computing, Applications and Technologies (PDCAT), Gwangju, Korea, 20–22 October 2011; IEEE: Gwangju, Korea, 2011; pp. 412–416.
15. Ge, R.; Feng, X.; Feng, W.C.; Cameron, K.W. CPU MISER: A Performance-Directed, Run-Time System for Power-Aware Clusters. In Proceedings of the 2007 International Conference on Parallel Processing (ICPP 2007), Xi'an, China, 10–14 September 2007; IEEE: Xi'an, China, 2007; pp. 18–25.
16. Huang, S.; Feng, W. Energy-Efficient Cluster Computing via Accurate Workload Characterization. In Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, Shanghai, China; 18–21 May 2009; IEEE: Shanghai, China, 2009; pp. 68–75.
17. Chetsa, G.L.T.; Lefrvre, L.; Pierson, J.M.; Stolf, P.; Da Costa, G. A Runtime Framework for Energy Efficient HPC Systems without a Priori Knowledge of Applications. In Proceedings of the 2012 IEEE 18th International Conference on Parallel and Distributed Systems, Singapore, 17–19 December 2012; IEEE: Singapore, 2012; pp. 660–667.
18. Zong, Z.; Ruan, X.; Manzanares, A.; Bellam, K.; Qin, X. Improving Energy-Efficiency of Computational Grids via Scheduling. In *Handbook of Research on P2P and Grid Systems for Service-Oriented Computing*; Antonopoulos, N., Exarchakos, G., Li, M., Liotta, A., Eds.; IGI Global: Hershey, PA, USA, 2010; Chapter 22.
19. Zong, Z.; Nijim, M.; Manzanares, A.; Qin, X. Energy efficient scheduling for parallel applications on mobile clusters. *Clust. Comput.* **2007**, *11*, 91–113.
20. Bash, C.; Forman, G. Cool job allocation: Measuring the power savings of placing jobs at cooling-efficient locations in the data center. In Proceedings of the 2007 USENIX Annual Technical Conference on Proceedings of the USENIX Annual Technical Conference, Santa Clara, CA, USA, 17–22 June 2007; USENIX Association: Berkeley, CA, USA, 2007; pp. 29:1–29:6.
21. Tang, Q.; Gupta, S.K.S.; Varsamopoulos, G. Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Data Centers: A Cyber-Physical Approach. *IEEE Trans. Parallel Distrib. Syst.* **2008**, *19*, 1458–1472.
22. Alonso, P.; Badia, R.M.; Labarta, J.; Barreda, M.; Dolz, M.F.; Mayo, R.; Quintana-Orti, E.S.; Reyes, R. Tools for Power-Energy Modelling and Analysis of Parallel Scientific Applications. In Proceedings of the 2012 41st International Conference on Parallel Processing; IEEE: Pittsburgh, PA, USA, 10–13 September 2012; pp. 420–429.
23. Schubert, S.; Kostic, D.; Zwaenepoel, W.; Shin, K.G. Profiling software for energy consumption. In Proceedings of the 2012 IEEE International Conference on Green Computing and Communications, GreenCom 2012, Besancon, France, 20–23 November 2012; IEEE: Besancon, France; pp. 515–522.
24. Freeh, V.W.; Lowenthal, D.K. Using multiple energy gears in MPI programs on a power-scalable cluster. In Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming—PPoPP '05, Chicago, IL, USA, 15–17 June 2005; ACM Press: New York, NY, USA, 2005; pp. 164–173.
25. Li, D.; Nikolopoulos, D.S.; Cameron, K.; de Supinski, B.R.; Schulz, M. Power-aware MPI task aggregation prediction for high-end computing systems. In Proceedings of the 2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS), Atlanta, GA, USA, 19–23 April 2010; IEEE: Atlanta, GA, USA, 2010; pp. 1–12.

26. Xian, C.; Lu, Y.H.; Li, Z. A programming environment with runtime energy characterization for energy-aware applications. In Proceedings of the 2007 International Symposium on Low Power Electronics and Design—ISLPED '07, Portland, OR, USA, 27–29 August 2007; ACM Press: New York, NY, USA, 2007; pp. 141–146.
27. Pinheiro, E.; Bianchini, R.; Carrera, E.V.; Heath, T. Load balancing and unbalancing for power and performance in cluster-based systems. In Proceedings of the Workshop on Compilers and Operating Systems for Low Power, Barcelona, Spain, 9 September 2001; Volume 180, pp. 182–195.
28. Das, R.; Kephart, J.O.; Lefurgy, C.; Tesauro, G.; Levine, D.W.; Chan, H. Autonomic multi-agent management of power and performance in data centers. In Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems: Industrial Track—AAMAS '08, Estoril, Portugal, 2008; International Foundation for Autonomous Agents and Multiagent Systems: Richland, SC, USA, 2008; pp. 107–114.
29. Elnozahy, E.; Kistler, M.; Rajamony, R. Energy-efficient server clusters. In *Power-Aware Computer Systems*; Springer-Verlag: Berlin/Heidelberg, Germany, 2003; pp. 179–197.
30. Berral, J.L.; Goiri, Í.; Nou, R.; Julià, F.; Guitart, J.; Gavalda, R.; Torres, J. Towards energy-aware scheduling in data centers using machine learning. In Proceedings of the 1st International Conference on Energy Efficient Computing and Networking—e-Energy '10, Passau, Germany, 13–15 April 2010; ACM Press: New York, NY, USA, 2010; Volume 2, p. 215.
31. Lang, W.; Patel, J.M.; Naughton, J.F. On energy management, load balancing and replication. *ACM SIGMOD Rec.* **2010**, *38*, 35–42.
32. Entrialgo, J.; Medrano, R.; García, D.F.; García, J. Autonomic power management with self-healing in server clusters under QoS constraints. In *Computing*; Springer: Vienna, Austria, 2015; pp. 1–24.
33. VMware. VMware Distributed Power Management Concepts and Use. Available online: <http://www.vmware.com/files/pdf/Distributed-Power-Management-vSphere.pdf> (accessed on 1 March 2016).
34. Citrix Systems. XenServer - Server Virtualization and Hypervisor Management. Available online: <http://www.citrix.com/products/xenserver/overview.html> (accessed on 1 March 2016).
35. Alvarruiz, F.; de Alfonso, C.; Caballer, M.; Hernández, V. An Energy Manager for High Performance Computer Clusters. In Proceedings of the 2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications, Leganes, Spain, 10–13 July 2012; IEEE: Leganés, Spain, 2012; pp. 231–238.
36. Dolz, M.F.; Fernández, J.C.; Iserte, S.; Mayo, R.; Quintana-Ortí, E.S.; Cotallo, M.E.; Díaz, G. EnergySaving Cluster experience in CETA-CIEMAT. In Proceedings of the 5th Iberian GRID Infrastructure Conference, Santander, Spain, 8–10 June 2011.
37. Kiertscher, S.; Zinke, J.; Gasterstädt, S.; Schnor, B. Cherub: Power Consumption Aware Cluster Resource Management. In Proceedings of the 2010 IEEE/ACM International Conference on Cyber, Physical and Social Computing (CPSCom) Green Computing and Communications (GreenCom), Hangzhou, China, 18–20 December 2010; pp. 325–331.
38. Cocaña-Fernández, A.; Ranilla, J.; Sánchez, L. Energy-efficient allocation of computing node slots in HPC clusters through parameter learning and hybrid genetic fuzzy system modeling. *J. Supercomput.* **2014**, *71*, 1163–1174.
39. Cocaña-Fernández, A.; Sánchez, L.; Ranilla, J. Leveraging a predictive model of the workload for intelligent slot allocation schemes in energy-efficient HPC clusters. *Eng. Appl. Artif. Intell.* **2016**, *48*, 95–105.
40. Cocaña-Fernández, A.; Ranilla, J.; Sánchez, L. Energy-Efficient Allocation of Computing Node Slots in HPC Clusters through Evolutionary Multi-Criteria Decision Making. In Proceedings of the 14th International Conference on Computational and Mathematical Methods in Science and Engineering, CMMSE 2014, Cádiz, Spain, 3–7 July 2014; pp. 318–330.
41. Hendrik, A.; Bidwell, V.R. *Measuring Eco-Efficiency: A Guide to Reporting Company Performance*; World Business Council for Sustainable Development: Geneva, Switzerland, 2000.
42. Deng, L.; Babbitt, C.W.; Williams, E.D. Economic-balance hybrid LCA extended with uncertainty analysis: Case study of a laptop computer. *J. Clean. Prod.* **2011**, *19*, 1198–1206.
43. Ministerio de Agricultura, Alimentación y Medio Ambiente. *Factores de Emisión: Registro de Huella de Carbono, Compensación y Proyectos de Absorción de Dióxido de Carbono*; Technical Report; Ministerio de Agricultura, Alimentación y Medio Ambiente: Madrid, Spain, 2015.

44. Yeo, C.S.; Buyya, R.; Hossein, P.; Rasit, E.; Graham, P.; Sommers, F. Cluster Computing: High-Performance, High-Availability, and High-Throughput Processing on a Network of Computers. In *Handbook of Nature-Inspired and Innovative Computing*; Zomaya, A., Ed.; Springer US: New York, NY, USA, 2006; pp. 521–551.
45. National Science Foundation. *Advisory Committee for Cyberinfrastructure Task Force on Grand Challenges*; Technical Report; National Science Foundation: Arlington, VA, USA, 2011.
46. Cacheiro, J. *Analysis of Batch Systems*; Technical Report; CESGA: Santiago de Compostela, Galicia, Spain, 2014.
47. IRPC Group. EECluster: A Software Tool to Efficiently Manage the Energy Consumption of HPC Clusters. Available online: <http://pirweb.edv.uniovi.es/eecluster> (accessed on 1 March 2016).
48. SourceForge. EECluster download | SourceForge.net. Available online: <http://sourceforge.net/projects/eecluster/> (accessed on 1 March 2016).
49. Cocaña-Fernández, A.; Sánchez, L.; Ranilla, J. A software tool to efficiently manage the energy consumption of HPC clusters. In Proceedings of the 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Istanbul, Turkey, 2–5 August 2015; pp. 1–8.
50. Ishibuchi, H.; Nakashima, T.; Nii, M. *Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining (Advanced Information Processing)*; Springer-Verlag New York, Inc.: Secaucus, NJ, USA, 2004.
51. Takagi, T.; Sugeno, M. Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. Syst. Man Cybern.* **1985**, SMC-15, 116–132.
52. MOEA Framework, a Java library for multiobjective evolutionary algorithms. Available online: <http://moeaframework.org/> (accessed on 1 March 2016).
53. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, 6, 182–197.
54. University of Oviedo. Scientific Modelling Cluster. Available online: <http://cms.uniovi.es> (accessed on 1 March 2016).



© 2016 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons by Attribution (CC-BY) license (<http://creativecommons.org/licenses/by/4.0/>).