# Energy and Carbon Aware Scheduling in Supercomputing

Mikko Majanen, Olli Mämmelä
*Autonomic Networking Team*
*VTT Technical Research Centre of Finland*
*Kaitoväylä 1, Oulu, Finland*
*Email: mikko.majanen@vtt.fi, olli.mammela@vtt.fi*

André Giesler
*Jülich Supercomputing Centre*
*Forschungszentrum Jülich*
*52425 Jülich, Germany*
*Email: a.giesler@fz-juelich.de*

***Abstract*—At an early stage of information and communications technology and high-performance computing, performance and reliability were two important factors in research and development. Energy consumption was not considered as a serious topic, since the technical characteristics of hardware and software were limited and the amount of computing nodes in a computing cluster, i.e., a data centre was small. Gradually the situation has evolved a lot: nowadays there are multiple data centres located in geographically diverse locations and the software has become more complex. Modern data centres are equipped with a large amount of computing nodes having vast computing power. Consequently, energy consumption has become a major topic. This work presents two algorithms for optimizing energy and emissions in high-performance grid computing, in which multiple data centres are interconnected to each other. The algorithms are validated both in simulation and testbed environments. The effect of various parameters to energy and emission savings are studied and the performance of the algorithms is compared to commonly used default algorithms. Our simulation and testbed experiments show that the developed algorithms are able to reduce energy consumption and emissions drastically without significant increase in job turnaround or wait time.**

***Keywords-HPC; grid computing; energy; emissions; testbed.***

## I. Introduction

The increased demand for IT applications and services has encouraged the building of data centres worldwide. However, data centres consume an enormous amount of energy at an increasing financial and environmental cost. This has led to research efforts in both industry and academia to cut down data centre energy usage and emissions. This work is a continuation to our prior work in ENERGY 2012 [1] to save energy and reduce the $CO_2$ emissions in federated High-Performance Computing (HPC) data centres. Previously, we have also researched the energy saving potential inside single site data centres [2].

In 2006, U.S. servers and data centres consumed around 61 billion kilowatt hours (kWh) at a cost of about 4.5 billion U.S. Dollars [3]. This is equal to about 1.5% of the total U.S. electricity consumption or the output of about 15 typical power plants. High energy consumption naturally causes huge environment pollution. It has been estimated that Information and Communications Technology (ICT), as a whole, covers 2% of world's $CO_2$ emissions [4] and this amount looks set to grow at 6% each year until 2020 [5]. Data centres were 14% of the total ICT footprint in 2002 and 2007, and it is estimated that the amount will rise to 18% in 2020.

In HPC, the ever-growing demand for higher performance seems to increase the total power consumption, even though more flops per watt are achieved. In order to provide even greater computing capabilities, HPC data centres can be interconnected to each other to form larger, federated or HPC grid data centres. The connection is implemented by using special grid software (e.g., UNICORE (UNiform Interface to COmputing REsources) [6]) that manages the job submissions to all data centres belonging to the grid.

The energy consumption between the data centres may vary radically due to the different characteristics of the centres. For example, the server hardware in each centre may be different and consume different amount(s) of energy. The centres may also locate geographically far from each other and the surrounding climate can cause large differences in the needed cooling, i.e., the Power Usage Effectiveness (PUE) [7] values between different centres may vary due to the surrounding climate. Also, since the energy sources can differ between the centres, the $CO_2$ emissions of the data centres may vary radically depending on the available energy sources. The differences between the data centres naturally enable optimizations regarding energy consumption and $CO_2$ emissions.

In our prior work [1] we introduced two algorithms for selecting the data centre inside the grid in energy- and $CO_2$-aware manner. The performance of the algorithms was studied by simulations and the results showed significant savings in energy consumption and $CO_2$ emissions. This work extends our previous work by a feasibility study [8] of the algorithms in a testbed environment that consists of three clusters: two located in Germany and one in Finland. The testbed results confirm the

possibilities for energy and emission savings achieved in the simulations, and can be used as a basis for the design of specific federated cluster environments when using a developed software plug-in to enable an energy-aware scheduling of the resources. Furthermore, we also consider energy and $CO_2$ emission savings inside single HPC data centres by using energy-aware scheduling algorithms presented in [2].

The rest of the paper is organized as follows: Section II describes the related work. Section III introduces the cluster selection algorithms. Section IV describes the simulation model and scenario, and presents the simulation results. Testbed experiments are presented in Section V, including the scenario and results. Conclusion and future work are presented in Section VI.

## II. Related work

As described in [2], several methods for saving energy in single HPC data centres have been studied. The methods include mainly the use of energy-efficient or energy proportional hardware (e.g., embedded low-power chips), Dynamic Voltage and Frequency Scaling (DVFS) techniques, shutting down idle hardware components at low system utilizations, power capping, and thermal management. Recently, there has also been approaches to solve HPC energy issues in Graphics Processing Unit (GPU) computing [9], [10], [11]. GPU computing aims at combining the use of a GPU with a CPU to accelerate general-purpose scientific and engineering applications. The compute-intensive portions of the application are offloaded to the GPU, while the remainder of the code still runs on the CPU. However, the energy consumption is a major concern in these systems.

In our prior work [2], we used an energy-aware job scheduler to schedule the jobs inside single data centres and shut down idle computing nodes whenever possible. We also noted that merely the choice of a different scheduling algorithm can affect the energy consumption of a data centre. Out of commercial HPC schedulers, Moab offers a Green Computing plug-in [12] that tries to reduce power consumption and costs in a data centre in a quite similar way as our energy-aware job scheduler. In the Moab plug-in, it is possible to turn off idle nodes that do not have reservations on them, and turn on additional nodes when jobs require them. Moab uses a MAXGREENSTANDBYPOOLSIZE parameter, where users can specify a "green pool", which is the number of nodes that are kept on and ready to run jobs (even if some nodes are idle). Idle nodes that exceed the number specified with the MAXGREENSTANDBYPOOLSIZE parameter are turned off. The requirements for the Green Computing plug-in are a license for green computing, Moab 5.3.5 or later, a script that Moab can call to programatically turn nodes on and off, and a resource manager that can monitor and report power state. In a test run, the Green Computing plug-in was able to decrease the energy consumption by 8.2% with the penalty of 7.5% increased workload completion time [13]. The savings with Moab Green Computing depend highly on the workload.

In this paper we extend our scope from single HPC data centres to HPC grid data centres and introduce two algorithms for selecting the data centre inside the grid in energy- and $CO_2$-aware manner. Moreover, we provide HPC grid simulation and testbed results, and new single HPC data centre results.

Until recently, there has not been much previous research that addresses the energy efficiency or $CO_2$ emissions of the grids from the whole grid perspective; mainly only optimizations inside a single data centre have been studied. Perhaps the most similar approach to our approach is the Heterogeneity Aware Meta-scheduling Algorithm (HAMA) [14]. HAMA first selects the most energy-efficient cluster for the job based on the power consumption of the servers and the efficiency of the cooling system. Additionally, when running the job, DVFS is used to reduce the power consumption of the CPU. The simulation results show that HAMA can reduce up to 23% energy consumption in the worst case and up to 50% in the best case as compared to other algorithms (EDF-FQ, which prioritizes jobs based on a deadline and submits jobs to resource sites in earliest start time (FQ) manner with the smallest waiting time). Without DVFS, HAMA can still result in power savings of up to 21%.

Lynar *et al.* [15] have explored the effect on energy consumption by using different resource allocation mechanisms, both in a cluster and in a grid. The results show that different resource allocation methods can result in a significantly different energy usage while computing a stream of tasks. The Pre-processed Batch Auction (PPBA) and batch auctions almost always result in a significantly lower energy use than a random resource allocation. By using a simple batch auction allocation method, energy consumption can be reduced by up to 37.5%, and possibly even more by using the PPBA method.

Patel *et al.* [16] have presented an energy-aware policy for distributing computational workload in the Grid resource management architecture. They introduce a data centre energy coefficient that is taken into account as a policy when making allocation decisions for compute workloads. This coefficient is determined by the thermal properties of each data centre's cooling infrastructure including regional and seasonal variations. The estimated energy savings in case of three data centres located in two different time zones were large enough to give sufficient reason for the economic viability of the approach.

Shah and Krishnan [17] also analyze the climatic conditions as a means to reducing cooling energy costs. They show that dynamic optimization of the thermal workloads based on local weather patterns can reduce the environmental burden by up to 30% in their case study. Additionally, the data centre operational costs can be potentially reduced by nearly 35%. Due to the variability of fuel mixes encountered in a global grid, they also found that the use of pure energy consumption as a metric for environmental sustainability — a common practice in the ICT literature — can be erroneous.

The GREEN-NET framework [18] consists of an ON/OFF model, which includes prediction heuristics and green advice for the users and takes the decision to switch on or off the nodes, and an adapted energy efficient Resource Management System (RMS) at the grid level.

There has also been research on the feasibility of powering data centres more by renewable energy ( [19], [20], [21]) and studying the environmental potential of Geographical Load Balancing (GLB) [22], in which processes are shifted to data centres located in regions where energy currently has low cost. In [23], a renewable and cooling aware workload management plan is introduced. The availability of renewable energy and IT demand is predicted and IT resources are allocated according to a time varying power supply and cooling efficiency. A similar approach is taken in GreenSlot [24], which is a parallel batch job scheduler powered by solar power and the electrical grid (as a backup). It predicts the amount of solar energy that will be available in the near future and schedules the workload to maximize the use of green energy without breaking any Service Level Agreement (SLA).

## III. Optimization in the HPC Grid

The optimization algorithm in the HPC grid focuses on optimizing the scheduling process in the UNICORE middleware [6]. The scheduling process is triggered by submitting a job from the UNICORE Commandline Client, or from the UNICORE Rich Client to the UNICORE Workflow Engine. The UNICORE Workflow Engine queries a UNICORE Service Orchestrator (USO), on which cluster the job should be submitted. As a default, the USO uses round-robin algorithm for choosing the cluster. After cluster decision, the job is submitted to the RMS of the chosen cluster. The RMS takes care of executing the job according to the used scheduling algorithm, e.g., FIFO (First In, First Out) or backfilling.

In this work, we focus on reducing the energy consumption and the $CO_2$ emissions. The $CO_2$/energy related optimizations should not affect the current SLA or Quality of Service (QoS) agreements, or alternatively, a new green SLA [25] could be used. In HPC, there are no clear SLAs between users and data centres, but a reasonable turnaround time can be seen as sort of a QoS agreement. A possible green SLA for HPC data centres could mean that the users allow certain delay for the execution of their job. As a bonus, they will get some extra computing time for free.

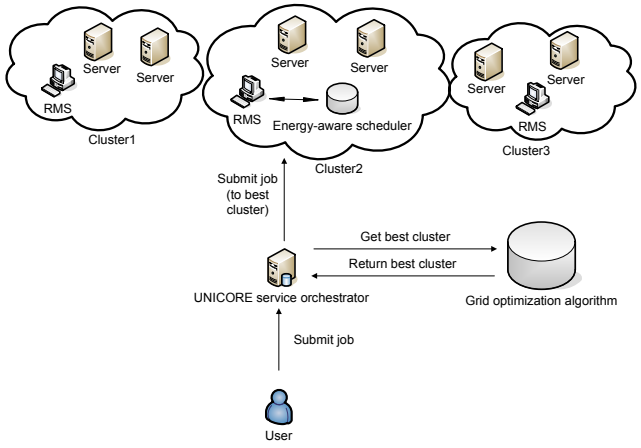


Figure 1. Job submission in a federated HPC data centre

For decreasing $CO_2$ emissions and/or energy consumption in federated HPC data centres, the optimization algorithm will be used for performing the cluster selection in $CO_2$/energy-aware manner. In addition to cluster selection algorithms, also energy-aware single site scheduling algorithms will be used. As depicted in Figure 1, the USO in UNICORE receives job requests coming from the users. The jobs include the requirements for the needed resources (e.g., number of nodes/cores, memory, etc.). If the user wants to use the green SLA, it is also included in the job requirements. The grid optimization algorithm is used to select the most suitable cluster for the job and the job is subsequently submitted to the RMS of the selected cluster. The RMS uses energy-aware job scheduling algorithms to schedule the job and power off idle servers. The energy-aware job scheduling algorithms for single site data centres were defined in our previous work [2]. The main principle of the single site algorithms is to keep the system active with the lowest amount of resources as possible. If a node is not needed for job execution, it is shut down or placed into an energy saving mode. Once the node is needed for the job execution again, it is woken up.

### A. Carbon Usage Effectiveness

Carbon Usage Effectiveness (CUE) is a sustainability metric developed by the Green Grid organization [26]. The main purpose of the metric is to address carbon emissions associated with data centres. The CUE can be calculated as follows:

Table I
ENERGY EMISSION COEFFICIENT FACTORS

| Generation type | Conversion factor ($kgCO_2$ per kWh) |
|---|---|
| Closed cycle gas turbine | 0.360 |
| Coal | 0.910 |
| Electricity, France interconnector | 0.083 |
| Electricity, Ireland interconnector | 0.699 |
| Non pumped storage hydro | 0.0 |
| Nuclear | 0.0161 |
| Open cycle gas turbine | 0.479 |
| Oil | 0.610 |
| Pump storage | 0.0 |
| Other | 0.610 |

$$CUE = \frac{SiteEmissions}{ICTEnergy}, \quad (1)$$

where $ICTEnergy$ is the energy consumpted by the ICT equipment in the data centre. An alternative approach for calculating the CUE is to multiply the Energy Source Coefficient (ESC) by the data centre's PUE:

$$CUE = ESC * PUE, \quad (2)$$

where PUE is a metric for defining how efficiently the power in the data centre is used, i.e., how much power is actually used by the ICT equipment and how much power is used for cooling and other equipment. ESC is defined as follows:

$$ESC = \sum ESP * EEC, \quad (3)$$

where Energy Source Percent (ESP) indicates the percentage of the energy generation source, and Energy Emission Coefficient (EEC) indicates how many kilograms of $CO_2$ are emitted per 1 kWh of energy. Example values of the EEC can be found in Table I [27]. By using the formulas described earlier and the values in Table I, we are able to estimate how much emissions are caused by data centres with different energy sources:

$$SiteEmissions = CUE * ICTEnergy \quad (4)$$
$$= PUE * ESC * ICTEnergy. \quad (5)$$

### B. Algorithm description

This subsection describes the functionalities of the default round-robin cluster selection algorithm, as well as the two developed algorithms for optimizations: Fastest possible (FP) that tries to minimize the wait time, and Energy and $CO_2$-aware (ECA) that tries to minimize the energy or $CO_2$ emissions.

*1) Round-robin (RR):* RR algorithm is generally used in USO for selecting the cluster. It balances the number of jobs between different clusters by always choosing the next cluster compared to the previous selection. After the last cluster, the selection is started again from the first cluster.

*2) Fastest possible (FP):* FP cluster selection algorithm tries to select the cluster that could possibly execute the job with minimal wait time. For this, the algorithm first checks if there are enough idle nodes/cores in some cluster for executing the job. If yes and the cluster's queue is also empty, the job is submitted to that cluster. If not, an estimated wait time for the job in each cluster is calculated by using the current status of each cluster: number of nodes and cores, status of running jobs, number of jobs in the queue, and walltimes of each queued job. The cluster with the shortest estimated wait time is then selected.

The algorithm relies on the dynamic cluster properties (status of nodes and queues), which can be obtained by a single site monitoring system. Otherwise, this dynamic information is not available for the USO, so the normal cluster selection algorithms can exploit only static cluster information for the decision making.

It should be noted that the wait time can only be estimated. The walltimes of the jobs are given by the users and, in general, they are inaccurate [28], [29]. Also, the used scheduling algorithm affects in which order the jobs are executed (especially backfilling). Thus, it is possible to calculate only the maximum wait times for the jobs, not the exact wait times.

*3) Energy and $CO_2$-aware (ECA):* This algorithm tries to find the cluster with the smallest amount of estimated energy consumption or $CO_2$ emissions; the optimization goal can be chosen by the user. The $CO_2$ emissions of the job can be calculated in the same way as for the whole site in Equation (4):

$$JobEmissions = CUE * ICTEnergyOfTheJob. \quad (6)$$

The simplest way is to select the cluster with the smallest CUE value. This works if the clusters have significant differences in their CUE values (CUE = ESC * PUE). If there are only small differences in the CUE values, then additional estimations should be done, since the job may consume different amount of ICT energy in different clusters due to the different computing node properties (CPU, RAM, etc.), and this difference may become a greater factor than CUE for the $CO_2$ emissions. The ICT energy of the job can be estimated by using the job requirements (number of nodes/cores, walltime) and cluster's computing node properties (CPU, RAM, etc.) as inputs for power consumption models such as those described in [2] and [30]. Moreover, the job execution time may differ greatly between clusters because of

varying hardware parameters. Thus, it is very important to try to estimate the execution time of the job on different clusters. Special application benchmarks for estimating the clusters' job execution times were chosen for this purpose and they were used in the testbed experiments. These benchmark applications are run once on each cluster beforehand, and the execution times are measured. Based on the measurements, the data centre operators rank each cluster and benchmark application combination. The higher the rank, the faster the execution time is.

Thus, the ICT energy of the job is given by

$$ICTEnergy = ICTPower * ExecutionTime, \quad (7)$$

where $ICTPower$ is the power consumption of the job and is calculated by using the power consumption models, and $ExecutionTime$ is calculated by using the walltime estimate and the rank of the cluster.

However, selecting always the cluster with the least amount of estimated $CO_2$ emissions would cause huge load and queue on the cluster with the least $CO_2$ emissions. This would mean large delay for the users. Thus, some form of load balancing is needed for this algorithm. In the conducted simulations (described in the next sections), we used a queue size limit: If the queue exceeded its size limit, the job was submitted to the cluster with the second least $CO_2$ emissions, and so on. In the case of green SLA, the users set a certain deadline for the completion of their job. This limit can be used for load balancing: The estimated completion time for the job can be calculated as a sum of the estimated wait time and walltime of the job. If this is in the limits, the cluster can be chosen. If not, the same calculations should be made to the cluster with the second least $CO_2$ emissions, and so on. If the user sets too strict time limit for the job that none of the clusters can fulfill, the job should be either denied or the cluster should be chosen by the Fastest possible algorithm. In the testbed experiments (described later) we used this green SLA method for load balancing, and FP algorithm in case of too strict time limits. The ECA algorithm can be used for selecting the cluster with minimal energy consumption, too, by replacing CUE by PUE.

The ECA algorithm takes into account the dynamic properties of the cluster and compute nodes. This information is stored in a meta-model, which is updated accordingly if any of the parameters, such as PUE, CUE or compute node hardware parameters are changed.

## IV. Simulation studies

The simulation model for the HPC grid has been developed with the OMNeT++ discrete event network simulator [31] and the INET Framework [32]. The design of the model is similar as in [2], except that the model is extended from a single site scenario to a federated site scenario.

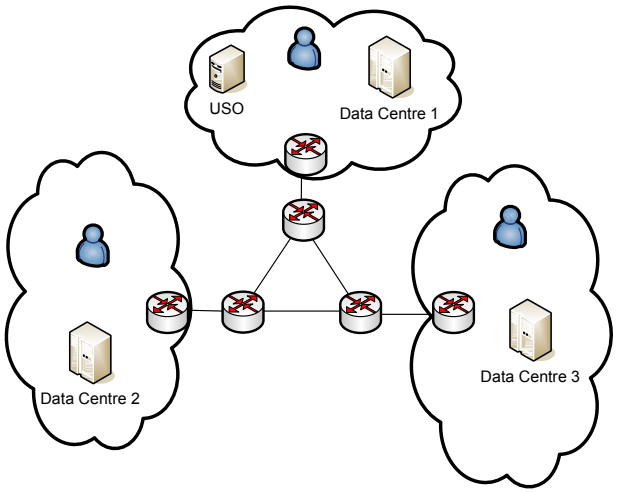


Figure 2. Network topology

Figure 2 illustrates the network topology used in the simulations. It consists of three backbone routers, three gateway routers, three data centre modules, three clients and a USO module. In this scenario, the clients send HPC job requests to the USO, which is responsible for choosing an appropriate data centre, i.e., an HPC cluster, for executing the job. The USO has been adapted for the simulation so that it is capable of using the developed optimization algorithms and making decisions based on the dynamic properties of the clusters. Normally, only static information of the clusters is available for the USO.

For the decision making, the USO can query the status and properties of each cluster from the corresponding RMS. Once the cluster is chosen, the USO forwards the job request to the RMS of the chosen cluster. The RMS uses the policies and scheduling algorithms of the cluster to choose suitable servers for job execution. When the job execution finishes, the RMS informs the USO, which again forwards the information to the client that submitted the job for execution.

The data centre module can be seen in Figure 3, which is similar as in the single site scenario used in our previous work [2]. It contains a RMS, a fixed number of servers and a router between them. The RMS handles all incoming job requests arriving to the data centre and allocates the jobs to the servers for execution according to the selected policies and algorithms. Thus, the RMS also functions as a scheduler in the simulation. The RMS supports 6 different scheduling algorithms: standard FIFO, Backfill First Fit and Backfill Best Fit algorithms and their energy-aware counterparts developed previously (see [2] for more information on these).

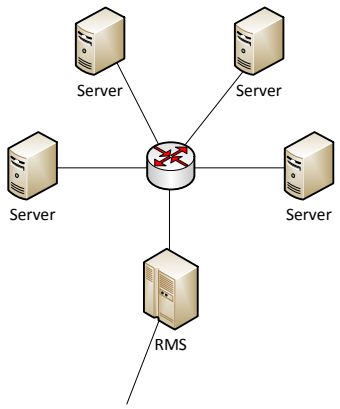


Figure 3. Data centre module

The RMS module includes parameters for the PUE and the CUE. By using these two values, the USO is able to select a cluster that is the most energy-efficient or produces the least amount of $CO_2$ emissions.

### A. Simulation scenario

In this subsection, we describe the simulation scenario and parameters. For evaluation we consider a scenario that includes three data centres and 75 clients that are sending job requests to the USO. The simulation is stopped once 1500 jobs have been completed. During the simulation we measure the energy consumed by each data centre and present the obtained results in the next section. General simulation parameters are presented in Table II. Uniform(a,b) means randomly selected value according to a uniform distribution between a and b. The scheduling and cluster selection algorithms are shortened as follows:

- FP = Fastest possible USO cluster selection algorithm
- ECA = (Energy and) $CO_2$-aware USO cluster selection algorithm set to minimize the $CO_2$ emissions
- RR = Round-robin USO cluster selection algorithm
- FIFO = First In, First Out job scheduling algorithm
- BFF = Backfilling first fit job scheduling algorithm
- BBF = Backfilling best fit job scheduling algorithm
- E-FIFO, E-BFF, E-BBF = energy-aware counterparts for the job scheduling algorithms (idle nodes are powered off whenever possible)

In Table III, we can see the parameters for the three clusters in the considered federated HPC data centre. The clusters have different characteristics, such as, the number of servers, PUE, and ESC. The energy sources (O = Oil, C = Coal, H = Hydro, N = Nuclear) for the clusters were selected so that both extreme ends in terms of ESC were represented in the simulations, while the third one represents something in the middle of them. Also, servers have different operating systems (OS) and processor architectures. In the simulations, the ECA algorithm optimization goal was set to minimize the $CO_2$ emissions.

Table II
Simulation parameters

| Parameter | Value |
|---|---|
| Simulation runs | 10 |
| Number of jobs | 1500 |
| Number of data centres | 3 |
| Number of clients | 75 |
| Number of gateway routers | 3 |
| Number of backbone routers | 3 |
| USO cluster selection algorithm | RR, FP, ECA |
| RMS scheduling algorithm | FIFO, BFF, BBF, E-FIFO, E-BFF, E-BBF |
| Server memory | 4 * 2 GB = 8 GB |
| Server cores per CPU | 2 |
| Server CPUs | 2 |
| Server CPU idle power | 15 W |
| Server core voltage | 1.2 V |
| Client job cores | 1, 2, 4 |
| Client job load | Uniform(30,99) |
| Client job nodes | Uniform(1,20) |
| Client job memory | Uniform(100MB, 2GB) |
| Client job run time | Uniform(600s, 86400s) |

Table III
Data centre parameters

| Parameter | Cluster 1 | Cluster 2 | Cluster 3 |
|---|---|---|---|
| Servers | 30 | 40 | 50 |
| Energy source | C 50% H 20% N 30% | C 80% O 20% | O 20% H 40% N 40% |
| PUE | 1.5 | 1.8 | 1.3 |
| ESC | 0.45983 | 0.85 | 0.12844 |
| CUE | 0.689745 | 1.53 | 0.166792 |
| OS | Linux | Windows | Linux |
| CPU arch. | AMD | Intel | Intel |

### B. Simulation Results

Figure 4 presents the total ICT energy consumption of the three clusters for different USO cluster selection and job scheduling algorithms. As can be seen, RR with normal job scheduling algorithms consumes the most amount of energy. RR with normal job scheduling algorithms represents a generally used, un-optimized algorithm combination in federated HPC data centres. Thus, it serves as a comparison point when calculating the energy savings and $CO_2$ emission reductions.

Figure 5 presents the energy savings achieved by using Fastest possible and $CO_2$-aware USO cluster selection algorithms instead of the default RR algorithm, and by using energy-aware job schedulers on each cluster. The energy-aware job schedulers are compared to their normal counterparts; for example, the first bar (E-FIFO FP) means the savings compared to FIFO RR. The last three bars present the savings when using RR but with energy-aware job scheduling. It can be seen that by using energy-

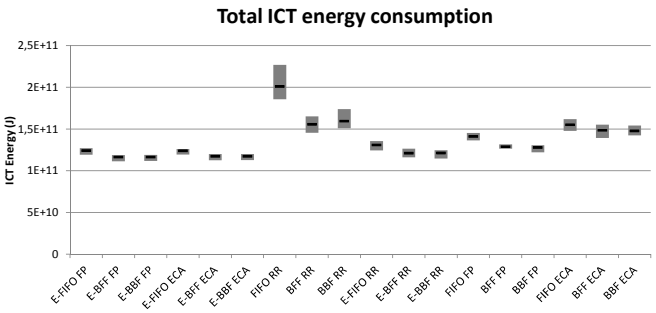


Figure 4. Total ICT energy consumption. Black lines represent the average value and the floating bars show the range of values from minimum to maximum

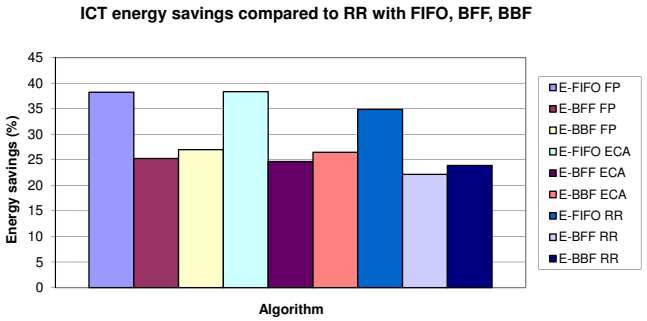


Figure 5. ICT energy savings compared to un-optimized, generally used RR with FIFO, BFF, and BBF

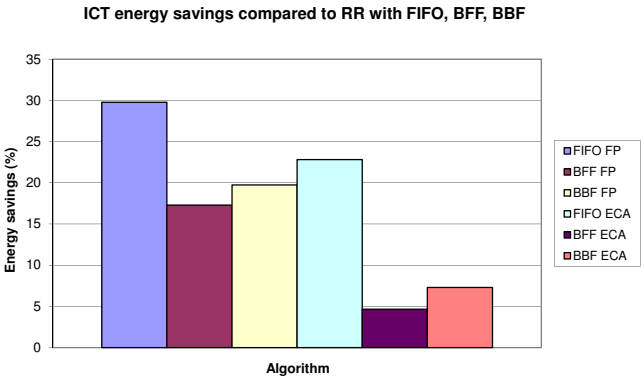


Figure 6. ICT energy savings compared to RR with normal job scheduling

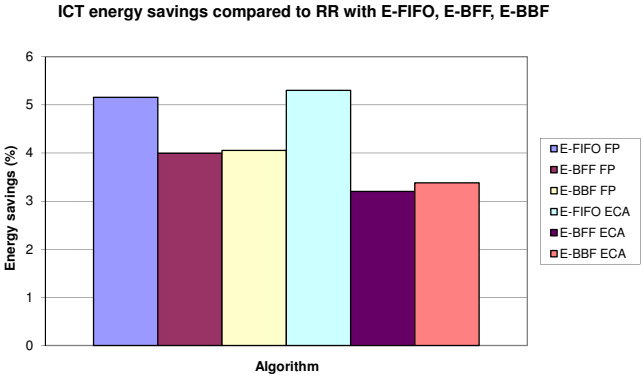


Figure 7. ICT energy savings compared to RR with energy-aware job scheduling

aware job scheduling, 22% to 35% energy savings can be achieved. Together with FP and ECA cluster selection, the savings are about 25% to 38%. When comparing the energy-aware algorithms to their normal counterparts, the savings with E-FIFO are the largest. This is because backfilling exploits the idle nodes more efficiently by running shorter jobs while with standard FIFO the nodes that cannot be used for job execution are left in an idle state and can thus be shut down by the energy-aware scheduler.

However, if we only change the cluster selection algorithm, and keep the normal job scheduling algorithms, we can see from the Figure 6 that with FP we can save 17% to 30 %. Since the cluster selection is performed before job scheduling, we can say that about 8 % of the total savings are due to the energy-aware job scheduling, while the rest is due to the FP cluster selection. When comparing to RR with energy-aware job scheduling (as depicted in Figure 7), we can see that FP and ECA cluster selection algorithms can save additionally about 3% to 5%. For the explanation, we have to take a look at the jobs' average wait and turnaround times and the simulation duration.

Figure 8 presents the average wait times of the jobs (i.e., the average waiting times of the jobs in the queue) in case of different USO cluster selection and job scheduling algorithms. As can be seen, the average wait time is clearly shorter with the FP USO algorithm. The ECA USO algorithm with backfilling has about the same average wait time as RR, even though RR with FIFO clearly has the longest waiting time. Also, there are basically no differences between RR with energy-aware and normal job scheduling. This is true also in general, as reported in [2]: energy-aware job scheduling does not cause significant increase in wait time.

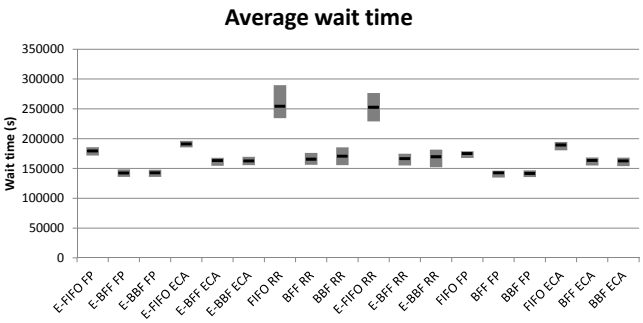


Figure 8. Average job wait times

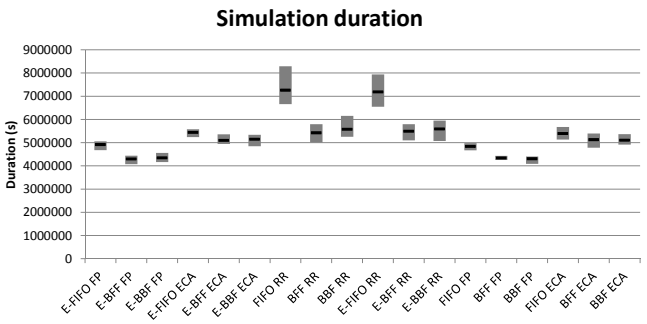


Figure 9. Average simulation duration

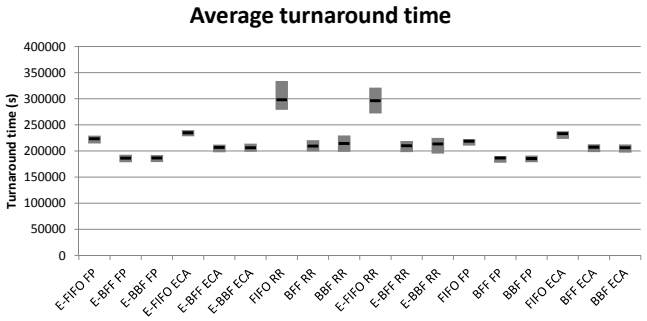


Figure 10. Average job turnaround time

Figure 9 depicts the simulation duration, i.e., how long time it took to execute all the 1500 submitted jobs. The graph shows the same as Figure 8: because the wait times are longer with RR USO cluster selection, also the simulation duration is longer.

Figure 10 presents the average job turnaround times in case of different scheduling algorithms. The story is the same as in previous figures: RR is slower due to the longer wait time.

Based on the results above, we can conclude that RR cluster selection with normal job scheduling algorithms can be very inefficient in terms of energy. This is because RR only balances the number of jobs among the clusters. It does not take into account the differences in the clusters (e.g., number of nodes/cores) or the differences in the submitted job characteristics (e.g., number of nodes/cores, walltime estimate). This can lead to a situation where one cluster is over utilized with many jobs waiting in the queue, while the other clusters can be under utilized at the same time, with nodes running idle. The energy-aware job schedulers (E-FIFO, E-BFF, E-BBF) power off the idle nodes whenever possible, and this is why a substantial amount of energy can be saved. On the other hand, the FP cluster selection algorithm inherently takes into account the differences in the clusters and submitted jobs: it always selects the cluster with the estimated minimal wait time, and thus balances the utilization between the clusters. Then fewer nodes are running idle

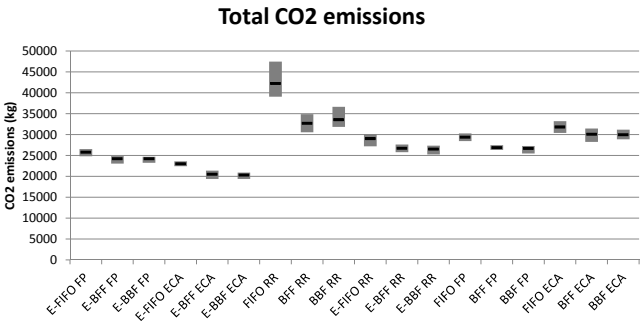


Figure 11. Total $CO_2$ emissions

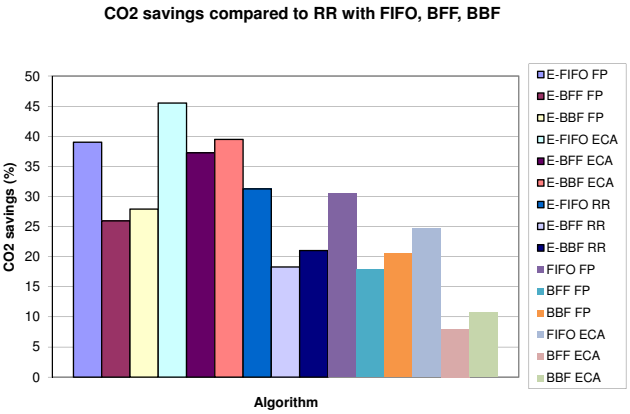


Figure 12. $CO_2$ savings compared to RR with FIFO, BFF, and BBF

and energy is saved. Also ECA saves some energy even though its goal was set to minimize the $CO_2$ emissions.

Figure 11 presents the total $CO_2$ emissions of the federated HPC data centre. As can be seen, RR with normal job scheduling causes the largest $CO_2$ emissions. Using energy-aware job scheduling reduces the emissions due to the reduced energy consumption. Using FP cluster selection reduces the energy consumption still a bit more due to the better load balancing among clusters, and thus the $CO_2$ emissions are also smaller. ECA cluster selection algorithm favours the cluster with the best CUE value, i.e., least amount of $CO_2$ emissions, and hence achieves the greatest savings in $CO_2$ emissions, about 37% to 45% compared to RR with normal job scheduling. Note that this requires also using the energy-aware job scheduler; without energy-aware job scheduling the emission reductions are smaller since ECA prefers Cluster 1 over Cluster 2 due to the smaller CUE, which in turn results in worse utilization in the bigger Cluster 2. Energy-aware job scheduling turns off the idle nodes on Cluster 2 and hence cuts the emissions. The $CO_2$ savings are depicted in Figure 12 as percentages.

## V. Testbed experiments

The energy-aware job scheduling and cluster selection algorithms were also implemented as part of the software plug-in developed within the project FIT4Green [33]. The developed plug-in [34] is a set of software components that add energy management capabilities to a data centre by interfacing with the existing management and automation tools of the data centre. Its goal is to dynamically optimize the deployment of the applications and services hosted and running across the ICT resources of a single or federated site data centre, in order to minimize the energy consumption or $CO_2$ emissions, that is, trying to consolidate load so as some hardware resources can be turned off or set to low-power state. One core software component of the plug-in is the *Optimizer* that contains the cluster selection and job scheduling algorithms and uses them together with the information on data centre resources for suggesting a list of actions that can potentially lead to reduced energy consumption.

In particular, the plug-in communicates with the RMS and UNICORE at the HPC data centre environment by acquiring information about the data centre status: the jobs in the queue, status of the nodes and the jobs that are currently running. All this information is stored and kept up-to-date in an XML-based meta-model. Based on this information, the plug-in can send actions to the RMS and UNICORE, such as, start job or shutdown node in the single site scenario. In the federated scenario, the plug-in decides to which cluster the job will be submitted to.

The plug-in can be used also in other types of data centres, i.e., in traditional service or enterprise portals, or cloud computing data centres. However, these are out of scope of this work. An interested reader can find more detailed information about the plug-in in [34]. The plug-in's source code is available at [35] as open source licensed under the Apache License, Version 2.0.

### A. Testbed scenario

This subsection describes the testbed scenario: the clusters used in the tests, their configuration and characteristics, how the tests were conducted and what kind of test workload was used.

*1) Environment and configuration:* The testbed scenario consists of three HPC clusters: Juggle, Jufit and Dune. Juggle and Jufit are located at the Jülich Supercomputing Centre in Jülich, Germany, while Dune is set up at the VTT Technical Research Centre in Oulu, Finland. By having three testbed clusters located at different sites and countries, it is possible to analyse the impact of different CUE and PUE parameters of real distributed systems. In Juggle and Jufit it is possible to set the compute nodes to a low-power standby mode, while in Dune it is only possible to shut down nodes.

Table IV
Operating numbers of the Juggle cluster

| Parameter | Value |
|---|---|
| Processor type | Dual AMD Opteron F2216 2.4GHz |
| Number of nodes | 1 head node, 12 compute nodes |
| Cores per node | 4 |
| Overall number of cores | 48 |
| Main memory | 8 GB per node |
| Network | InfiniPath(QLOGIC), Gigabit Ethernet |
| 2 file servers | disk capacity: 6 TB |
| Power supply efficiency | 83% |
| Operating system | SLES 10, Scientific Linux 5.2 |
| RMS | Torque (PBS Scheduler) |
| Node power consumption | |
| - standby | 117W |
| - idle | 162.5W |
| - maximum | 230W |

Juggle involves altogether 12 compute nodes (see Table IV). This allows executing jobs requiring many resources in parallel. The Juggle is a relatively old system compared to the other clusters in the testbed. Intelligent Platform Management Interface (IPMI) services and monitoring tools if not already provided by the operating system have been installed on each node of the system to enable the monitoring of dynamic system parameters, such as core voltage and frequency, memory load, fan RPM, and disk read/write rates.

Jufit is a more modern system providing a more modern generation of processors. It consists of two compute nodes with 12 cores each (see Table V). Compared to Juggle, Jufit is in general more energy efficient in terms of CPU power consumption and power supply efficiency. The Jufit cluster has been used in the measurements for the federated scenario.

The test environment at VTT consists of a Linux cluster framework called Dune that includes four compute nodes and a head node. All nodes are rackable Dell PowerEdge R510 servers with equal characteristics as can be seen in Table VI. There is no separate file server available in the cluster, but instead all the nodes hold adequate hard disk drives, with 1 TB of disk space.

The Torque RMS is installed on all clusters for managing nodes and the scheduling and monitoring of jobs. Furthermore, Target System Interface (TSI) modules of the UNICORE middleware are installed on the head nodes of the clusters to allow submitting jobs by UNICORE, which is needed in the case for the federated scenario.

All clusters are connected to Power Distribution Units (PDUs), which measure the power consumption of each single head and compute node. The results are requested conveniently by clients through the Simple Network Management Protocol (SNMP). The measurements were

Table V
Operating numbers of the Jufit cluster

| Parameter | Value |
|---|---|
| Processor type | Quad-core Intel Xeon X5660 (Westmere), 2.6 GHz |
| Number of nodes | 1 head node, 2 compute nodes |
| Cores per node | 12 |
| Overall number of cores | 24 |
| Main memory | 24 GB per node |
| Network | InfiniPath(QLOGIC), Gigabit Ethernet |
| 2 file servers | disk capacity: 6 TB |
| Power supply efficiency | 91% |
| Operating system | OpenSuSE 11.3 |
| RMS | Torque (Maui Scheduler) |
| Node power consumption | |
| - standby | 142W |
| - idle | 175W |
| - maximum | 232W |

Table VI
Operating numbers of the Dune cluster

| Parameter | Value |
|---|---|
| Processor type | 2 x Quad-core Intel Xeon E5606 (Westmere), 2.13 GHz, 64-bit |
| Number of nodes | 1 head node, 4 compute nodes |
| Cores per node | 8 |
| Overall number of cores | 32 |
| Main memory | 4 x 16 GB (aggregate 64 GB) |
| Network | Gigabit Ethernet |
| Power supply efficiency | 90% |
| Operating system | 64-bit Rocks cluster distribution (based on CentOS 5.6 Linux) |
| RMS | Torque (Maui Scheduler) |
| Disk space | 1 TB SATA HDD on each node |
| Node power consumption | |
| - off | 0-2W |
| - idle | 85-90W |
| - maximum | 165-175W |

updated every 3 seconds during the tests, so that the values could be provided reliably.

The motherboards of the Juggle and Jufit compute nodes support the ACPI [36] state S1, which is also known as 'standby' state. As soon as the software plug-in is generating a standby action, the appropriate compute node will be set to standby mode. While on Juggle 'wake-on-lan' is used to bring the machine back from standby to normal state, the same result on Jufit is achieved by an IPMI wake up command. The VTT testbed cluster Dune is using instead the ACPI state S5 or better known as 'soft-off', which requires a full reboot of the system.

Additionally, a DVFS feature has been enabled on Juggle to analyse the impact of using DVFS instead of ACPI energy saving mechanisms.

*2) Testing methodology:* The goal of the test measurements was to compare the energy consumption of the plug-in adapted supercomputing environments with systems using default state-of-the-art solutions. The tests tried to analyse the energy saving capabilities of the plug-in software in two different areas:

- Savings on a single cluster by using the energy-aware job scheduler of the plug-in
- Savings in a federated cluster scenario by using the cluster selection algorithms of the plug-in

For each of these investigation areas specific test approaches were carried out regarding the used benchmark workloads, type of job submission, and energy metering. All measurements depended on the available hard- and software. The workloads stressing the testing environment consisted of jobs that made use of the installed HPC applications. The jobs were either submitted directly from the test user's home directory on the head node of the cluster or alternatively from the UNICORE client, so that the user did not need to be logged in to the cluster. Both submission types are quite common in supercomputing scenarios.

The single site scenario tests were performed on the Juggle system, which provides a suitable number of nodes for testing the energy saving potential by using the plug-in's energy-aware job scheduler. Firstly, this scheduling mechanism schedules jobs in the queue of the cluster to the particular nodes and cores of the cluster, and, secondly, it sets nodes to standby if they are completely idle. Alternatively, DVFS instead of ACPI standby can be used to save energy. When having equal workloads and comparing the energy-aware scheduler with a default state-of-the-art scheduler, the best possible energy consumption depends on how efficiently the jobs can be scheduled without loss of time, so that as many nodes as possible can be set to standby. In the supercomputing testbed at FZJ the energy measurements were analysed by comparing the default PBS scheduler with the plug-in's energy-aware scheduler. While the PBS scheduler is based on an enhanced FIFO (first in, first out) algorithm, the plug-in one used the backfill first fit approach. The particular measurements were performed with workloads generating different system utilization profiles on the Juggle cluster (0%, 40%, 60%, and 80%) to simulate potential loads of real supercomputing machines.

The total energy consumption generated by a single test workload has been calculated in Joule as a product of the measured average power of all cluster nodes and the elapsed time that was needed to run all jobs of the workload. The elapsed time involves the total time elapsed from the submission of the test user's first job until the output files of the last executed job have been stored where requested. So, this period includes the time for transferring input files, the wait time in the RMS queue, the actual execution time, and the time to stage the output files to the requested locations. Each measurement was stopped immediately after all jobs

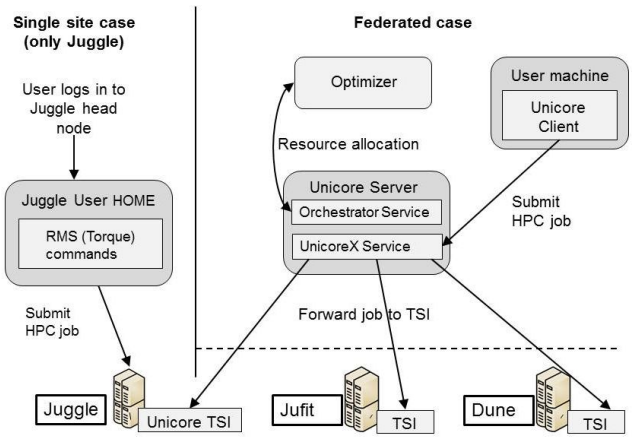


Figure 13. Difference of job submission in single and federated benchmark tests

of the test workload were finished. This approach was mandatory to measure the time that different scheduling strategies need to process the workload. So, the energy of one measurement was calculated as follows:

$$E_{SingleSite} = T_{Workload} * P_{Cluster}, \quad (8)$$

where $E_{SingleSite}$ is the total energy of the site, $T_{Workload}$ is the elapsed time to process the workload, and $P_{Cluster}$ is the average power consumption in the cluster during the workload processing.

Single site scenario tests were also performed on the Dune cluster following the same testing methodology. The only differences were that Dune supports only soft-off instead of standby and DVFS, and the comparison point was Maui scheduler instead of PBS scheduler.

When performing tests in the single site scenario, the jobs of the workload were submitted locally from the cluster's head node by using provided shell commands of the RMS on the dedicated cluster. This approach is a common practice in supercomputing environments when the dedicated target system of the job is already known. In the federated scenario another job submission approach was utilised. Since it is not known in advance on which cluster the job should be executed, the test user submits the jobs at first to a UNICORE server, which acts as a centralized entry point for all incoming jobs. This UNICORE instance is connected to the installed UNICORE TSIs on the testbed clusters, so that the server is able to submit incoming jobs to the RMS of an appropriate machine. Before that, the USO, service of the UNICORE server, initiates a resource allocation request to ask the plug-in for a suitable target machine for the job. Figure 13 highlights the difference in terms of job submission between the single and federated scenario.

In general, the workloads have been compared in each test case by using the plug-in's scheduling strategies as well as default mechanisms for getting results of how efficiently the plug-in is able to save energy. In the case when the plug-in was not used, the existing state-of-the-art mechanisms have been used to process the jobs on the test clusters.

The approach of the energy consumption measurement in the federated scenario is to consider only the elapsed time, which is needed on each testbed cluster to run the assigned jobs of the benchmark workload. This incorporates that each involved cluster can produce in one benchmark measurement a different elapsed time and different average power consumption. So, the total energy consumption $E_{Federated}$ in one measurement is the product of the elapsed time and the average power of each cluster $i$:

$$E_{Federated} = \sum_{i=1}^{N} T_{Cluster_i} * P_{Cluster_i}. \quad (9)$$

*3) Test workload:* The benchmark measurements on the testbed were aimed at stressing the testbed as close as possible to clusters in real supercomputing environments. For that purpose different typical HPC applications were installed on the test clusters, namely LINPACK [37], a collection of FORTRAN subroutines to solve linear systems, and PEPC (Pretty Efficient Parallel Coulomb Solver) [38], which is used to run astrophysical N-body simulations.

For the single site scenario the test workloads were created by a configurable Perl script, which can parameterise the jobs in terms of the used HPC application, the level of computation intensity, the number of used nodes and cores, as well as the planned walltime (also known as wall clock time), which is the time elapsed until a job should have been finished. In this way workloads were created stressing the system with different system utilization in order to analyse the energy savings under those different loads. Also real world clusters working in production show often varying system utilizations between entirely idle and working to capacity. The utilization factor is defined here as the percentage of time when cluster resources are stressed with jobs relative to the total elapsed time of the workload. For instance, a system load of 90% means that only on an average of 10% of the elapsed time cluster nodes are able to be set to the energy saving standby mode because they are otherwise busy with running jobs.

In case of the federated scenario the workloads of the single site scenario were adapted as a template to create workloads using the same HPC applications within the graphical UNICORE Rich Client (URC) [39]. This workload is embedded in a workflow from where the single jobs are submitted in parallel to the UNICORE server, which in turn initiates resource allocation requests to the plug-in's cluster selection algorithms, and forwards

Table VII
DUNE - NUMERICAL RESULTS OF SINGLE SITE MEASUREMENTS

| System utilization [%] | 0% | 50% | 80% | 90% |
|---|---|---|---|---|
| **Energy with plug-in [kJ]** | 11.8 | 133.2 | 433.4 | 480.5 |
| Elapsed Time [s] | 517.8 | 543.2 | 1276.9 | 1276.7 |
| Average Power [W] | 22.8 | 250.8 | 339.6 | 376.6 |
| **Energy without plug-in [kJ]** | 176.9 | 254.5 | 499.1 | 515.9 |
| Elapsed Time [s] | 515.7 | 648.0 | 1258.3 | 1268.0 |
| Average Power [W] | 343.4 | 393.0 | 396.8 | 406.9 |
| **Energy saving [%]** | 93.3 | 47.7 | 13.2 | 6.9 |

subsequently the jobs to the chosen cluster.

Roughly 90% of the workload jobs can run on both clusters, while the requirements of always 10% of the jobs can only be met on one of the clusters. This distribution creates a base load on the clusters to map better real world environments where jobs are usually not able to run on every available target machine.

### B. Testbed results

This subsection presents the results of the testbed experiments for both single and federated site scenarios.

*1) Single site scenario:* In our previous work [2], we showed the potential energy savings in a testbed that considered the ACPI standby mechanism on the compute nodes. In this work, we have performed additional single site measurements on the new VTT testbed cluster Dune, which is using ACPI S5 soft-off power shut down to save energy on unused nodes, while on Juggle and Jufit only ACPI standby is available. ACPI standby is faster than soft-off in switching to the power-on state but, on the other hand, does not have the same power saving potential. The results can be found in Table VII. Each workload measurement was repeated with several iterations.

From the results it is possible to note that the energy savings are highly dependent on the test workload and the utilization of the whole system. The single site optimization algorithm attempts to keep the system active with the lowest possible amount of resources and still providing suitable turnaround times for the jobs. With lower utilization values the energy savings are higher, but if the system is very busy there are not many opportunities to shut down idle resources.

The single site scenario tests at FZJ were performed on the Juggle system, which provides a suitable number of nodes for testing the energy saving potential by setting nodes to standby status. However, apart from the different ACPI mechanisms it was worth to evaluate the energy saving potential of using the DVFS feature, which works on the principle of setting unused nodes to the powersave governor, i.e., the lowest CPU frequency and core voltage is set on the node. The performance governor is set again by the plug-in once the nodes are requested again by jobs. That setting implies that the maximum available frequency is set on the CPUs of the nodes. On-demand governors were consequently not used in the configuration, since it is in general not desirable to use frequency scaling when running CPU intensive jobs. Administrators experienced performance drawbacks with on-demand governors, which is a critical issue in HPC environments. The software plug-in on Juggle has thus been enabled to make use of different energy saving methods:

- ACPI power saving statuses (e.g., standby or soft-off)
- DVFS (switching between powersave and performance governor)
- ACPI standby + DVFS (observed slightly higher savings on the testbed cluster when using both mechanisms in parallel).

In general, the energy measurements have been analysed by comparing the PBS default scheduler with the the plug-in's energy aware job scheduler. These measurements were performed with workloads generating a different total load on the cluster (0%, 40%, 60%, and 80%). Each workload measurement was repeated with several iterations. The results in Table VIII show the average values of those tests.

The energy consumption of a single workload has been calculated in Joule as a product of the measured average power of all cluster nodes and the elapsed time, which was needed by the appropriate workload. The elapsed times of each workload depend on the composition of the workload to achieve certain system utilization, so there is no correlation between the elapsed time values of different system loads. In contrast, the average power consumption increases with more intensive workloads, since less idle nodes can be set to an energy saving status.

When comparing the values of default scheduler measurements with energy-aware scheduler tests it is apparent that the elapsed time is most time slightly higher with the energy-aware strategies, which is caused by the overhead of the optimization process. In particular, cluster information such as node and job statuses must be read and analysed at the remote plug-in server, and generated actions must be sent back to the RMS of the cluster. However, this process has been optimized in terms of performance during the implementation phases, so that the time impact has been minimized. Overall, the expected overhead in the single site scenario is round about 2-3% compared to default mechanisms.

Taken into account similar elapsed time results for

Table VIII
JUGGLE - NUMERICAL RESULTS OF SINGLE SITE MEASUREMENTS

| System utilization [%] | 0% | 40% | 60% | 80% |
|---|---|---|---|---|
| **Energy with standby [kJ]** | 1300 | 2652 | 3018 | 4410 |
| Elapsed Time [s] | 1000 | 1634 | 1580 | 2143 |
| Average Power [W] | 1300 | 1623 | 1910 | 2058 |
| **Energy with DVFS enabled [kJ]** | 1400 | 2701 | 3033 | 4433 |
| Elapsed Time [s] | 1000 | 1556 | 1558 | 2126 |
| Average Power [W] | 1400 | 1736 | 1947 | 2085 |
| **Energy with standby+DVFS [kJ]** | 1280 | 2616 | 3003 | 4389 |
| Elapsed Time [s] | 1000 | 1620 | 1579 | 2144 |
| Average Power [W] | 1280 | 1615 | 1902 | 2047 |
| **Energy with default scheduler [kJ]** | 1960 | 3304 | 3345 | 4625 |
| Elapsed Time [s] | 1000 | 1554 | 1552 | 2088 |
| Average Power [W] | 1960 | 2126 | 2155 | 2215 |
| **Energy saving with standby [%]** | 33.7 | 19.7 | 9.8 | 4.6 |
| **Energy saving with DVFS [%]** | 28.6 | 18.2 | 9.3 | 4.2 |
| **Energy saving with standby + DVFS [%]** | 34.7 | 20.8 | 10.2 | 5.1 |

the single site measurements, the main factor for saving energy is clearly the measured average power of the tested cluster, which is proportional in the measurements to the decreasing system load. While the default RMS scheduler cannot make advantage of idle compute nodes, the plug-in sets them in an energy saving standby mode, which reduces noticeably the average power of the system.

When using only DVFS we could observe energy saving between 28.6% and 4.2% compared to the default scheduler depending on the generated system utilization. When enabling DVFS and ACPI standby together we could even achieve between 34.7% and 5.1%, which confirmed the assumption that DVFS + standby generates a slightly lower consumption than ACPI standby alone (between 33.7% and 4.6%). However, this behaviour could only be detected on the used testbed hardware. It cannot be stated as a general rule. Nevertheless, administrators can check their systems if it makes sense to use both mechanisms in parallel. On Juggle an additional gain of about 1% saving is possible when using both features. Using DVFS alone could be a benefit when high job fluctuations can be expected on a system. Especially, when the hardware is supporting only a full shut down ACPI status, which would need 2-3 minutes for powering the system on again, and a high job submission rate is supposed, it could be more efficient to use the supported DFVS feature. That mechanism needs only one second to switch between the governors.

*2) Federated scenario:* In the supercomputing federated scenario we wanted to measure the emission and energy saving capabilities of the plug-in's cluster selection strategies. Jobs should be assigned to suitable cluster resources in the most energy efficient way. The plug-in's Optimizer implements two different strategies to achieve that resource allocation, namely the ECA and FP cluster selection algorithms.

Section III gives already a detailed description of both strategies. In short, the FP algorithm calculates the wait time of a job on all potential suitable clusters and submits the job to the system that provides the most minimal estimated queue time. In contrast, the ECA algorithm estimates at first the energy or respectively the emission (depending on the chosen objective), which would be produced by a job on a particular cluster. The energy/emission is calculated by considering the PUEs/CUEs of the clusters as well as estimating the ICT energy consumption of particular. In the supercomputing testbed at FZJ, PUE and CUE indexes are equal for both testbed clusters, since both machines are located in the same data centre environment. However, the Dune cluster of VTT is located at Oulu in Finland and provides different PUE and CUE specific values.

Additionally, the ECA algorithm checks if the user defined 'latest job finishing time' can be satisfied, i.e., we used user defined allowed delay value for load balancing between clusters. This means that the plug-in verifies if the job can be executed and finished on a cluster within a user defined time limit. If not, the job cannot be scheduled in the best energy efficient way and the next cluster is chosen, where the estimated wait time is smaller than the user defined value. By this mechanism, the user can set a threshold value from where jobs must not be scheduled energy efficiently anymore.

The crucial factor in terms of energy efficiency is the power consumption of the testbed clusters over a dedicated time. Furthermore, it was worth to analyse the impact of an application benchmark parameter, which takes into account the different performance of an application produced on a particular cluster.

Scheduling jobs in a federated environment of different supercomputers is not yet very common in HPC. Resource brokering in heterogeneous environments is still an on-going research area in HPC. Meta-schedulers are usually not yet deployed in real production environments but rather in smaller research and test environments. One of the main barriers is that HPC job requirements are often strongly system-related so that the jobs can only run in a dedicated hard- and software environment. There is also a lack of dynamic resource information

on the broker level about node and queue statuses of the bounded supercomputers. Usually, users have a clear picture where to submit their jobs. Often these jobs can only run on particular nodes, since only there the required application software is installed.

Accordingly, there are no matured solutions of meta-schedulers available that could be used as reference software. So, in the testbed experiments we compared our solution with a simple round-robin scheduling mechanism provided by the UNICORE software. This solution does not consider any dynamic system information about jobs in queues or the statuses of nodes. A more intelligent solution for scheduling jobs in federated environments is the developed plug-in's FP algorithm, which does not take into account energy or emission aware parameters but only the fastest execution time of jobs on the clusters. So the focus of the federated scenario evaluation was not only to compare ECA and FP with default RR algorithm, but to check the impact of the provided scheduling parameters on the results. In the federated tests the energy-aware job scheduling used in the single sites assessment was deactivated to set the focus on the energy saving potential of the federated algorithms.

For analysing the impact of different PUEs on the sites we changed the PUE value of the FZJ site in each trial and used a fixed value on the VTT site where Dune is located, so that we could analyse the system utilization of the clusters while changing the PUE difference of them. The PUE value on Dune, which was calculated by VTT administrators, was set to 1.2. For the FZJ site we actually calculated a value of 1.4. In the evaluation we iterated the FZJ parameter from 1.0 to 1.8.

Figure 14 shows clearly the impact on the utilization of the particular testbed clusters when iterating over the FZJ PUE value. The graph of the Juggle is relatively constant, since the Optimizer schedules only some default jobs to the system, which can run only there (requesting 8 nodes per job). Apart from these jobs no others are submitted to that machine, since the Optimizer considers its high basic power consumption compared to the other two clusters. In contrast, Jufit is getting at first the most jobs of the workload when having a lower PUE than Dune which was set to 1.2. The consequence is at the beginning higher system utilization on Jufit compared to Dune.

When Jufit and Dune have the same PUE (1.2) Dune is already preferred clearly by the Optimizer since it has the most efficient power consumption per compute node in the testbed. Higher PUEs for FZJ-Jufit result in even lower utilizations for Jufit and higher ones for Dune. With more than 1.4 PUE on Jufit saturation on the utilization of Dune can be detected. Dune is utilized almost with 100% and also Jufit levels out at about 35%.

The CUE evaluation was performed in a similar way

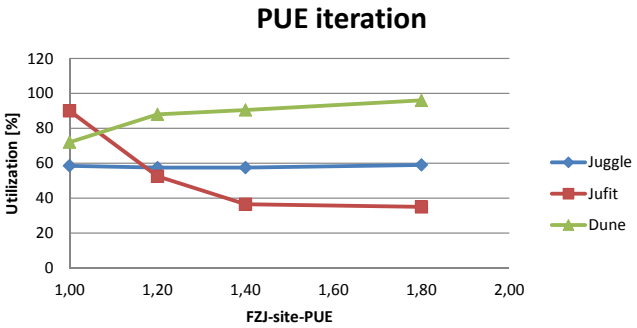


Figure 14. Impact on cluster utilization when iterating over PUE

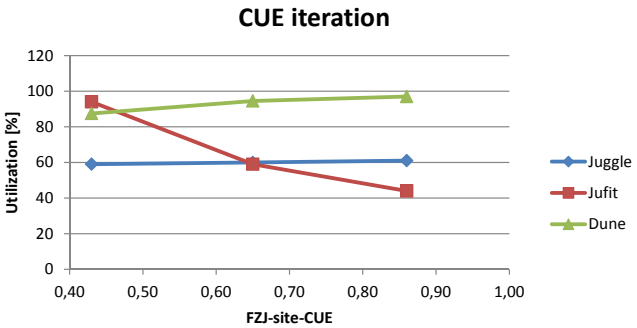


Figure 15. Impact on cluster utilization when iterating over CUE

as the PUE iteration. After changing the Optimizer objective to $CO_2$ emissions, the CUE parameter of the available two sites was used for calculating the job scheduling in the federated scenario tests. For Dune a CUE value of 0.43 was calculated based on local energy and emission characteristics. For the FZJ site we ascertained a value of 0.8. Again, for analysing the impact of different CUEs, we iterated over different CUEs at the FZJ site.

The test results in Figure 15 show that the Juggle utilization is again constant with different CUEs since the Optimizer schedules only some workload jobs to Juggle which can only run on that machine. Apart from these jobs no other ones were assigned to that cluster since its basic energy consumption is too high compared to the other testbed machines. Jufit shows a slightly higher utilization than Dune when the CUE value is on a similar level than the Dune one. However, this effect is reversed when the CUE value of FZJ site is increased. The Optimizer calculates then a higher emission on Jufit and generates therefore a higher utilization on Dune for meeting the $CO_2$ emission objective.

The runtime of a job on a dedicated machine can have an important impact on the overall energy consumption. Therefore, the plug-in's application benchmark feature was introduced to map the different performance of supercomputers when running jobs with known HPC

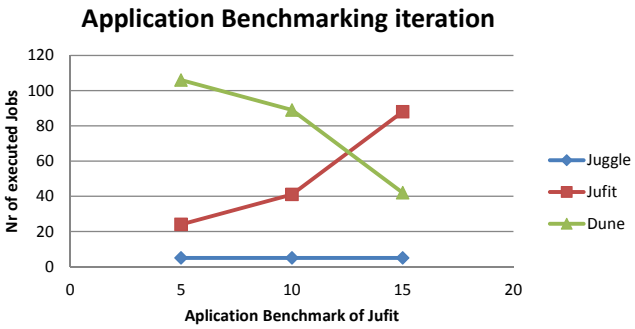


Figure 16. Impact of application benchmarking on job scheduling

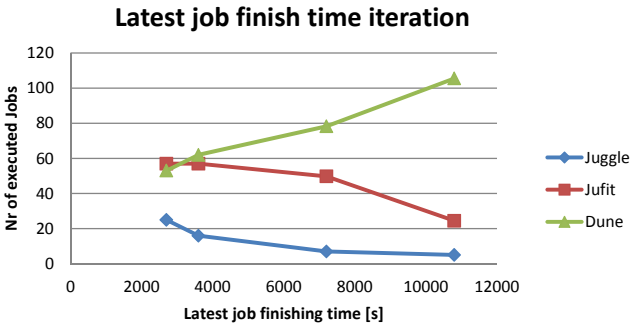


Figure 17. Job distribution and latest job finishing time

applications. The mapping is implemented by adding an application ID and a corresponding integer value, which indicates how efficiently the application can be executed.

In the test trials we used the LINPACK application for the evaluation of that parameter. LINPACK is installed on all available testbed clusters. Test benchmarks showed that all machines provide a different performance when executing jobs using the same LINPACK configuration. Finally, after measuring the execution time of the test jobs we were able to define different benchmark values for all the clusters of the testbed. Nevertheless, we wanted to evaluate not only this benchmark distribution but the impact of the application benchmarking on the job distribution and system utilization of the testbed. So, we performed additional tests with changing the benchmark parameter of the Jufit cluster and measuring the number of executed jobs on the clusters when iterating over different parameters (5, 10, and 15).

Figure 16 shows the distribution of the jobs during these different iterations. The Juggle cluster receives only a few jobs for all iterations. More revealing results could be detected in that evaluation on Jufit and Dune. When having similar benchmarks (Jufit 5, Dune 4) Dune receives clearly most of the workload jobs, since there is no big difference in that parameter between both machines and Dune is preferred by the Optimizer because of the node's better power consumption. When increasing the Jufit benchmark parameter from 5 to 15 it can be detected that Jufit gets as more jobs from the Optimizer as higher the benchmark parameter has been set. Finally, a benchmark value at Jufit of 3 times higher than the Dune value results that the actually more power efficient Dune machine is overpowered by the much better benchmark efficiency of the Jufit machine, i.e., even if Jufit consumes more power, it consumes less energy since it is expected to execute the job much faster than Dune. This shows that the benchmark application parameter can have a strong impact on the job distribution, which again affects the energy consumption of the federated clusters.

The Optimizer checks additionally if the user defined 'latest job finishing time' can be satisfied. This means that it is checked if the job can be executed and finished on a cluster within a user defined limit. If not the job cannot be scheduled in the most energy efficient way and the next cluster is chosen where the estimated wait time is smaller than the user defined value. By this mechanism the user can set a threshold value as part of the given job description from where on jobs must not be scheduled energy efficiently anymore. It is assumed that users will set a multiple value of the wall time of a job when considering the latest job finishing time. For example, if a job gets a walltime of 8 hours, it can be assumed that the user expects in general some wait time before his job can be executed. So, we analysed the impact of the latest finishing parameters which were set between three and ten times as high as the average walltime of the submitted workload jobs. Figure 17 shows the results of the job distribution when iterating the same workload with different job finishing parameters.

When relatively short latest job finishing times were set by users the Optimizer is not able to schedule most of the jobs in an energy efficient way. Taking into account the power estimation the Optimizer submits the jobs at first to Dune, calculates then that the overall finishing time (wait time + runtime) of the other waiting jobs would exceed their latest finish time, and thus will submit new jobs to other clusters until Dune provides again time slots for waiting jobs. In that way, it can be detected that even the Juggle cluster, which is usually underutilized by the federated plug-in scheduling algorithms because of its bad energy efficiency, gets more jobs as usual with low latest job finishing time parameters. With increasing latest job finishing times the plug-in has a greater margin to schedule jobs to energy efficient clusters, which is the Dune cluster at VTT in this evaluation.

In general, the latest job finishing time parameter is very important to guarantee as much as possible a fair sharing of the federated resources when there is high system utilization. On the other hand this parameter

helps to submit jobs each time to the most efficient cluster as long as the user's latest job finishing time can be satisfied. This condition can be rather satisfied in underutilized federated resources. Using additionally some of the developed single site algorithms to save energy on these underutilized resources would lead to further energy savings.

In the final assessment we analysed the overall energy and emission saving potential of the plug-in's federated algorithms. For this purpose, we used the constructed PUE and CUE values of the two testbed sites FZJ and VTT. While at VTT a CUE of 0.43 and a PUE of 1.2 has been defined, we set at Jufit and Juggle a CUE of 0.8 and a PUE of 1.4. For the LINPACK application benchmark parameter we used the results of our test measurements to detect the most efficient benchmark distribution, which is Juggle=1, Jufit=10, and Dune=4.

Concerning the latest job finishing time parameter we compared the energy consumption of different values since this parameter reflects very reasonably the overall utilization of the federated resources. The results were compared at first glance with the default round-robin strategy of the UNICORE middleware. However, the round-robin strategy is a bad algorithm in terms of energy efficiency. It just submits the jobs by considering time slots in the scheduling calculation and distributes thus the jobs pretty equally to the resources without regarding available idle resources, job queue information, and power consumption.

A much better algorithm in terms of scheduling jobs in federated environments without regarding the power consumption, PUE, and CUE values is the FP algorithm, which is a good reference point when comparing energy-aware- with non-energy-aware-algorithms in federated environments.

Table IX shows the results regarding energy consumption and produced emissions when using the plug-in for scheduling jobs efficiently in federated environments. The first column shows the measurement that was performed with a relatively low latest finishing time parameter of 3600 s while running test jobs with an average walltime of 720 seconds. Compared to the USO round-robin strategy one could save up to 52% energy when using the plug-in. When the latest finishing time is increased to 10800 s, this saving was even raised to 67%. This seems to be a very high saving. The reason is that the round-robin strategy submits in contrast to the plug-in a lot more jobs to Juggle. This results obviously in a much higher energy consumption of the Juggle nodes. Whereas more jobs on Jufit or Dune only raise the energy consumption moderately, on Juggle more jobs generate a significantly higher consumption.

The result of a measurement with the plug-in's FP algorithm is listed in the third column. Also that strategy schedules some jobs to Juggle. However, in that case, the waiting time of the queued jobs is checked continuously by the Optimizer, so that much less jobs run at the end on that machine. Moreover, considering the elapsed times of the workload on the particular clusters it can be detected that the total summarized elapsed time is shorter compared to the energy aware algorithm. Nevertheless, the FP algorithm is not as energy efficient as the energy aware strategy. At the end, the higher energy consumption of the most inefficient cluster is particularly disadvantageous for the total workload energy consumption. So, compared with the FP strategy and using same latest finishing times, the ECA algorithm saves about 21% energy. In the federated scenario, the energy-aware job scheduler (if used) causes the same 2-3 % overhead as in single site scenario. Additionally, FP and ECA algorithms cause some more overhead compared to the simpler round-robin algorithm that does not need any dynamic information from the clusters. Clearly, the overhead is much smaller than the achieved savings.

Table IX
Energy and Emissions in HPC federated measurements

| Latest finish time [s] | ECA 3600 | ECA 10800 | FP 10800 | RR - |
|---|---|---|---|---|
| **Juggle [kJ]** | 4668 | 1916 | 3705 | 12893 |
| CUE | 0.80 | 0.80 | 0.80 | 0.80 |
| PUE | 1.40 | 1.40 | - | - |
| Jobs | 16 | 5 | 11 | 43 |
| Elapsed Time [s] | 2979 | 1245 | 2434 | 8181 |
| Average Power [W] | 1567 | 1540 | 1522 | 1576 |
| Utilization [%] | 66 | 56 | 46 | 81 |
| **Jufit [kJ]** | 961 | 767 | 1051 | 962 |
| CUE | 0.80 | 0.80 | 0.80 | 0.80 |
| PUE | 1.40 | 1.40 | - | - |
| Jobs | 57 | 25 | 61 | 45 |
| Elapsed Time [s] | 2940 | 2536 | 3213 | 2838 |
| Average Power [W] | 327 | 304 | 327 | 339 |
| Utilization [%] | 65 | 49 | 93 | 45 |
| **Dune [kJ]** | 1442 | 2163 | 1426 | 1029 |
| CUE | 0.43 | 0.43 | 0.43 | 0.43 |
| PUE | 1.20 | 1.20 | - | - |
| Jobs | 62 | 106 | 63 | 47 |
| Elapsed Time [s] | 3163 | 4721 | 3211 | 2839 |
| Average Power [W] | 456 | 458 | 444 | 430 |
| Utilization [%] | 82 | 97 | 88 | 79 |
| **Total cluster [kJ]** | 7072 | 4846 | 6181 | 14884 |
| **Saving to FP [%]** | - | 21.59 | - | - |
| **Saving to USO [%]** | 52.49 | 67.44 | - | - |
| **Total cluster emissions [kgCO2eq/kwh]** | 1.42 | 0.86 | 1.31 | 3.62 |

## VI. Conclusion and Future Work

The results show that the generally used round-robin cluster selection algorithm can lead to unbalanced utilizations among clusters. This can be very inefficient in terms of energy consumption and $CO_2$ emissions. Using energy-aware job scheduling to power off idle computing nodes whenever possible greatly enhances the energy-efficiency. Load can also be balanced by replacing round-robin cluster selection by the Fastest possible selection algorithm. This leads to energy savings due to the better utilization of clusters and shorter wait times. Using both energy-aware job scheduling and FP cluster selection simultaneously leads to greater energy savings than using only one of them. The greatest $CO_2$ emission savings can be achieved by using ECA cluster selection algorithm to favour the cluster with least $CO_2$ emissions. The actual savings in each case depends on the cluster and job characteristics. In these simulations, for example, the energy sources were chosen so that one cluster had rather small CUE, another one rather big CUE, while the third one was something between them. With smaller differences in CUE, also the possible savings in $CO_2$ emissions would be smaller.

Based on the simulation results presented above, we propose to use FP cluster selection algorithm for the jobs without green SLA, since it leads to energy and $CO_2$ emission savings due to the better utilization of the clusters, and to better QoS due to the shorter wait time. For the jobs with green SLA, we propose to use the ECA cluster selection algorithm, since it can lead to even greater $CO_2$ emission savings than FP, while keeping the QoS (in terms of time) at the user specified level. It can be used also without green SLA, if some other parameter (e.g., queue size limit) is used for load balancing to prevent excessive load on the "greenest" cluster.

The results of the single site testbed assessment, with a focus on a DVFS feature, confirmed the experiences of our previous work [2] to the effect that the energy saving potential when using energy-aware scheduling increases with decreasing system utilization. When having loads above 80%, the saving potential is very limited (<10%). On the other hand utilizations below 60% show more reasonable savings (10% to 34%). The highest possible energy saving on the Juggle cluster was 34.7%, which happens when the system is completely idle. That value is limited by the fact that a compute node that was set to ACPI state standby cannot save more than 35% energy. It remains to be stated that the higher the system utilization the less compute nodes can be set to an energy saving status.

For the federated testbed results, it must be stated that the energy saving potential is absolutely dependent on the available hardware. A slow and high-consumption cluster as Juggle in a federated supercomputing environment can have a major impact on the results. If we had used three similar testbed clusters regarding their energy consumption, it can be expected that the energy saving potential would have been far smaller. In that case also a round-robin strategy would have produced better results and the FP strategy in the same way. Nevertheless, the performed test studies revealed many new insights into the saving potential of scheduling jobs energy efficiently in federated supercomputing environments. The results obtained can be used as a basis for the design of specific federated cluster environments when using the developed plug-in to enable an energy-aware scheduling of the resources.

The simulation studies and testbed experiments were performed with a different set of parameters e.g., hardware, type of jobs and cluster configurations. However, it is possible to note from both results that there is an energy saving potential in federated HPC environments that is dependent on the available hardware and cluster utilization. A poor utilization of clusters can ultimately lead to an increase in energy and emissions.

Previous research in the energy-efficiency of HPC grid computing has mainly focused on performing optimizations inside a single data centre. This work presented a global view by taking into account the whole grid: the characteristics of the data centres, compute nodes and the computing hardware. The most comparable approach to our work is HAMA, described in [14]. The results of HAMA are similar to our approach: energy savings are between 23% and 50%.

Recently, the plug-in's energy-aware job scheduler has been enhanced by adding several additional scheduling policies that are available in commercial schedulers like Moab. The new supported policies are mainly for enhancing the QoS for the users, including for example fair share and job exclusive policies. Fair share policy prevents a single user from conquering all the nodes with multiple jobs if there are also other users' jobs waiting in the queue. Job exclusive policy means that the job will get all the resources of the entire node for itself; only one job per node is then allowed. Our future work include testing the effect of these policies to the energy and emission savings.

## Acknowledgment

This work extends our earlier work [1]; the invitation to submit this extended version is highly appreciated. This work was supported by EU FP7 project FIT4Green [33]. The authors would like to thank all the colleagues that have worked in the project, as well as the anonymous reviewers for their comments.

## References

[1] M. Majanen and O. Mämmelä, "Optimization of Energy and Emissions in High-Performance Grid Computing Data Centres," in *The Second International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies (ENERGY 2012)*, St. Maarten, Netherlands Antilles, Mar. 2012.

[2] O. Mämmelä, M. Majanen, R. Basmadjian, H. Meer, A. Giesler, and W. Homberg, "Energy-aware job scheduler for high-performance computing," *Computer Science - Research and Development*, vol. 27, pp. 265–275, 2012. [Online]. Available: http://dx.doi.org/10.1007/s00450-011-0189-6

[3] Y. Liu and H. Zhu, "A survey of the research on power management techniques for high-performance systems," *Softw. Pract. Exper.*, vol. 40, pp. 943–964, October 2010.

[4] Gartner. (2012, Sep.) Gartner Estimates ICT Industry Accounts for 2 Percent of Global $CO_2$ Emissions. [Online]. Available: http://www.gartner.com/it/page.jsp?id=503867

[5] The Climate Group, "SMART 2020: Enabling the low carbon economy in the information age," Tech. Rep., 2008.

[6] D. Erwin and D. Snelling, "UNICORE: A Grid Computing Environment," in *Euro-Par 2001 Parallel Processing*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2001, vol. 2150, pp. 825–834.

[7] C. Belady, "Green Grid Data Center Power Efficiency Metrics: PUE and DCIE," Green Grid, Tech. Rep., 2008.

[8] M. Di Girolamo and et. al., "Final evaluation report on pilots of full-featured enhanced control plug-in and control desk for federated data centre," FIT4Green, Deliverable D6.4 v2.0, Jul. 2012, http://www.fit4green.eu/.

[9] S. Hong and H. Kim, "An integrated gpu power and performance model," in *Proceedings of the 37th annual international symposium on Computer architecture*, ser. ISCA '10. New York, NY, USA: ACM, 2010, pp. 280–289. [Online]. Available: http://doi.acm.org/10.1145/1815961.1815998

[10] D. Li, S. Byna, and S. Chakradhar, "Energy-aware workload consolidation on gpu," in *2011 40th International Conference on Parallel Processing Workshops (ICPPW)*, Sep. 2011, pp. 389 –398.

[11] K. Ma, X. Li, W. Chen, C. Zhang, and X. Wang, "Greengpu: A holistic approach to energy efficiency in gpu-cpu heterogeneous architectures," in *2012 41st International Conference on Parallel Processing (ICPP)*, Sep. 2012, pp. 48 –57.

[12] Moab Green Computing. (2012, Dec.). [Online]. Available: http://www.adaptivecomputing.com/resources/docs/mwm/archive/6-0/18.0greencomputing.php

[13] F. Ehmke, "Moab evaluation," University of Hamburg, Project Report, Dec. 2011, http://wr.informatik.uni-hamburg.de/_media/research/labs/2011/2011-09-florian_ehmke-moab_evaluation-report.pdf.

[14] S. K. Garg and R. Buya, "Exploiting Heterogeneity in Grid Computing for Energy-Efficient Resource Allocation," *Seventeenth Annual International Conference on Advanced Computing and Communications (ADCOM 2009)*, 2009.

[15] T. Lynar, R. Herbert, Simon, and W. Chivers, "Reducing grid energy consumption through choice of resource allocation method," *2010 International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW)*, May 2010.

[16] C. Patel, R. Sharma, C. Bash, and S. Graupner, "Energy aware grid: Global workload placement based on energy efficiency," Hewlett Packard, HP Laboratiories Palo Alto, Tech. Rep., November 2002.

[17] A. J. Shah and N. Krishnan, "Optimization of global data center thermal management workload for minimal environmental and economic burden," *IEEE Transactions on Components and Packaging Technologies*, vol. 31, no. 1, pp. 39–45, March 2011.

[18] G. Da Costa, J.-P. Gelas, Y. Georgiou, L. Lefevre, A.-C. Orgerie, J.-M. Pierson, O. Richard, and K. Sharma, "The GREEN-NET Framework: Energy Efficiency in Large Scale Distributed Systems," *HPPAC 2009 : High Performance Power Aware Computing Workshop in conjunction with IPDPS 2009*, May 2009.

[19] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Geographical load balancing with renewables," *SIGMETRICS Perform. Eval. Rev.*, vol. 39, no. 3, pp. 62–66, Dec. 2011. [Online]. Available: http://doi.acm.org/10.1145/2160803.2160862

[20] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi, "Capping the brown energy consumption of internet services at low cost," in *Proceedings of the International Conference on Green Computing*, ser. GREENCOMP '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 3–14. [Online]. Available: http://dx.doi.org/10.1109/GREENCOMP.2010.5598305

[21] I. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Greenhadoop: leveraging green energy in data-processing frameworks," in *Proceedings of the 7th ACM european conference on Computer Systems*, ser. EuroSys '12. New York, NY, USA: ACM, 2012, pp. 57–70. [Online]. Available: http://doi.acm.org/10.1145/2168836.2168843

[22] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, ser. SIGMETRICS '11. New York, NY, USA: ACM, 2011, pp. 233–244. [Online]. Available: http://doi.acm.org/10.1145/1993744.1993767

[23] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser, "Renewable and cooling aware workload management for sustainable data centers," in *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '12. New York, NY, USA: ACM, 2012, pp. 175–186. [Online]. Available: http://doi.acm.org/10.1145/2254756.2254779

[24] I. Goiri, K. Le, M. Haque, R. Beauchea, T. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Greenslot: Scheduling energy consumption in green datacenters," in *2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, Nov. 2011, pp. 1 –11.

[25] S. Klingert, T. Schulze, and C. Bunse, "GreenSLAs for the eco-efficient management of data centres," in *2nd International Conference on Energy-Efficient Computing and Networking 2011 (E-energy 2011)*, New York, NY, USA, 2011.

[26] C. Belady, D. Azevedo, M. Patterson, J. Pouchet, and R. Tipley, "Carbon Usage Effectiveness (CUE): A Green Grid Data Center Sustainability Metric," Green Grid, Tech. Rep., December 2010.

[27] RealtimeCarbon.org. (2012, Sep.) $CO_2$ conversion factors. [Online]. Available: http://www.realtimecarbon.org/resources/RealtimeCarbonMethodology.pdf

[28] W. Cirne and F. Berman, "A comprehensive model of the supercomputer workload," in *IEEE International Workshop on Workload Characterization, WWC-4. 2001*, dec. 2001, pp. 140–148.

[29] C. Bailey Lee, Y. Schwartzman, J. Hardy, and A. Snavely, "Are user runtime estimates inherently inaccurate?" in *Job Scheduling Strategies for Parallel Processing*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2005, vol. 3277, pp. 253–263.

[30] R. Basmadjian, N. Ali, F. Niedermeier, H. De Meer, and G. Giuliani, "A methodology to predict the power consumption of servers in data centres," in *Proc. of the ACM SIGCOMM 2nd Int'l Conf. on Energy-Efficient Computing and Networking (e-Energy 2011)*. ACM, 2011.

[31] OMNeT++. (2012, Sep.). [Online]. Available: http://www.omnetpp.org

[32] INET Framework. (2012, Sep.). [Online]. Available: http://inet.omnetpp.org/

[33] FIT4Green project. (2012, Sep.) FIT4Green: Energy aware ICT optimization policies. [Online]. Available: http://www.fit4green.eu

[34] V. Georgiadou, A. Salden, C. Dupont, A. Somov, A. Giesler, J. C. L. Egea, P. Barone, G. Giuliani, O. Abdelrahman, R. Lent, M. Kessel, T. Schulze, R. Basmadjian, H. D. Meer, M. Majanen, and O. Mämmelä, "Enhanced control plug-in and control desk software design specifications," FIT4Green, Deliverable D-5.3 v1.0, May 2012, http://www.fit4green.eu/.

[35] FIT4Green plug-in source code. (2012, Sep.). [Online]. Available: https://github.com/fit4green/FIT4Green

[36] Linux/ACPI - Documentation. (2012, Sep.). [Online]. Available: http://acpi.sourceforge.net/documentation/sleep.html

[37] LINPACK. (2012, Sep.). [Online]. Available: http://www.netlib.org/linpack/

[38] PEPC - Pretty Efficient Parallel Coulomb Solver. (2012, Sep.). [Online]. Available: www2.fz-juelich.de/zam/pepc

[39] Unicore Client Layer. (2012, Sep.). [Online]. Available: http://www.unicore.eu/unicore/architecture/client-layer.php