Improving EECluster to optimize the carbon footprint and operating costs of HPC clusters

Alberto Cocaña-Fernández
Departamento de Informática
Universidad de Oviedo
Gijón, Spain
Email: cocanaalberto@gmail.com

Emilio San José Guiote
Departamento de Informática
Universidad de Oviedo
Gijón, Spain
Email: emilio.sanjose
@outlook.com

José Ranilla
Departamento de Informática
Universidad de Oviedo
Gijón, Spain
Email: ranilla@uniovi.es

Luciano Sánchez
Departamento de Informática
Universidad de Oviedo
Gijón, Spain
Email: luciano@uniovi.es

***Abstract*—High Performance Computing Clusters (HPCCs) are essential platforms for solving up-to-date challenges through parallel and distributed applications. Nevertheless, HPCCs have an important economic and environmental impact owing to the large amounts of energy required for their operation. In this work, an improved version of our EECluster software focused on reducing the operating costs and emissions of HPCCs is presented. Multicriteria learning algoritms are leveraged to jointly optimize cluster performance, direct and indirect operating costs both from an economic and an environmental standpoint. EECluster is a mature software that can be integrated with the resource management systems OGE/SGE and PBS/TORQUE.**

***Index Terms*—Energy-efficient cluster computing, Multicriteria decision making, Evolutionary algorithms**

I. Introduction

High Performance Computing Clusters (HPCCs) are the core of today's supercomputers (see Top500[1] and The Green500[2]) owing to their price, performance and the availability of variety of tools for parallel and distributed computing. The economic impact of HPCCs in IT companies is high. First, there are the direct costs, mainly electricity [1], [2]. Second, there are indirect costs, including life-cycle related events, such as replacing deteriorated or obsolete computing nodes and secondary equipment. Third, there are environment-related costs that can be directly attributed to a centre, such as greenhouse gas emissions. Greenhouse emissions depend, in turn, on the proportion of renewable generation at the points where the electricity is being produced; these can and are easily monetized. Fourth, there are also indirect environment-related costs, such as the cost of disposing of and recycling e-waste, or the cost of the energy devoted to the manufacture of replacement parts, minus the savings in reuse and the second life of equipment. With all these factors considered, the impact of HPCCs is comparable to that of aviation industry. In particular, HPCCs are responsible for the emission of 100 million metric tons of CO_2 every year [3].

These figures show accurately the magnitude of the problem at hand, which itself is the bottleneck restraining the expansion of HPCCs. Hence, maximizing the efficiency of HPCCs to reduce operation costs, carbon footprint and to improve their reliability is an overriding need. In recent years, many efforts have been made to achieve a energy-efficient cluster computing, following both static and dynamic approaches. Static approaches focus on the development of hardware to maximize the FLOPS/watt ratio, while the dynamic ones seek to adapt the cluster to the current workload by turning off/on some components, or reducing/increasing their speed (see [4] for a more detailed introduction).

This work focuses on adaptive resource clusters, a method consisting of the automatic reconfiguration of the cluster resources to fit the workload at every moment by switching on/off its compute nodes, thus saving energy whenever these are idle. This technique already has been applied to Load-Balancing clusters (e.g., [5], [6]), and in commercial hypervisors such as VMware vSphere[3]. Recently, software tools implementing this technique in HPCCs have also been developed [7]–[9].

Notwithstanding this, the dependence of this software tools of closed sets of operational rules following a "one size fits all" approach inherently includes a compromise between the different working modes for each cluster environment. This fundamentally limits their applicability to real world clusters which require sufficient flexibility to find an optimal balance between service quality, energy savings and node thrashing within the cluster administrator tolerances. Consequently, we introduced in [10] the software tool EECluster to improve the efficiency of HPCCs while complying with administrator preferences. EECluster achieves this due to the Hybrid Genetic Fuzzy System (HGFS) that uses as decision-making mechanism, combining expert-defined knowledge with computer-generated rules elicited from past workload records from the cluster.

In our previous works, trade-offs were sought between the quality of service, the amount of time that nodes are powered off, and the number of power-on/off cycles. Different approximations were used for converting power-on/off times to money, but indirect environment-related costs were disregarded. To be specific, in Reference [11], a model was used to assess the impact of EECluster on the *eco-efficiency* of the

[1] http://www.top500.org/

[2] http://www.top500.org/green500

[3] https://labs.vmware.com/vmtj/vmware-distributed-resource-management-design-implementation-and-lessons-learned

cluster. The present work extends reference [11]: we leverage a more sophisticated and exhaustive model that covers a wider range of environmental aspects and balances service quality and power consumption with all indirect costs, including hardware failures and subsequent replacements, measured in both monetary units and carbon emissions.

The remainder of the paper is as follows. Section II explains the architecture of the EECluster tool and its decision-making mechanism. Section III details how the learning process is conducted. Section IV explains how EECluster can be installed and used. Section V discusses use cases and shows some experimental results. Section VI concludes the paper.

II. Architecture

EECluster is a software to convert HPCCs running a Resource Management System (RMS) into adaptive resource clusters that automatically reshape its compute nodes to suit the current workload in order to improve their eco-efficency. EECluster is comprised of a service (*eeclusterd*) and an administration dashboard, having the former periodically retrieve information regarding the cluster status and then using the decision-making mechanism to reconfigure the compute nodes by issuing a set of power-on or shutdown commands. The administration dashboard is a web application deployed on an application server that allows the cluster administrator to remotely access the information of the cluster status, nodes, job records, et cetera. Figure 1 provides a high-level overview of the system components. A detailed description of the EECluster components and the working cycle of *eeclusterd* can be seen in [10].

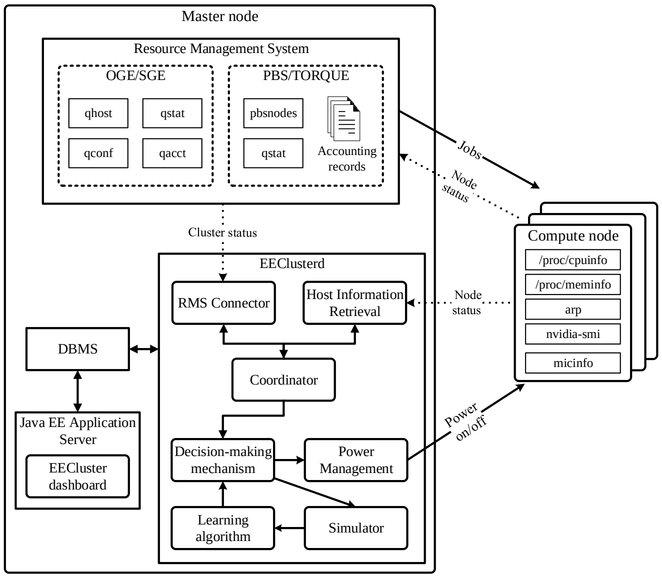


Fig. 1. EECluster: High-level overview of the system components

EECluster decision-making mechanism is implemented using Genetic Fuzzy Systems (GFS) to combine human expertise and computer-generated rules. The first Knowledge Base (KB) consists of a set of human-generated rules (adapted from Reference [8]) and is aimed at increasing the number of compute nodes in the cluster when average waiting time or number of queued jobs are too high, and reducing in the opposite case, while maintaining a minimum number of nodes to run the queued jobs. The linguistic definition of these rules can be found in Reference [4].

The second KB consists of a set of fuzzy rules learnt from past data, in the form of a zero-order Takagi-Sugeno-Kang (TSK) model [12], [13] and are based on a set of fuzzy thresholds defining the maximum waiting time for each node. The intuitive idea under this design is that each rule defines the degree of truth of the assert "the i-th node must be switched off", having the degrees of truth of each rule combined with the total number of active nodes determined from this aggregated value. The linguistic description of this KB applied to each compute node is as follows:

if t is $\widetilde{T}_1$ **then** off $= w_1$
if t is $\widetilde{T}_2$ **then** off $= w_2$
if $\cdots$ **then** $\cdots$
if t is $\widetilde{T}_N$ **then** off $= w_N$

where the fuzzy sets $\widetilde{T}_1, \ldots, \widetilde{T}_N$ have a triangular membership function and define a uniform fuzzy partition of the input variable t [12], and the defuzzified output of this model is

$$\mathrm{off}(t) = \frac{\sum_{r=1}^{N} \widetilde{T}_r(t) \cdot w_r}{\sum_{r=1}^{N} \widetilde{T}_r(t)}. \tag{1}$$

Lastly, the number of nodes that are powered off at a certain time is given by the sum of the outputs of the fuzzy model for all values of idle_i, where idle_i is the time that the i-th node has been at idle state:

$$\text{Powered off nodes} = \left\lfloor \sum_{h=1}^{c} \mathrm{off}(\mathrm{idle}_h) \right\rfloor. \tag{2}$$

III. Tuning the fuzzy rule set and the parameters of the expert-defined rules

The Hybrid GFS described in the preceeding section combines the performance achieved by Fuzzy Ruled-Based Systems with the robustness of expert systems when it comes to changes in workload patterns, thus avoiding the potential overtrain of the former in unforeseen scenarios while attaining great adaptability to the workload demand and flexibility to meet the preferences of the cluster administrators. However, in order for the Hybrid GFS to perform as required, both the fuzzy rule set and the parameters of the expert-defined rules must be tuned. An EECluster policy is defined by $2 \cdot N + 4$ parameters [4]:

$$(t_{min}, t_{max}, n_{min}, n_{max}, \widetilde{T}_1, \ldots, \widetilde{T}_N, w_1, \ldots, w_N) \tag{3}$$

These parameters are tuned through multiobjective evolutionary algorithms (MOEAs) in a distal supervised learning approach (Fig. 2). Specifically, EECluster uses the MOEAFramework implementation of these evolutionary algorithms (NSGA-II, SPEA2, SMPSO, etc.) [14].

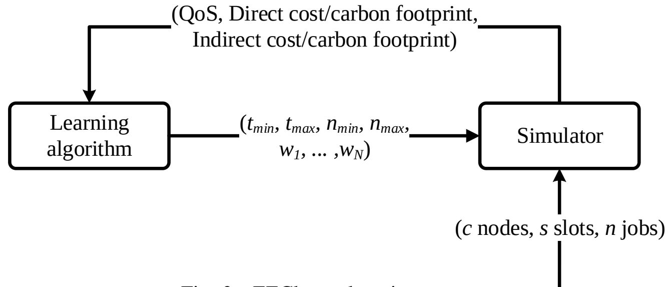

(QoS, Direct cost/carbon footprint, Indirect cost/carbon footprint)
Learning algorithm
(t_{min}, t_{max}, n_{min}, n_{max}, w_1, ... ,w_N)
Simulator
(c nodes, s slots, n jobs)

Fig. 2. EECluster learning process

Every individual in the population is expressed as a chain of parameters (recall Eq. 3), whose fitness is computed by running a cluster simulation, as shown in Fig. 2. Cluster simulations play the role of the training set in conventional learning algorithms, and depend on n jobs and c computing nodes each. The goal of the algorithm is to optimize a fitness function consisting of three conflicting criteria: the quality of service, the direct operating cost and the indirect operating cost due to hardware failures and replacements. It must be noted that in our previous works (see [10]) the direct and indirect costs where hinted by hardware-agnostic metrics such as the amount of time that a node is powered off or the number of node starts/stops cycles. Nevertheless, these metrics vaguely picture the effect for the cluster, as they lack a direct translation to energy consumed, monetary costs or carbon footprints, and this lack of information makes difficult for the cluster administrators to choose their preferred configuration. To improve the representativity of the metrics used in the fitness criteria, in this paper we measure the direct operating cost as the amount of energy consumed by the compute nodes measured in Kilowatts-hour (kWh) along with equivalent figures in terms of carbon footprint (Kg CO_2) and monetary cost (EUR). On the other hand, indirect costs are measured in terms of manufacturing costs of hardware replacements according to estimated failures rates under simulated stress conditions and power on/off cycles, as these are known to be the main factors affecting servers and disks reliability [15], [16]. In order to estimate the effective failure rates of both the computes nodes as a whole along with their hard disk drives, the models described in [15], [17], [18] are used to approximate the effect of temperature rise and thermal cycling in processors, and increased disks start/stop frequencies.

We will assume that the Quality of Service (QoS) in a HPCC depends only on the waiting time of the jobs before their requested resources are assigned (i.e. jobs cannot be interrupted). The waiting time is not directly measured in seconds but it is divided by the job execution time. Let us suppose that there are n jobs and the j-th job ($j = 1 \dots n$) is scheduled to start at time tsch_j but it actually starts at ton_j and stops at time toff_j. QoS is defined as follows [10]:

$$\mathrm{QoS} = \min \left\{ p : ||\{ j \in 1 \dots n : \frac{\mathrm{ton}_j - \mathrm{tsch}_j}{\mathrm{toff}_j - \mathrm{ton}_j} \leq p \}|| > \frac{9n}{10} \right\} \quad (4)$$

where $||A||$ is the cardinality of the set A. It is remarked that the 90 percentile is used instead of average, because outliers are possible.

Let c be the number of nodes, let $\mathrm{state}(i,t)$ be 1 if the i-th node ($i = 1 \dots c$) is powered on at time t and 0 otherwise, let $\mathrm{power}(i,t)$ be the power consumption of the i-th node at time t. Lastly, let the time scale be the lapse between tini=$\min_j\{\mathrm{sch}_j\}$ and tend= $\max_j\{\mathrm{toff}_j\}$, then the overall cluster power consumption is measured by adding the power consumption of every compute node:

$$\text{Power consumption} = \sum_{i=1}^{c} \int_{\mathrm{tini}}^{\mathrm{tend}} \frac{\mathrm{power}(i,t)}{1000} \cdot \mathrm{state}(i,t)\mathrm{d}t.$$

The direct operating costs measured in both monetary units and carbon footprint are computed by multiplying the overall cluster power consumption by a translation factor according to the energy mix of the cluster power supply. Let $\rho_{\mathrm{CO_2}}$ be the amount of Kg of CO_2 emitted per kWh generated, and $\rho_{€}$ the electricity charge measured in euros per kWh, then the direct cost are calculated as:

$$\text{Direct monetary cost} = \text{Power consumption} \cdot \rho_{€}. \quad (5)$$

$$\text{Direct carbon footprint} = \text{Power consumption} \cdot \rho_{\mathrm{CO_2}}. \quad (6)$$

The indirect operating costs for each compute node are approximated by multiplying the probability of failure for the node and its disks during the simulation time lapse $(\mathrm{tini}, \mathrm{tend})$ by their manufacturing cost (monetary and carbon footprint), and by the remaining useful life of the device. These partial results are then added to estimate the costs for the whole cluster:

$$\begin{aligned}\text{Ind. monetary cost} = \sum_{i=1}^{c} &(1 - \mathrm{ulife}^i) \cdot (\mathrm{SFR}^i_{\mathrm{srv}} \cdot \mathrm{msrv}^i_{€} \\ &+ \mathrm{SFR}^i_{\mathrm{HDD}} \cdot \mathrm{num}^i_{\mathrm{HDD}} \cdot \mathrm{mHDD}^i_{€}).\end{aligned} \quad (7)$$

$$\begin{aligned}\text{Ind. carbon footp.} = \sum_{i=1}^{c} &(1 - \mathrm{ulife}^i) \cdot (\mathrm{SFR}^i_{\mathrm{srv}} \cdot \mathrm{msrv}^i_{\mathrm{CO_2}} \\ &+ \mathrm{SFR}^i_{\mathrm{HDD}} \cdot \mathrm{num}^i_{\mathrm{HDD}} \cdot \mathrm{mHDD}^i_{\mathrm{CO_2}}).\end{aligned} \quad (8)$$

where $(1 - \mathrm{ulife}^i)$ is the remaining percentage of useful life for the i-th node before it would be replaced anyway due to its obsolescence. $\mathrm{SFR}^i_{\mathrm{srv}}$ and $\mathrm{SFR}^i_{\mathrm{HDD}}$ are the Simulation Failure Rates for the i-th node and for one of its disks, and represent the estimated probability of failure of these devices during the simulation time lapse. $\mathrm{msrv}^i_{€}$/$\mathrm{msrv}^i_{\mathrm{CO_2}}$ and $\mathrm{mHDD}^i_{€}$/$\mathrm{mHDD}^i_{\mathrm{CO_2}}$ are, respectively, the manufacturing costs in euros and carbon footprint for the i-th node and for one of its disks. $\mathrm{num}^i_{\mathrm{HDD}}$ is the number of disks of the i-th node.

The effective failure rates during the simulation for the i-th compute node as a whole are approximated by:

$$\mathrm{SFR}^i_{\mathrm{srv}} = \frac{\mathrm{hours}(i, \mathrm{tini}, \mathrm{tend})}{\mathrm{MTTF}^i_{\mathrm{srv}}}.$$

$$\begin{aligned}\mathrm{MTTF}^i_{\mathrm{srv}} = \frac{1}{\mathrm{on}(i, \mathrm{tini}, \mathrm{tend})} \int_{\mathrm{tini}}^{\mathrm{tend}} &([\mathrm{MTTF}^i_{\mathrm{base}} \cdot (1 - \mathrm{hload}(i,t)) \\ &+ \frac{\mathrm{MTTF}^i_{\mathrm{base}}}{AF_i} \cdot \mathrm{hload}(i,t)] \cdot \mathrm{state}(i,t)\mathrm{d}t) - \Delta\mathrm{MTTF}^i_{\mathrm{tc}}.\end{aligned}$$

$$\Delta\mathrm{MTTF}^{i}_{\mathrm{tc}} = \mathrm{nd}(i) \cdot \left[\left(\frac{\mathrm{T_{avg\_a}} - \mathrm{T_{amb}}}{\mathrm{T_{avg\_b}} - \mathrm{T_{amb}}} \right)^{q} - 1 \right] \cdot \mathrm{MTTF}^{i}_{\mathrm{base}}.$$

where $\mathrm{hours}(i, \mathrm{tini}, \mathrm{tend})$ is the number of hours the i-th node has been powered on during the simulation, $\mathrm{MTTF}^{i}_{\mathrm{srv}}$ is the effective Mean Time To Failure of the node representing the estimated number of hours until failure, $\mathrm{MTTF}^{i}_{\mathrm{base}}$ the baseline Mean Time To Failure reported by the hardware manufacturer prior to the simulation, $\mathrm{on}(i, \mathrm{tini}, \mathrm{tend})$ is the amount of time the i-th node is powered on during the simulation, $\mathrm{hload}(i, t)$ is 1 if the i-th node is at high load at time t and 0 otherwise. AF_i is the acceleration factor for CPU degradation between the higher CPU temperature when the server is at high load ($\mathrm{T_{hload}}$) compared with the CPU temp when is at idle ($\mathrm{T_{idle}}$), as described in [15], [17] and based on the time-to-fail model of the Arrhenius equation. $\Delta\mathrm{MTTF}^{i}_{\mathrm{tc}}$ is the reduction in the Mean Time To Failure due to node power-on/off cycles (thermal cycling) and is calculated as decribed in [15], [17] where $\mathrm{T_{avg\_b}}$ and $\mathrm{T_{avg\_a}}$ are the average CPU temperatures before and after the power-on/off cycle, $\mathrm{T_{amb}}$ is the ambient temperature and q is the Coffin-Mason exponent taking the value 2.35 [17].

The effective disk failure rates for the i-th node are estimated as:

$$\mathrm{SFR}^{i}_{\mathrm{HDD}} = \frac{\mathrm{hours}(i, \mathrm{tini}, \mathrm{tend})}{\mathrm{MTTF}^{i}_{\mathrm{HDD}} - \Delta\mathrm{MTTF}^{i}_{\mathrm{HDD}}}.$$

$$\Delta\mathrm{MTTF}^{i}_{\mathrm{HDD}} = \frac{H}{\mathrm{AFR_{HDD}}(f^{i})} - \frac{H}{\mathrm{AFR_{HDD}}(f^{i} + \Delta f^{i})}.$$

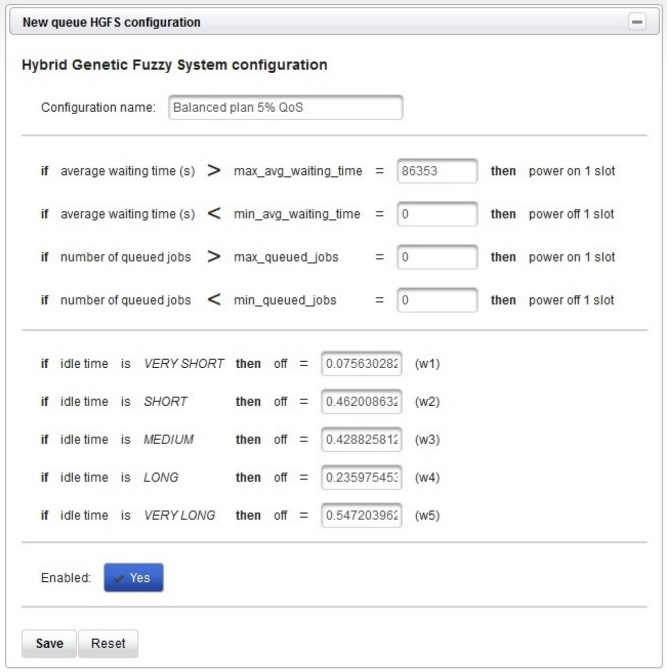
$$\Delta f^{i} = f^{i} + \frac{\mathrm{nd}(i)}{\mathrm{months}(i, \mathrm{tini}, \mathrm{tend})}.$$

$$\mathrm{AFR_{HDD}}(f) = 1.51\mathrm{e}{-5}f^{2} - 1.09\mathrm{e}{-4}f + 1.39\mathrm{e}{-4}.$$

where $\mathrm{MTTF}^{i}_{\mathrm{HDD}}$ is the effective Mean Time To Failure of the disk representing the estimated number of hours until failure, H is the number of hours in a year $(8, 760)$, $\Delta\mathrm{MTTF}^{i}_{\mathrm{HDD}}$ is the reduction in the Mean Time To Failure of the disk due to node power-on/off cycles, $\mathrm{AFR_{HDD}}(f)$ is a function that empirically quantifies the disk Annualized Failure Rate (AFR) based on its spindle start/stop frequency [18], f^{i} is the baseline disk spindle start/stop frequency (in times per month) prior to any node reconfiguration, Δf^{i} is the increment in the disk start/stop frequency due to node reconfigurations, $\mathrm{nd}(i)$ is the number of discontinuities of the function $\mathrm{state}(i, t)$ in the time interval $t \in (\mathrm{tini}, \mathrm{tend})$, and $\mathrm{months}(i, \mathrm{tini}, \mathrm{tend})$ is the number of months the i-th node has been powered on during the simulation.

IV. Using EECluster

EECluster is publicly accesible via web[4]. This website displays all information regarding the tool, including a brief description of the software, quick start guide, distribution tarballs, contact address, and acknowledgments. Once the pre-requirements are met (a compatible RMS, MySQL, Java EE application server and Java Runtime Engine), EECluster can be deployed in a cluster by downloading the tarball and extracting its content into a local directory of the master node. Then, the configuration script *configure.sh* is run to automatically check all pre-requirements, configure and install the *eeclusterd* service, and deploy the administration dashboard. Once installed and running, the *eeclusterd* will start collecting information from the cluster status (nodes, jobs, queues, etc.) via the RMS records and displaying it in the dashboard. More detailed information on the installation of EECluster and its basic use can be found in [10]. After EECluster is deployed and running, the Hybrid GFS must be configured to start reshaping the cluster. This is done using the tuning algorithm detailed in Section III, which is launched to find a set of non-dominated configurations (Pareto Efficient Frontier) by running the *HGFS_tuning.sh* script located in the EECluster installation directory and passing it as parameters the name of the target queue, the dates interval for the training and the validation set, the MOEA algorithm [14], the maximum number of fitness evaluations, and the number of seeds for the algorithm. The results are stored in a CSV file indicating for every learned HGFS configuration the fitness values for the training and validation datasets, allowing the cluster administrator to choose the preferred one according with his or her subjective preferences, and then configuring it in the dashboard (see Figure 3).



New queue HGFS configuration

Hybrid Genetic Fuzzy System configuration

Configuration name: Balanced plan 5% QoS

if average waiting time (s) > max_avg_waiting_time = 86353 then power on 1 slot

if average waiting time (s) < min_avg_waiting_time = 0 then power off 1 slot

if number of queued jobs > max_queued_jobs = 0 then power on 1 slot

if number of queued jobs < min_queued_jobs = 0 then power off 1 slot

if idle time is *VERY SHORT* then off = 0.075630282 (w1)

if idle time is *SHORT* then off = 0.462008632 (w2)

if idle time is *MEDIUM* then off = 0.428825812 (w3)

if idle time is *LONG* then off = 0.235975453 (w4)

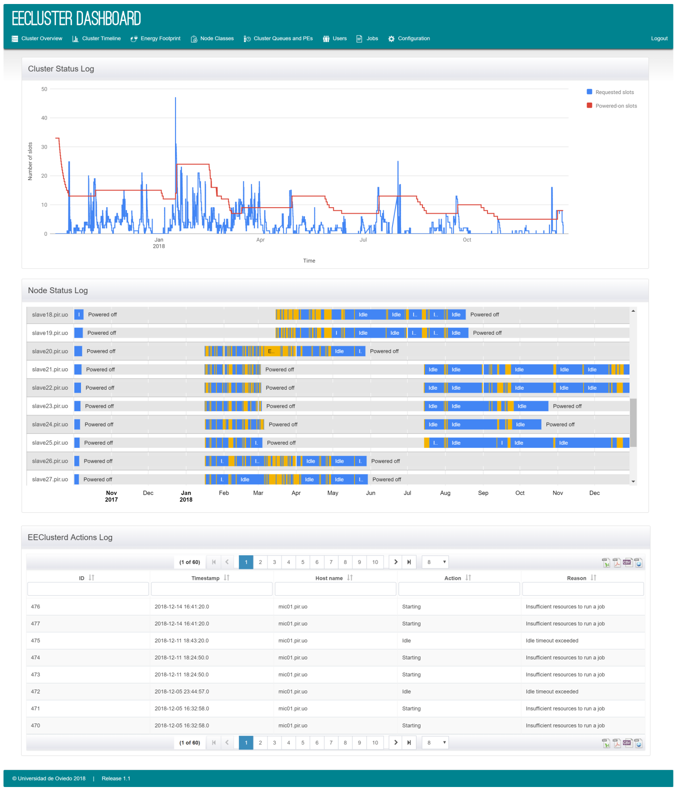
if idle time is *VERY LONG* then off = 0.547203962 (w5)

Enabled: ✓ Yes

Fig. 3. HGFS configuration in the EECluster dashboard.

In this newer version, EECluster also includes a more accurate and detailed reporting on its reconfiguration decisions made by the HGFS and its effect on the cluster status over time, as can be seen in Figure 4. Moreover, in regard to the eco-efficiency and the operating costs of the cluster,

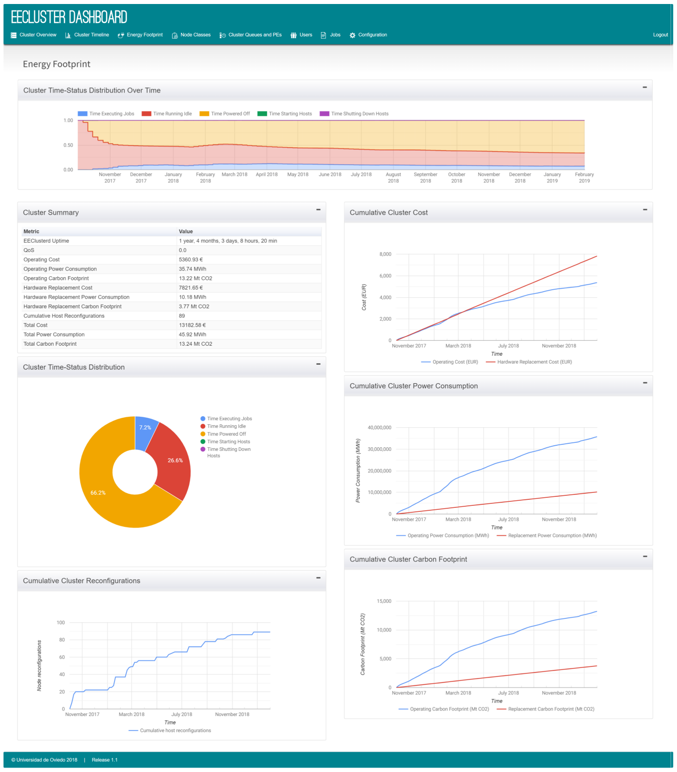
[4] http://sourceforge.net/projects/eecluster/



EEClusterd Actions Log

ID	Timestamp	Host name	Action	Reason
476	2018-12-14 16:41:20.0	mic01.pir.uo	Starting	Insufficient resources to run a job
477	2018-12-14 16:41:20.0	mic01.pir.uo	Starting	Insufficient resources to run a job
475	2018-12-11 18:43:20.0	mic01.pir.uo	Idle	Idle timeout exceeded
474	2018-12-11 18:24:50.0	mic01.pir.uo	Starting	Insufficient resources to run a job
473	2018-12-11 18:24:50.0	mic01.pir.uo	Starting	Insufficient resources to run a job
472	2018-12-05 23:44:57.0	mic01.pir.uo	Idle	Idle timeout exceeded
471	2018-12-05 16:32:58.0	mic01.pir.uo	Starting	Insufficient resources to run a job
470	2018-12-05 16:32:58.0	mic01.pir.uo	Starting	Insufficient resources to run a job

Fig. 4. Cluster status over time.


EECLUSTER DASHBOARD
Cluster Overview
Cluster Timeline
Energy Footprint
Node Classes
Cluster Queues and PEs
Users
Jobs
Configuration
Logout
Energy Footprint
Cluster Time-Status Distribution Over Time
Time Executing Jobs
Time Running Idle
Time Powered Off
Time Starting Hosts
Time Shutting Down Hosts
Cluster Time-Status Distribution
7.2%
26.6%
66.2%
Cumulative Cluster Reconfigurations
Node reconfigurations
Cumulative host reconfigurations
Cumulative Cluster Cost
Cost (EUR)
Operating Cost (EUR)
Hardware Replacement Cost (EUR)
Cumulative Cluster Power Consumption
Power Consumption (MWh)
Operating Power Consumption (MWh)
Replacement Power Consumption (MWh)
Cumulative Cluster Carbon Footprint
Carbon Footprint (Mt CO2)
Operating Carbon Footprint (Mt CO2)
Replacement Carbon Footprint (Mt CO2)
Time
© Universidad de Oviedo 2018 | Release 1.1

Cluster Summary

Metric	Value
EEClusterd Uptime	1 year, 4 months, 3 days, 8 hours, 20 min
QoS	0.0
Operating Cost	5360.93 €
Operating Power Consumption	35.74 MWh
Operating Carbon Footprint	13.22 Mt CO2
Hardware Replacement Cost	7821.65 €
Hardware Replacement Power Consumption	10.18 MWh
Hardware Replacement Carbon Footprint	3.77 Mt CO2
Cumulative Host Reconfigurations	89
Total Cost	13182.58 €
Total Power Consumption	45.92 MWh
Total Carbon Footprint	13.24 Mt CO2

Fig. 5. Direct and indirect costs and carbon footprint of the cluster.

another new section was included to represent the cumulative monetary costs, energy consumption and carbon footprints since EECluster was deployed, as well as their evolution over time (Figure 5). To compute these figures, the same formulae described for the fitness in Section III were used. The hardware parameters used as inputs in these formulae (ulife, $\text{MTTF}_{\text{base}}$, etc.) are configured in the "Node Classes" section of the EECluster dashboard.

V. Empirical Analysis

The EECluster tool has been tested in various cluster environments with different applications. This includes the Scientific Modelling Cluster of the University of Oviedo (CMS)[5] composed of three clusters and five transversal queues using PBS as RMS, a 5-node research cluster with heterogeneous hardware (3 Dell PowerEgde servers for OpenMP jobs, one ASUS server with 2 NVIDIA Tesla for CUDA jobs, and one Supermicro server with 3 Intel Xeon Phi coprocessors) using OGE/SGE as RMS and supporting research in the field of algorithm and chemical computational modelling, and a 34-node OGE/SGE academic cluster used for educational purposes, allowing students to learn and experiment with multicore, distributed and GPU computing. Is it also noteworthy that another GFS-based mechanism following EECluster's approach has been successfully used to optimize virtual machine allocation and server consolidation in Virtual Data Centers [19].

In order to measure the savings achieved with EECluster in terms of monetary cost and carbon footprint in a real world environment, an experiment was done simulating a 33-node cluster with each node characterized by the hardware parameters detailed in Table I, and running actual workloads of the aforementioned Scientific Modelling Cluster of the University of Oviedo spanning 22 months with a total of 2907 jobs. To assess the results of EECluster under different set of tradeoffs between QoS, direct and indirect costs, three configurations were machine-learned and validated using the holdout method with a 70-30% split in training and test. The first (labelled as "HGFS QoS 0.0") priorities QoS above all other criteria, the second (labelled as "HGFS QoS 0.1") seeks the lowest direct costs long the QoS value is below or equal to 0.1, and the last one (labelled as "HGFS QoS 0.5") rises the QoS threshold to 0.5. In all cases, indirect costs are used to break ties between direct costs. To serve as reference, a fourth configuration labelled "None" represents the fitness values if no controller is used.

As shown in Table II, EECluster is capable of optimizing overall cluster operating expenses and carbon footprints by suiting the preferred tradeoff between service quality, direct and indirect costs, and enabling an informed decision-making for the cluster administrator by translating the results achieved to representative metrics.

VI. Concluding Remarks

It is well known that a substantial reduction of the carbon footprint of High Performance Computing Clusters (HPCCs)

[5] Further information on the CMS can be found in its web site (http://cms.uniovi.es)

TABLE I
HARDWARE PARAMETERS OF THE COMPUTE NODES USED IN THE SIMULATION.

Parameter	Value	Unit
$\rho_{\text{€}}$	0.15	/kWh
ρ_{CO_2}	0.37	Kg CO_2/kWh
Power at idle	150	Watts
Power at high load	350	Watts
ulife	60	%
$\text{msrv}_{\text{€}}$	4,200	
msrv_{CO_2}	9,536	Kg CO_2
$\text{mHDD}_{\text{€}}$	150	
mHDD_{CO_2}	238.41	Kg CO_2
$\text{MTTF}_{\text{base}}$	219,000	hours
T_{amb}	22	°C
$T_{\text{avg\_a}}$	24	°C
$T_{\text{avg\_b}}$	50	°C
T_{hload}	65	°C
T_{idle}	50	°C
num_{HDD}	4	HDDs
MTTF_{HDD}	146,000	hours
f	350	times/month

TABLE II
EXPERIMENT RESULTS FOR THE TEST SET.

	Test set			
	QoS	Direct cost (EUR)	Indirect cost (EUR)	Carbon footpr. (Mt CO_2)
None	0.00	3,950.21	1,517.49	13.03
HGFS QoS 0.0	0.00	2,437.28	893.25	7.96
HGFS QoS 0.1	0.07	1,747.44	664.10	5.76
HGFS QoS 0.5	0.19	1,516.09	650.54	5.17

can be achieved just by software. As such, there is a number of packages that improve the energy efficiency of HPCCs. Furthermore, energy savings are multiplied by the cascade effect: each watt that is saved by an efficient management of the compute nodes propagates reductions in the power supply, UPS, cooling and transmission lines.

EECluster is a tool that transforms OGE/SGE and PBS/TORQUE-based HPCCs into energy-efficient adaptive resource clusters by combining human experience with rules learnt from data in a system that is tailored to each specific cluster. In this paper the package EECluster is improved to optimize direct and indirect operating costs and environmental impacts, taking into account the power use of the underlying hardware and the effect of the cluster reconfigurations on the hardware reliability and its subsequent replacements whenever failures occur. As a result of this, the tool is now better suited to improve the eco-effiency of the HPCCs by explicitly balancing service quality with direct and indirect costs from both an economic and an environmental standpoint. The tool has been tested at the Scientific Modelling Cluster of Oviedo University and deployed at different clusters for research tasks in algorithm parallelization and chemical computational modelling.

ACKNOWLEDGEMENTS

This work has been partially supported by the Ministry of Economy, Industry and Competitiveness (“Ministerio de Economía, Industria y Competitividad”) from Spain/FEDER under grants MTM2017-83506-C2-2-P, TIN2017-84804-R and by the Regional Ministry of the Principality of Asturias (“Consejería de Empleo, Industria y Turismo del Principado de Asturias”) under grant GRUPIN18-226.

REFERENCES

[1] M. Ebbers, Mike Archibald, C. F. F. da Fonseca, M. Griffel, V. Para, and M. Searcy, “Smarter Data Centers: Achieving Greater Efficiency,” IBM Redpaper, Tech. Rep., 2011.

[2] The Economist Intelligence Unit, “IT and the environment A new item on the CIOs agenda?” The Economist, Tech. Rep., 2007.

[3] Gartner, “Gartner Estimates ICT Industry Accounts for 2 Percent of Global CO2 Emissions,” STAMFORD, 2007.

[4] A. Cocaña-Fernández, J. Ranilla, and L. Sánchez, “Energy-efficient allocation of computing node slots in HPC clusters through parameter learning and hybrid genetic fuzzy system modeling,” *The Journal of Supercomputing*, vol. 71, no. 3, pp. 1163–1174, oct 2014.

[5] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, “Load balancing and unbalancing for power and performance in cluster-based systems,” in *Workshop on compilers and operating systems for low power*, vol. 180. Barcelona, Spain, 2001, pp. 182–195.

[6] R. Das, J. O. Kephart, C. Lefurgy, G. Tesauro, D. W. Levine, and H. Chan, “Autonomic multi-agent management of power and performance in data centers,” pp. 107–114, May 2008.

[7] F. Alvarruiz, C. de Alfonso, M. Caballer, and V. Hernández, “An Energy Manager for High Performance Computer Clusters,” in *2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications*. IEEE, Jul. 2012, pp. 231–238.

[8] M. F. Dolz, J. C. Fernández, S. Iserte, R. Mayo, E. S. Quintana-Ortí, M. E. Cotallo, and G. Díaz, “EnergySaving Cluster experience in CETA-CIEMAT,” in *5th Iberian GRID Infrastructure conference*, Santander, 2011.

[9] S. Kiertscher, J. Zinke, S. Gasterstadt, and B. Schnor, “Cherub: Power consumption aware cluster resource management,” in *2010 IEEE/ACM Int’l Conference on Green Computing and Communications Int’l Conference on Cyber, Physical and Social Computing*, Dec 2010, pp. 325–331.

[10] A. Cocaña-Fernández, L. Sánchez, and J. Ranilla, “A software tool to efficiently manage the energy consumption of HPC clusters,” in *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, aug 2015, pp. 1–8.

[11] ——, “Improving the eco-efficiency of high performance computing clusters using eecluster,” *Energies*, vol. 9, no. 3, 2016. [Online]. Available: http://www.mdpi.com/1996-1073/9/3/197

[12] H. Ishibuchi, T. Nakashima, and M. Nii, “Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining (Advanced Information Processing),” Nov. 2004.

[13] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-15, no. 1, pp. 116–132, Jan. 1985.

[14] “MOEA Framework, a Java library for multiobjective evolutionary algorithms.”

[15] W. Deng, F. Liu, H. Jin, X. Liao, H. Liu, and L. Chen, “Lifetime or energy: Consolidating servers with reliability control in virtualized cloud datacenters,” in *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, Dec 2012, pp. 18–25.

[16] B. Guenter, N. Jain, and C. Williams, “Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning,” in *2011 Proceedings IEEE INFOCOM*, April 2011, pp. 1332–1340.

[17] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, “Lifetime reliability: toward an architectural solution,” *IEEE Micro*, vol. 25, no. 3, pp. 70–80, May 2005.

[18] T. Xie and Y. Sun, “Sacrificing reliability for energy saving: Is it worthwhile for disk arrays?” in *2008 IEEE International Symposium on Parallel and Distributed Processing*, April 2008, pp. 1–12.

[19] A. Cocaña-Fernández, J. Rodríguez-Soares, L. Sánchez, and J. Ranilla, "Improving the energy efficiency of virtual data centers in an it service provider through proactive fuzzy rules-based multicriteria decision making," *The Journal of Supercomputing*, Mar 2018. [Online]. Available: https://doi.org/10.1007/s11227-018-2301-1