Interference of billing and scheduling strategies for energy and cost savings in modern data centers

Interference of Billing and Scheduling Strategies for Energy and Cost Savings in Modern Data Centers

Julian M. Kunkel[a,*], Hayk Shoukourian[b], Reza Heidari[c], Torsten Wilde[b]

[a] *University of Reading*
Reading, UK
[b] *Leibniz Supercomputing Centre (LRZ) of the Bavarian Academy of Sciences and Humanities*
Boltzmannstraße 1, 85748 Garching bei München, Germany
[c] *German Climate Computing Center (DKRZ)*
Fuhlentwiete 10, 20355 Hamburg, Germany

Abstract

The high energy consumption of HPC systems is an obstacle for ever-growing systems. Unfortunately, energy consumption does not decrease linearly with reduced workload; therefore, energy conservation techniques have been deployed on various levels which steer the overall system. While the overall saving of energy is useful, the price of energy is not necessarily proportional to the consumption. Particularly with renewable energies, there are occasions in which the price is significantly lower. The potential of saving energy costs when using smart contracts with energy providers is lacking research. In this paper, we conduct an analysis of the potential savings when applying cost-aware schedulers to data center workloads while considering power contracts that allow for dynamic (hourly) pricing.

The contributions of this paper are twofold: 1) the theoretic assessment of cost savings; 2) the development of a simulator to replay batch scheduler traces which supports flexible energy cost models and various cost-aware scheduling algorithms. This allows to approximate the energy costs savings of data centers for various scenarios including off-peak and hourly budgeted energy prices as provided by the energy spot market. An evaluation is conducted with four annual job traces from the German Climate Computing Center (DKRZ) and Leibniz Supercomputing Centre (LRZ).

*Corresponding author

The final publication is available at ScienceDirect via the DOI 10.1016/j.suscom.2019.04.003

The theoretic analysis indicates a cost savings for 4-8% when shutting down unused client nodes, and 6-20% with hourly cost models and optimal scheduling. The experimental validation of a practicable scheduler increases the accuracy against the theoretical best case analysis. As expected, a cost-efficient scheduling algorithm that is fed with the information about future energy costs shifts the jobs to the timeslots where the job execution is cheaper and reduces the energy expenditure, yet increases the waiting times of pending jobs. However, the expected savings for this effort are not justifiable compared to the simple strategy of turning off the unused nodes. Additionally, we compare the cost savings to the total costs of ownership showing that smaller systems with on-demand provisioning yield better cost efficiency.

Keywords: HPC, energy-efficiency, cost-aware scheduling, cost savings, spot market, discrete event simulation

---

1. Introduction

The interest in power reduction is manifold. Product manufacturers are interested in extending the lifetime of its battery while simultaneously reducing the device size and extending the set of supported features, making low-power integrated circuit design a key. In environments such as High-Performance Computing (HPC) data centers - with large-scale systems connected to a power cord, the dependence on fossil fuels, production of high carbon footprints, ever-increasing energy-consumption and associated costs lead to the necessity of low-power solutions for all the building blocks of a modern HPC data center, involving electronics and the supporting software. According to a June 2016 report from Lawrence Berkeley National Laboratory [1], the U.S. data centers are estimated to consume around 73 billion kWh in 2020.

This growing power consumption tendency can also be seen in Europe. Figure 1 shows the trend of the energy costs at the Leibnitz Supercomputing Centre (LRZ) from 2000 till 2017 including an estimate for 2018 and for 2019. As can be seen, the energy costs keep rising, reaching almost 7 M€ for 2016. This increase includes: *(i)* the rise in electricity prices from 0.07€ in 2000 to 0.157€ in 2016 for 1 KWh (which, after a drop to 0.145€ in 2017[1],

[1]Explaining the descend in energy costs for 2017 as shown in Figure 1.

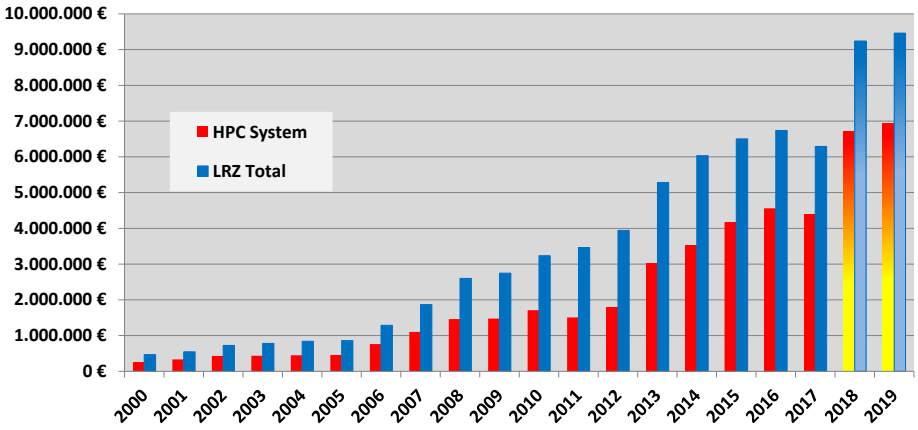


Figure 1: Trend of energy costs for LRZ from 2000 till 2017, and prediction for 2018, 2019

rises again to 0.180€ in 2018), as well as *(ii)* the rise in the power consumption of the deployed HPC systems. Even though new generations of HPC systems provide better power efficiency, i.e., better performance/watt ratio, the density and overall performance of these machines improve substantially and an overall increase in the energy consumption - which is about 42 MWh (of which 29 MWh stem from the HPC systems) for LRZ in 2017.

All these make high-performance design and energy-efficient design in many ways synonymous. Energy-efficiency and cost-efficiency are not the same; energy-efficient data centers may have a reduced scientific output but cost-efficient systems may waste energy while maximizing scientific output. This paper focuses on cost-efficiency, but it links these topics when discussing energy contracts. Utility providers (energy suppliers) regulate the costs depending on demand and supply, and they provide monetary incentive to contracts. This, in turn, stabilizes the overall utilization and optimizes the power grid. This is particular useful as the power consumption of recent HPC systems is highly depending on the executed application mix.

The power consumption of an HPC system at a given point in time is mainly dominated by the power profiles of applications that are utilizing the system at that particular time. Changes in these power profiles can occur in a rate of milliseconds due to, e.g., application termination, start of a power-intensive application-phase, that lead to high fluctuations in overall system power profile.

Figure 2 shows the power consumption profiles of LRZ during a time frame of five days in 2017. The topmost curve, labeled as *P_DC*, shows

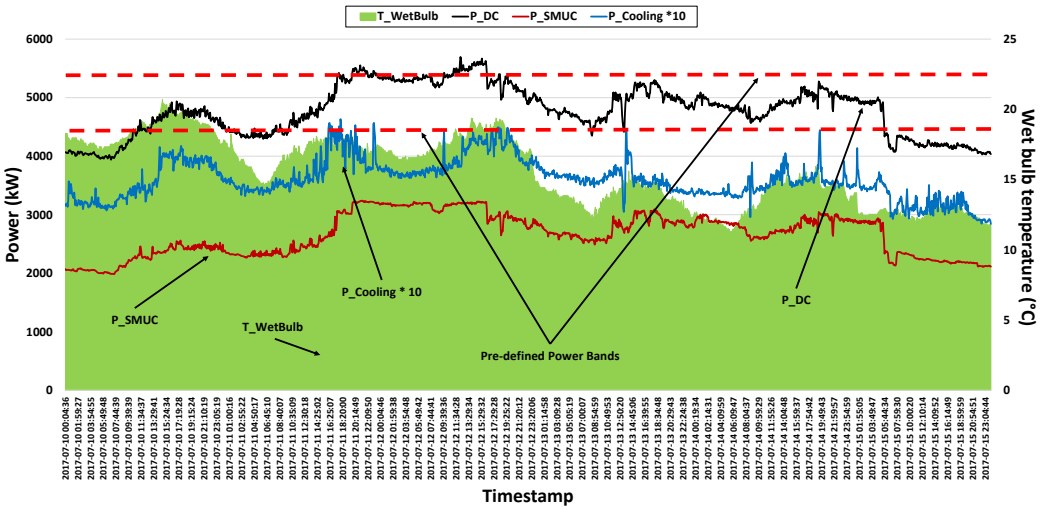


Figure 2: Power consumption profile of LRZ for a five-day interval in July 2017

the total power consumption of the data center; the second curve (counted from the top), labeled as *P_Cooling *10* shows the power consumption of the cooling infrastructure in kW and multiplied by 10[2]; the third curve, labeled as *P_SMUC*, shows the power consumption profile of the deployed HPC system SuperMUC (for both installation phases of the system: Phase 1 and Phase 2). The area marked in green and labeled as *T_WetBulb* shows the wet bulb temperature.

As can be seen, the variability of *P_SMUC* has a major impact on the data center power profile *P_DC*. But *P_DC* is also affected by the power consumption of the cooling infrastructure - i.e., *P_Cooling*10*, thus showing additional peaks and swings.

These fluctuations lead to the following affects: *(i)* the majority of currently existing contractual agreements with the utility providers do include a penalty fee for violating the pre-defined power band boundaries (dashed red lines on *P_DC* power profile), and this fee can propagate to the entire operational year bill, even if the power band was violated for a single time; *(ii)* in an Exascale system, these picks and swings might translate into fluctuations of several MWs that might not only breach the safety limits of the data center's power infrastructure but they also impact on the stability of

[2]Scaled for illustrative purposes.

underlying power grid.

Renewable power generations make the power grid prone to variability. Figure 3 shows the solar power generations in Germany in 2017, depicting the power variability of the grid due to the renewable energies. Thus, the existing Multi-Petascale and future Exascale data centers – as major power consumers – could contribute to the overall stability of the power grid via an intelligent management of several MWs of power in short time.

The impact of demand and supply on fluctating prices can best be seen on the energy spot market. For example, the European Power Exchange (EPEX) spot market provides both the day-ahead and intraday energy trading schemes. In the day-ahead trading scheme, a blind auction is conducted on the blocks of energy (at least 0.1 MW) for every individual hour of the following day and the prices for the following day are fixed at noon. The Physical Electricity Index (Phelix) is computed from the hourly contracts – an example of the price development is shown in Figure 4. The intraday energy trading scheme supports contracts for fine-grained timeslots of 15 minutes, but this paper does not consider them further. Typically, energy is significantly cheaper at base load than peak load; e.g., the energy price list for 2018-02-13 indicates that the electricity costs 10 € per MWh until 6 am and around 40 € per MWh for the rest of the day. Depending on the contract with the utility provider, such fluctuating prices can be exploited to reduce the data centers energy costs.

The **contribution of this paper** is the investigation of energy expenditure when applying various energy cost models and scheduling strategies to

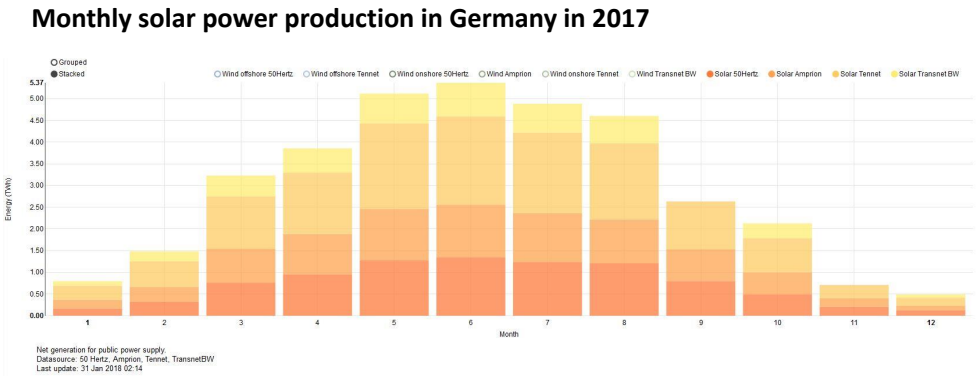


Figure 3: Peak generation of solar energy in Germany in 2017. Source: Fraunhofer Institute for Solar Energy Systems [2]

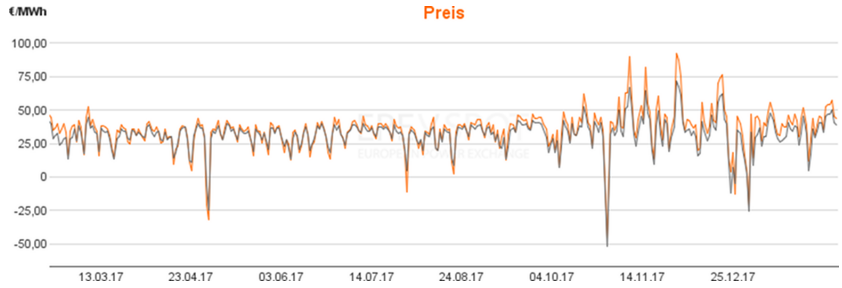


Figure 4: Phelix day-ahead auction price
Source: https://www.epexspot.com/de/marktdaten/dayaheadauktion

data centers. Firstly, a theoretical model is used to estimate the potential gain of a scheduling strategy. Secondly, a discrete event based simulator is developed to replay job traces from data centers and support novel models for costs and scheduling. This allows us to estimate and quantify the benefit of certain strategies implementations on a given production system, and, therewith, guides RD&E efforts to make better use of HPC systems.

The rest of this paper is structured as follows: the state-of-the-art and related work is described in Section 2. In addition, the HPC data centers considered in this study and their energy contracts are described in Section 3. In in Section 4, several alternative cost saving and billing strategies are described and illustrated. Moreover, the theoretical model and the design and implementation of the discrete event simulator is described in Section 5. The simulation results from various schedulers and cost models are, however, presented in Section 6. This includes a discussion and assessment of the Total Cost of Ownership (TCO). Finally, in Section 7, we summarize the paper and describe the relevant actions required to realize the discussed cost savings.

2. Related Work

Related work can be categorized into metrics to quantify and assess energy consumption and costs, optimization techniques for energy efficiency, scheduler strategies to minimize energy costs, and the simulation of HPC workloads.

Energy/Cost metrics. As in [3], a metric is a measure for quantitative assessment of a system and useful in system improvement and performance

comparison. Energy efficiency metrics in HPC systems aim at higher computational output while reducing the energy cost. Table 1 shows a number of important energy consumption and cost metrics.

Energy-efficiency optimization techniques. There have been numerous research on energy efficiency in HPC systems. The work in [10] gives an overview of static and dynamic power management techniques useful to configure the performance level of hardware components. Google [11] adopts a machine learning approach that learns from the actual operational data and predicts PUE. Shoukourian et al. [12] suggests another machine learning approach based on recurrent neural networks for modeling the *Coefficient of Performance* (COP) of a data center's cooling loop and identification of key parameters as well as their optimal configuration for a given point in time.

Metric	Stands for	Description
PUE	Power Usage Effectiveness	Total data center energy divided by total IT energy. PUE=1 shows all power is consumed in IT equipment [4].
ITUE	IT-Power Usage Effectiveness	Total IT energy divided by computational energy [4].
TUE	Total-power Usage Effectiveness	Total data center energy divided by total computational energy[4].
CUE	Carbon Usage Effectiveness	Total Carbon emissions caused by data center energy divided by IT energy [5].
DWPE	Data center Workload Power Efficiency	Energy efficiency of a given workload on a specific HPC system [6].
OPEX	Operating Expenditure	Ongoing cost of running a system [7].
CAPEX	Capital Expenditure	Expenditure on investment in long-lived assets of a system [8].
TOTEX	Total Expenditure	Made of OPEX and CAPEX [8].
TCO	Total Cost of Ownsership	The purchase price plus the operational costs of an asset during its life cycle [9].

Table 1: Energy/Costs Metrics definitions

Scheduler optimizations. In recent years, a number of research attempts has been dedicated to developing energy-aware job schedulers in HPC systems and cloud computing. In [13], an efficient energy-aware resource provisioning and task scheduling algorithm is developed for the cloud environment. The authors in [14] report a meta-scheduler in a cloud environment to dispatch jobs to resources in different geographyical locations while minimizing CO_2 emissions but maximizing the profit and observing the Quality of Service (QoS). An energy-cost-aware scheduler in [15] shows a cost-saving of 25-50% by delaying low-priority workloads when energy prices are high, but ensuring a rapid service to high-priority jobs. In a different approach presented in [16], Mämmelä et al. suggests an energy-aware scheduler that reduces energy consumption by 6%–16% by switching off idle nodes until the next request becomes ready, without affecting turnaround time or waiting time of the jobs.

Roloff et. al in [17] argues that cloud environments can provid a higher performance, up to 27%, and cost efficiency, up to 41%, than traditional HPCs if they fit target applications. In [18], Shoukourian presents an energy-wise optimal resource configuration for a scheduling system to adhere to a predefined power cap and execution time constraint. This approach uses adaptive models for estimating energy, average power, as well as execution time of workloads with respect to different compute resource configurations (e.g., compute node count, CPU frequency, etc.) [19]. The developed models also account for node power variation [20] arisen due to the manufacturing tolerances and variations during the fabrication of the integrated circuits.

In [21], a novel algorithm exploits scheduling methods dynamically for real-time independent tasks in order to increase utilization and reduce energy consumption. A survey in [22] reports various applications of supervised, unsupervised and reinforcement learning on energy-efficient resource management in cloud computing environments, including: adjusting CPU frequency, power-aware server allocation, intelligent scheduling to turn off unused servers, power-aware task scheduling and consolidation, load prediction leading to optimal resource allocation, spatially-aware load placement to reduce cooling costs, workload classification leading to load consolidation and request forecasting to put unused machines to sleep. In [23], machine learning predicts system behavior such as resource usage and task service-level-agreement to help schedulers allocate tasks to hosts striking a balance between revenue for executed tasks, quality of service, and power consumption. T. Miyazaki [24] applies Bayesian optimization (BO) to reduce the occupancy time in a graphics processing unit cluster system earning the sec-

ond place in the Green500 list in June 2017.

Simulation of HPC and Cloud Workloads. It is important to analyze the system response to new scheduling policies before putting them into production. Cheng et al., in a recent work [25], applies reinforcement learning to optimize energy costs for the cloud. They develop sophisticated models for system and workload that apply for the cloud considering, e.g., dynamic energy consumption and realistic price models. Authors in [26] propose a means of reproducibly and accurately determining the true impact of changes in scheduling policy, resource configuration, and workload distribution. They benefit from the Maui Scheduler for simulating and producing statistics to analyze the impact of an immense array of real world system configurations, policy sets, and workloads. Authors in [27] introduce the Cluster Discrete Event Simulator (CDES) framework as a strong candidate for HPC workload simulation. This framework can take system definitions, multi-platform real usage logs and can be interfaced with any scheduling algorithm. The paper states a 95% accuracy against a production level HPC system.

Our work differs from previous works as we incorporate hourly energy cost models and investigate alternative scheduling strategies and policies for reducing the energy-costs at the scheduler level. While the work from cloud providers shares similarities, it differs by the applied workload, HPC workloads run tightly coupled on multiple nodes but are loosely dependent on other jobs, cloud workloads can be modeled as DAGs of tasks run on individual servers. We explore the energy and cost savings of alternative strategies – theoretically and practically for two data centers running HPC workloads.

3. Site Descriptions

The following two HPC systems are considered in this article.

3.1. Mistral

Mistral is the high-performance computer of the German Climate Computing Center (DKRZ). It was procured in two phases; in Phase 1, 1529 Intel Haswell E5-2680 nodes with two processors each were installed. Additionally, several handful of additional nodes serve as login and pre/post-processing nodes, about 20 nodes are equipped with GPUs. In Phase 2, additional 1750 nodes with Intel Broadwell CPU were installed, these Phase 2 nodes are not considered in this paper. The nodes are interconnected with an Infiniband

FDR fat-tree topology using a 4:1 blocking factor, i.e. the bisection bandwidth of the overall system is limited to 1/4 of the 6000 MiB/s throughput provided by each link. DKRZ hosts one of the biggest storage systems with about 53 Petabyte of storage in two file systems and a tape complex providing 75,000 slots.

15% of the compute nodes are air cooled mostly with water-cooled rack doors. Other nodes are hot water cooled with a maximum inlet temperature of 38°C[3]. In the main cooling circuit, water is pumped up to the roof utilizing free cooling; this is sufficient in Hamburg for nearly all days of the year. A rack has its own water circuit internally driven by two redundant pumps and is connected via a heat exchanger to the main circuit. In this setting, the directly liquid cooled nodes of the supercomputer itself achieved in 2016 a PUE of 1.03 (the average PUE across the whole infrastructure is 1.19).

Mistral compute and storage for both procurement phases consume in average about 1.15 MW. The storage and network accounts for 250 KW. This includes three Mellanox switches backing the fat-tree network (about 30 kW). The water pumps account for 5 to 10 kW and the dry coolers on the roof between 3 kW (winter) and 10 kW (summer). We assume the first and second phase consume each half of this power.

3.2. SuperMUC

SuperMUC, the flagship system of Leibniz Supercomputing Center (LRZ) [28], consists of two installation phases with an aggregated peak performance of 6.8 PetaFLOPS ($= 6.8 \cdot 10^{15}$ Floating Point Operations Per Second) [29] having a total of 3.4 MW average power consumption [30]. SuperMUC is a GCS (Gauss Centre for Supercomputing) [31] Tier-1 system and one of the PRACE (Partnership for Advanced Computing in Europe) [32] Tier-0 systems. It is the first High-Temperature (ASHRAE W4 chiller-less [33]) Direct Liquid Cooled (HT-DLC) system installed worldwide [34]. Its active components – e.g., processors, memory – are directly liquid cooled with an inlet water temperature of up to 45 °C.

SuperMUC Phase 1 is an IBM [35] iDataPlex DX360M4 Intel Sandy Bridge system, installed in 2012, having 3.18 PetaFLOPS [29] theoretical peak performance and an energy-efficiency rate of 0.85 GFLOPS/W [36].

[3]The temperature is regulated depending on the outside conditions between 20 and 40°C, for safety reasons, DKRZ choose 38°C.

Phase 1 consists of 18 thin node islands, where the compute nodes within an individual island are connected via a fully non-blocking Infiniband network (FDR10). These 18 thin node islands contain 9216 compute nodes, where each node is equipped with two 8 core Intel Sandy Bridge-EP Xeon E5-2680 8C [37] processors.

In SuperMUC Phase 2, additional 3072 nodes were installed in 2015. Due to a similar nature of the simulation presented later and for the provision of an in-depth analysis, this paper considers only the data related to the Phase 1 application queue. Both phases of SuperMUC have a high speed interconnect between the islands, which enables a bi-directional bi-section bandwidth ratio of 4:1 (intra-island/inter-island). Phase 1 and Phase 2 use the IBM LoadLeveler [38] as a resource management and scheduling system.

3.3. Energy Pricing Policy at DKRZ

DKRZ bought electricity from the stock market based on a fixed energy consumption and expenditure policy in 2016 and 2017. This envelop enforced the power consumption to stay constant within a small variation margin and, under this condition, the energy cost was fixed for the whole year. In retrospect, however, the average energy price can be calculated as the total energy expenditure divided by the total consumed electricity, which is as follows: Average Energy Price (2016) = 14,50 ¢/kWh. The aggregated energy cost would result from some components mainly energy cost (12%), grid cost (2%), green energy (50%), and taxation (15%).

3.4. Energy Pricing Policy at LRZ

LRZ has a power band contract, meaning, it can consume a variable amount of power in a specified range without any penalties (leading to a variable amount of energy consumed during the year). Therefore, the final energy price depends on the total consumed energy. LRZ purchases a basic power band for one or multiple years at the European power stock exchange. Hence, the power price is determined by the European stock market. The price per kWh can vary slightly depending on the consumed energy (more energy consumption can lead to a lower price per kWh). The contract has two major cost items:

(a) connection fee – charged by the owner of the physical power lines, fixed charges per kWh. A significant fraction of the fee can be saved if power peaks are minimized;

(b) energy consumption for complete year - charged by the power provider, price of used energy, depends on final (yearly) energy consumption (kWh) and maximum power usage.

If the energy consumption exceeds 10 GWh per year, special parts of the supply contract can be applied. For example, if the fraction of the total energy consumption per year to the maximum power consumption per year exceeds 7000 h, the amount of the connection fee (a) can be reduced. More details concerning the LRZ power contract and possible TCO optimization can be found in [39]. At LRZ a huge cost factor to OPEX is the energy consumption. LRZ uses energy-aware scheduling to improve the energy efficiency [40].

4. Cost Saving Strategies

The primary goal of this paper is to identify possible ways of maximizing the scientific output while minimizing the cost expenditure. This section discusses different strategies for achieving this goal.

A traditional resource management and job scheduling system optimizes two goals: utilization of the data center and user waiting time, defined as a time difference between job completion and job submission, i.e., $t_{completed} - t_{submit}$. During recent years various **power capping** techniques have been implemented to ensure to keep the system usage within a given power band and prevent possible power overloads [41, 18]. Similarly, optimization of Energy-to-Solution [42] and the problem of **energy capping** has been investigated [20].

With respect to power, aspects involving the power grid are relevant as theoretically energy consumption can be adjusted rather quickly in a data center. This can increase the stability of supporting power grid for green energy sources balancing energy or minute reserves. Besides scheduling of applications, an adaption of the CPU frequency per application offers a quick adjustment of used energy. When needed, applications can be run at higher frequency which consumes more power and leads to optimal runtime, or at lower frequency to consume less power.

An additional cost saving factor is that heat created by the data center could be re-used, for example, via the help of adsorption chillers, this is not further discussed in this paper, though. Let us look at possible strategies in more detail.

4.1. Alternatives to Procurement and Hardware Control

The list below outlines the possible set of cost-saving strategies and actions when system underprovisioning is possible, i.e., providing less system capacity than needed for minimizing the cost expenditure. The degree of underprovisioning is a decision to be made at design time or when additionally deploying new systems (or system components) based on the estimation of utilization spikes and the penalty costs with regard to the risks analysis of not adhering to certain service-level agreements.

1. *Procure a system for normal operational scenarios*
   This strategy suggests to design/deploy a system for meeting mean utilization requirements. For data centers that exclusive jobs occupied in average only 80% of the cluster during their lifetime, it might be possible to procure, for example, 85% of the hardware. However, this will increase the waiting time during spikes of user requests. Segments dedicated to cloud computing can be used for covering the rest, out-of-band, requirements. This will ensure the homogeneity of waiting time for users. If the waiting time increases too much, ad-hoc solutions, like purchasing, for instance, 1% of additional hardware can be considered.

2. *Incremental provisioning*
   Typically, after the inauguration of a new cluster, it is not utilized completely but the demand for resources grows slowly. Similarly to the previous strategy, we may start with a smaller cluster and incrementally grow the size depending on the user needs. This is more a theoretical scenario as the incremental provisioning requires policies for acceptance and dynamic integration of new resources into the existing resources. This elasticity is a strength of the cloud.

3. *Send client nodes to sleep mode (turn power off)*
   Power can be saved by turning off unneeded resources as nowadays supported by many resource management systems. This however, adds certain latency and imply additional challenges: Usually, bringing up a node after deep sleep mode can be challenging since network protocol synchronisation can be adversely affected by constant drop-off of nodes in the network. Additionally, if several compute nodes were powered-off within the same time window then, in some cases, this could potentially lock-up the corresponding file system like GPFS for other users while

the recovery processes take place[4]. This could cause failures of other jobs on different compute nodes as they exceeded their own timeouts for I/O completion [43]. At future multi-Peta/Exascale level this might also cause thrashing due to the irregular node booting failures when significant amount of compute nodes are being provisioned. A potential solution to reduce waiting time, issues, and booting is suspend to RAM that is nowadays common in desktop systems. Still, this strategy is rarely used in data centers.

4. *Adjusting the processor frequency*
   The data center utilization rate can be increased by reducing the operating frequency of the processors, for example, to the lowest possible value (typically, 1.2 GHz). This will also allow to mitigate some of the possible power spikes caused by power-hungry applications. This could be applied automatically, for example, when a scheduler observes few pending jobs.

5. *Support power grid stability*
   Last, we can adjust the power consumption of a data center depending on the power availability by utilizing the above mentioned actions/strategies. Currently, power can be varied by an estimated 50% (maximum CPU frequency vs. lowest CPU frequency). For an Exascale data center with a 35 MW average power consumption, this variability can be estimated as high as 25 MW at peak and 18 MW at average power consumption. Since power transitions can be nearly instantaneous, Exascale data centers are one of the few large power consumers with flexibility in their power profile.

4.2. Data center billing strategies

This subsection provides an overview on various billing strategies provided by local utility providers (e.g. Stadtwerke München GmbH [44], E.ON SE [45], etc.) and/or companies operating national grid (e.g. TenneT [46]) as well as outlines their exploitation range.

1. *Fixed cost per kWh (24h flat rate)*
   This policy indicates that data centers may pay a single unit rate for

[4] A GPFS (General Parallel File System) filesystem would need to start a recovery process if a compute node was not shutdown properly

electricity independently from the time of a day, power consumption, or any other constraint. There is no particular optimization available to reduce costs with this default mode.

2. *Base/Peak rate*
   These types of policies indicate that data centers get a reduction in electricity price for certain time slots within a day with low load, i.e., depending on a given electricity rate, which can be a day rate (e.g., ranging from 08:00am - 1:00am) or else a night rate[5]).

   The resource scheduling and management system can take advantage of this electricity rate policies by shifting power-hungry (low priority) workload to off-peak hours potentially even shutting down unneeded resources. Thus scheduling only high priority/critical workload (desirable with low power/energy consumption rate) during the peak hours and deferring low priority workload (having high energy consumption rate) to off-peak hours when the cost of electrical power is cheaper.

3. *Stock market*
   Instead of relying on fixed contractual terms, data centers may purchase power directly from the stock market (refer to the introduction for a description). While this strategy itself may be cheaper than a fixed unit rate, it offers more room for optimization than the base/peak model. Typically, the energy costs of the following 24 hours (they vary for each individual hour) are announced a day in advance allowing to delay jobs to cheaper periods of time, or to control the CPU frequency.

4. *Stabilizing the grid with green energy lookahead and weather dependency*
   This variant does not aim to necessarily optimize costs but stabilize the power grid – this may lead to a cost deduction as well.

   The renewable energy sources like wind, solar, and hydropower are slowly phasing out fossil fuels (oil, coal, and natural gas) and further transforming electricity by making its greenness variable throughout the day, where the amount of green power is increased with more windy and sunny weather. In order to support for a renewable energy source based operation of the target system, the dynamic of the available power must be taken into account, since in the considered case the

[5]During the summer time, this ranges are typically shifted by an hour.

available portion of the power will change with the time. The weather forecasting tools, that help in identifying greenest and dirtiest times of day, can further assist in achieving this goal.

Besides these general billing strategies, there are orthogonal options available for each of them:

1. *Power band*
   These types of policies usually indicate that a data center can consume a variable amount of power in a specified range without any penalties leading to a variable amount of energy consumed during the year. The vast majority of these contracts includes a penalty fee for violating the pre-defined power band boundaries, which can propagate to the entire operational year bill, even if this predefined power band is violated only a single time [47].

2. *Fixed energy budget*
   Some contractual agreements may include a fixed energy budget for the entire system life-time[6] which shifts the risk of energy costs from the data centers to the vendor. These contracts are usually negotiated with system vendor or with the corresponding power company. The challenge here is to maximize the scientific output within the given budget or alternatively pay the extra energy bill themselves.

Additionally, the majority of HPC users/clients are currently being charged according to the time usage of certain sets of compute resources. This policy can be modified to include costs or energy-driven charging policies by introducing energy budgets for users. This budget can be tracked via the help of a well-established in community metric: Energy-to-Solution [42]. Cost-to-solution extends the notion by accounting for the total expenditure.

4.3. Scheduling Strategies

The items below outline some scheduling strategies and their variations with regard to energy/power consumption awareness.

[6] Other timely limited variations of this type of contract are possible.

1. *First Come, First Served (FCFS)*
   First Come First Serve, also known as First In, First Out (FIFO) algorithm simply queues and dispatches application for execution in the order that they arrive in the queue.

2. *Job Delaying*
   This strategy ensures to delay certain types of jobs (for example power-hungry ones) for certain time period (for example, off-peak hours).

3. *Job Delaying With Node Shutdown*
   This strategy ensures to delay certain types of jobs and to power-off certain compute nodes for minimizing the cost expenditure during certain time frames.

4. *Green Job Advancing/Forwarding*
   This strategy ensures to advance certain types of jobs (for example power-hungry ones) when the amount of green power is maximized (see Subsection 4.2).

5. *Minimize Power Peaks*
   This strategy ensures to schedule jobs in a way that system power peaks are minimized. This will allow to adhere to predefined power bands (see Subsection 4.2). Additionally, this strategy can help to achieve a flat energy profile is achieved, which in turn can lead to cost-saving bonuses granted by the power utility provider. DVFS-like techniques [40] can further assist in achieving this goal.

6. *Priority Based System Utilization*
   Assume that the resource management system supports for a priority based scheduling of various applications with different power consumption profiles and the data center infrastructure is designed for providing the power capacity for near peak usage for high priority applications. Assume further that these near peak operating applications are served only for a fraction of a day, leaving the dedicated power capacity underutilized for the rest of the time. During these time fractions, the low priority applications can be executed for increasing the mentioned utilization rate.

We modeled and analyzed a subset of the aforementioned strategies. Note that there exist many alternative scheduling algorithms particularly targeting

time sharing or cloud workloads that are not suitable for our scenario with tightly coupled HPC workloads.

5. Design

This section focuses on the design of the discrete event simulator but also includes a theoretical model for optimal energy saving. The simulator is a discrete event simulator build with Python3[7], it reads traces of historic job execution generated by schedulers like SLURM. It supports alternative scheduling and energy policies. The simulator is started via selection of a system model and a trace file to execute; it then generates a file with the recorded events that can be used for inspecting utilization and waiting times.

5.1. System Model

The system model for the simulator and the theoretical model is based on the following characteristics:

- Nodes: number of nodes provided by the system.
- CPUs per node: number of microprocessors per node.
- Idle node power: fixed energy consumption of a node in Watts when the node is idle.
- CPU power: typical energy consumption per CPU when utilized. Note that it is replaced with actual measured energy consumption for LRZ.
- Infrastructure power: energy consumption of generic infrastructure components like network switches, storage, etc. that are shared among all nodes and added to the per node idle power.
- System costs: this covers the one time investment for the machine procurement and potential infrastructure changes.
- Annual costs: the estimated annual costs of managing the supercomputer. This covers OPEX, i.e., running costs of the data center.

Based on this model, the TCO of a system over the lifetime is:

$$TCO = system\ costs + annual\ costs \cdot lifetime + energy\ costs \quad (1)$$

[7]It is available at: https://github.com/JulianKunkel/schedsim

5.2. Energy Models

Provided energy models are:

- Fixed-price: a configurable fixed price per kWh is assumed.
- Base-peak: two periods can be defined each with its own energy costs, e.g., between 6 am and 10 pm one cost otherwise another.
- Hourly costs: for each hour and each day the energy costs can be defined and read from a trace file. Since the stock market price is provided in hourly granularity, this allows a direct mapping.

5.3. Scheduling Algorithms

As the main focus of this paper is the analysis of energy and total costs, we provide basic scheduling strategies that enable a comparison. Firstly, for comparison purpose, we use a theoretical approach.

Optimal. This is a mathematical approximation of the best case where we know the costs of the whole year, we fill the machine completely during the cheapest hour until the whole workload is done. The base system either remains idle for the more expensive periods (implying costs for the infrastructure power), or it is completely shutdown. An effective mean cost is computed when using the cheapest hourly costs during the runtime (see Listing 1). This algorithm is an oracle, as it requires to know the future price of energy and all jobs. It also implies no shutdown and restart delay and costs, hence it is an upper bound for any cost savings.

Listing 1: Optimal (pseudo) algorithm

```
# Input:
# utilization of BackfillShutdownDelay
# endtime of BackfillShutdownDelay

costs = list of all hourly cost data until endtime
costs.sort()

hoursUtilized = costs.len() * utilization
effectiveCostsUtilized = first hoursUtilized elements from costs
effectiveCostsIdle = remaining elements from costs
return mean(effectiveCostsUtilized), mean(effectiveCostsIdle)
```

The scheduler is responsible to select the next jobs for execution from the job submission queue. The following basic scheduling strategies are implemented in the simulator:

- FIFO: the traditional first-come first serve (FCFS) strategy.
- Backfill: FIFO with backfilling. Either the first job is dispatched or it iterates over the next 1000 pending jobs and dispatches the next job that meets the criteria: it fits in terms of requested nodes and does not delay the execution of the first job.
- BackfillDelay: this is a variant of FIFO with backfilling but much easier to compute. The condition to check for a delay of the first job is not checked, thus, this strategy may lead to a delay of the next job. The algorithm is illustrated in Listing 2.
- BiggestFirstBackfill: schedule the pending job which requests the most number of nodes first. This is a heuristic.
- LongestFirstBackfill: schedule the pending job with the longest execution time first. This is a heuristic.
- BackfillShutdown*: these are two variants of Backfill and BackfillDelay that shutdown idle nodes and restart them if needed. Shutdown and restart are optimistically performed immediately, in practice, this would take a couple of seconds (with hibernation).

Listing 2: BackfillDelay scheduling algorithm

```
1 checkedJobs = 0
2 for job in pendingList:
3   if job.nodes < availableNodes:
4     checkedJobs++
5     if checkedJobs > backfillLength:
6       return
7     continue
8   dispatchJob(job)
```

The next scheduling strategies aim to optimize the schedule by considering the future energy costs. All of them shutdown idle nodes identical to the BackfillShutdown algorithm. They assume the energy costs are specified in a granularity of an hour and the number of hours they explore is specified by the look-ahead argument (e.g., 12 hours).

- EnforceCheapEnergy: This FIFO scheduler may delay the execution of a job. It computes the energy costs for running the job (according to its actual runtime) in various intervals in its look-ahead window chooses the cheapest execution window. Note that in practice one would not exactly know the future runtime of a job but would use the requested wallclock time.

- PriceAware: It computes the energy costs for running the job in various intervals in its look-ahead window but also considers the energy costs for leaving infrastructure idle. The algorithm computes when the first pending job should be run once, then suspends itself until the time is met (Line 33; not accepting further wake ups in between) to prevent that job execution is moved again. The algorithm is illustrated in Listing 3. Note that the boundary cases for the first/last interval are not completely shown.

- PriceAwareDeadline: Variant of the PriceAware algorithm, it enforces the dispatching of a job if the waiting time is already bigger than the look-ahead window. In contrast, the algorithm now recomputes when the first pending job is computed every time it is called (Line 33 is changed; new jobs cause a reschedule), but also changes Line 6 by adding: $or(time - job.submissionTime) < lookAhead$. Thus, jobs are not delayed beyond the look-ahead window after their submission time.

6. Evaluation

6.1. Configuration

The used characteristics according to our system model are shown in Table 2. For DKRZ, The energy consumption per node has been measured using HDEEM [48] on an idle node (70 W) and a node running the COSMO-ART [49] model at 2.5 GHz on both microprocessors (260 W/Node). For the infrastructure, the overall power consumption of storage, network, and cooling infrastructure (260 kW, see Section 3, cooling is 10 kW, therefore, we included it) is equally accounted to both phases (but we investigate traces of Phase 1 system only).

In case of LRZ, the infrastructure power (which as was mentioned above covers supportive energy costs such as network switches and storage, that are

Listing 3: PriceAware scheduling algorithm

```
# time is the current simulation time
for job in pendingList:
  if job.nodes > availableNodes:
    return
  # Schedule jobs that do not fit into look ahead immediately
  if job.runtimeHours + 2 >= lookAhead:
    dispatchJob(job)
    return
  # Energy consumption when we leave the nodes empty
  nodesIdleP = idleNodePowerConsumption * job.nodes
  jobConsumption = job.consumption + nodesIdleP
  # Compute price for dispatching now
  costsToRun = 0
  for jobHour in 0, job.runtimeHours:
    costsToRun += energyCost(time + jobHour) *
                  jobConsumption * job.durationThisHour
  cheapestPrice = costsToRun
  cheapestTime = "now"
  # Now what happens when we start at the beginning of
  # each future hour in our look ahead
  idleCostsSoFar = energyCost(time) * secondsThisHourRemain
                  * nodesIdleP
  for hour in 1, lookahead:
    if idleCostsSoFar > cheapestPrice:
      # already the idle costs are higher, we can stop searching
      break
    costsNow = energyCost(time + hour)
    costsToRun = idleCostsSoFar
    for jobHour in 0, job.runtimeHours:
      costsToRun += energyCost(time + hour + jobHour) *
                    jobConsumption * job.durationThisHour
    if price < cheapestPrice:
      updateCheapestPrice/Time
    idleCostsSoFar += energyCost(time + hour) * nodesIdleP
  # Special case for last hour (not shown)
  if cheapestTime != "now":
    suspend scheduler until (cheapestTime)
  dispatchJob(job, cheapestTime)
```

Variable	Mistral Phase 1	SuperMUC Phase 1
Number of Nodes	1529	9216
Idle node power	70 W	49 W
CPUs per Node	2	2
CPU power	95 W	Trace APC
Infrastructure power	130 kW	176.5 kW
Costs system	16.5 M€	83 M€
Annual costs	4 M€	16 M€

Table 2: Configuration of the system characteristics

always operational) was obtained via the help of intelligent Power Distribution Units (PDUs), whereas the idle node power and application traces rely on paddle card readings [18] that are obtained through PowerDAM monitoring toolkit [50]. The mean power of the jobs used by the traces is the *Average Power Consumption (APC)* as reported by the measurements. The presented energy costs for LRZ cover storage and network but not the costs for cooling.

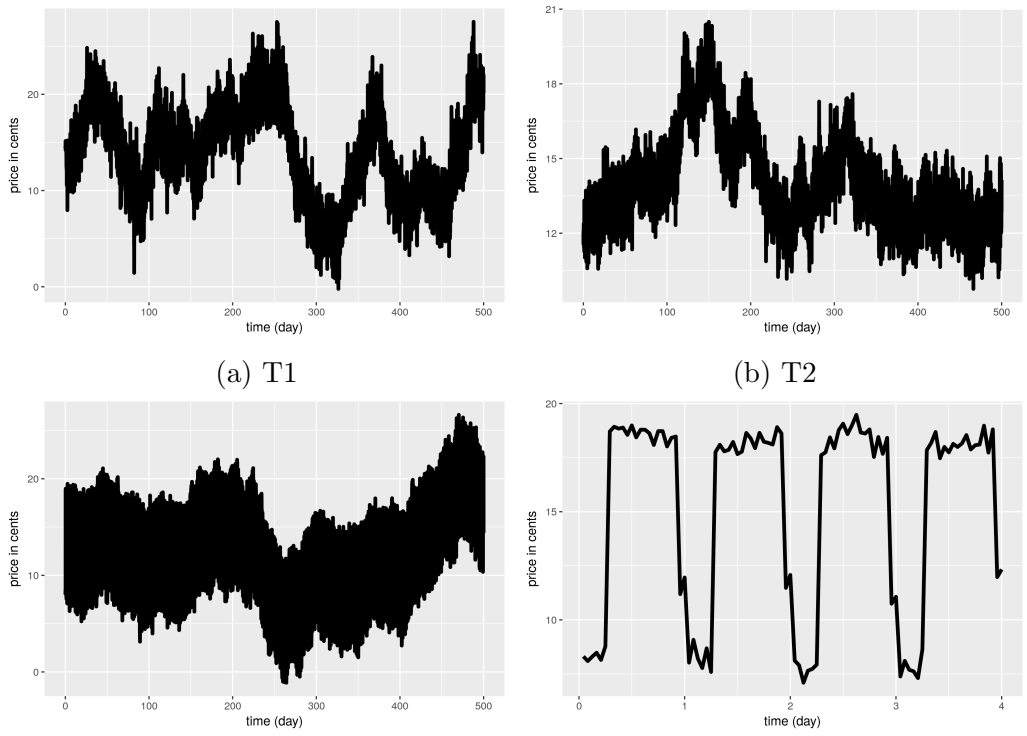
Note that neither the cent exact price is not necessary for the analysis conducted in this paper nor can it be determined with the data available from the center monitoring systems. The annual OPEX cover employees and are estimates based on the number of employees (75 for DKRZ and 150 for LRZ). Costs of the systems are according to press releases, however, as we simulate only DKRZ's Phase 1 system, we budget for the half annual and system costs.

6.2. Energy-Cost Models

The fixed-price model uses 14.5 ¢ per kWh as a price for both systems. We adjusted the mean energy costs of all other models to result in the same price as this makes it easier to compare all models – we can assume the resulting energy cost is similar for all models.

For the base/peak load model, the peak time is between 6 am and 10 pm and the price is 16.675 ¢ and 10 ¢ per kWh for peak and base load, respectively. The hourly cost model is more complex and as follows.

Hourly Energy Costs. Using actual data from Phelix for this cost model is not useful. Firstly, the energy provider will not give the auction price to the customer but has to add further costs to it like taxes; secondly, a comparison



(a) T1 (b) T2

(c) T3: all 500 days (d) T3: first 96 hours

Figure 5: Hourly energy costs simulated

with the other price models would be difficult as the price level differs significantly. Therefore, three timelines (T1-T3) have been synthetically created for 500 days (some scheduling models run longer than 370 days) using autoregression (see the Equation 2 and Equation 3 below)[8]. The first 24 hours are created using a random number, then an autoregressive model is used to compute the next values. After the initial series is created, its mean value is adjusted to 14.45 ¢ per kWh. Visualizations of the created timeseries are shown in Figure 5.

As can be seen from Figure 5, the dynamics and overall behavior of the three timelines differ; T2 is more static and has a smaller dynamic range than T1. T3 is created with the same rules as T2 but adds after the creation on each day a cost offset to mimic the base and peak prices; between 6 and 22

[8]The data is available in https://github.com/JulianKunkel/schedsim-data

by 10 ¢ and between 10 and 11pm by 3 ¢. It covers the local price difference of the day/night cycle but also inherits the dynamics from T2.

$$T1[t] = 0.25 \cdot sin(t) + 0.8 \cdot Y[t-1] + 0.2 \cdot Y[t-24] + rnorm \quad (2)$$

$$T2[t] = 0.4 \cdot Y[t-1] + 0.4 \cdot Y[t-2] + 0.2 \cdot Y[t-24] + 0.5 \cdot rnorm \quad (3)$$

6.3. Used Traces

The traces used for the following analysis are:

- **DKRZ 15**: this trace starts at 2015-08-15 of DKRZ's Phase 1 system.
- **DKRZ 17**: This trace starts at 2016-12-30.
- **LRZ 14**: This trace starts at 2014-01-01 of SuperMUC's Phase 1.
- **LRZ 17**: This trace starts at 2017-01-01 of SuperMUC's Phase 1.

Each trace covers a one year period of job execution. The trace file contains the submission time as we are interested to investigate waiting times. In the LRZ 17 trace, there are few jobs that are started at the beginning of Dec. 2016, this leads virtually to a longer period (391 days instead of 365)[9].

6.4. Understanding Energy Saving Potential

Firstly, Table 3 shows for each trace general characteristics: the average system utilization, the runtime of the trace (roughly a year), then the mean price for the base-peak and hourly model, and, finally, the implied energy costs. Note that downtime of nodes or the whole systems is not considered in the current execution model. As a data center needs maintenance for a few days (at least) a year, this impacts pending jobs (they are stalled) and delays submission of new jobs. Additionally, the priority of big jobs can cause idle time on nodes that are already blocked for the future execution of a big job. Therefore, the utilization can never reach 100%.

[9] Authors believe that the error induced by simulating from the first submission day is acceptable.

The mean price is simply the mean cost over the runtime of the trace. Runtime and utilization is computed using the FIFOShutdownDelay scheduler[10]. As the power models are generated to have a mean of 0.145 ¢ per kWh, the listed mean price is close to that value. T3 is a bit lower, the reason is that the price rises after the first year of the trace (see Figure 5c). Also, interestingly T3 contains a short period of time where the energy costs are negative. This actually happens in practice when there is an excess in energy.

The total energy consumption for the different trace is distributed across the subcomponents in Table 4. The table shows the total consumption of the infrastructure, the base costs for all nodes (empty nodes), and the additional job energy consumption. The job costs for LRZ is actually computed from the trace using the average power consumption as observed while for DKRZ the approximative model is used.

Next, we explore the potential benefit in terms of actual energy and cost savings for the different traces.

Energy savings when turning off the clients. Firstly, we look into the energy savings when turning off client nodes when they are unneeded. This can be easily computed with the utilization of the system and the infrastructure energy consumption. Again we use the average costs for a trace. Table 5 shows the relevant data: the energy consumption of the trace when leaving the system always on and when shutting down unneeded nodes. The shutdown of unneeded nodes is expected to make up for

$$(1 - \text{Utilization}) \cdot \text{IdleNodesEnergy} \cdot \text{MeanPrice} \tag{4}$$

The difference is the saved energy for shutting down client nodes is in the order of 5-10% of the overall energy consumption. This accumulates to 120-

[10]It will be shown that these values arevery close to the optimal runtime and energy consumption values.

Trace	Utilization	Runtime in days	Energy in MWh	MeanPrice per kWh in €				Energy costs in k€			
				BasePeak	T1	T2	T3	BasePeak	T1	T2	T3
LRZ 14	0.786	367	12,941	0.145	0.145	0.145	0.131	1,870	1,876	1,874	1,699
LRZ 17	0.738	391	14,189	0.145	0.146	0.144	0.131	2,050	2,064	2,039	1,857
DKRZ 15	0.744	367	3,986	0.145	0.145	0.145	0.131	576	578	577	523
DKRZ 17	0.832	367	4,214	0.145	0.145	0.145	0.131	609	611	610	553

Table 3: Trace characteristics and base costs for energy

Trace	Total in MWh	Jobs in MWh	Infrastructure in MWh	Node baseload in MWh
LRZ 14	12,941.19	7,411	1,554	3,976
LRZ 17	14,188.52	8,290	1,658	4,241
DKRZ 15	3,986.09	1,901	1,143	941
DKRZ 17	4,214.27	2,128	1,144	942

Table 4: Energy characteristics for the traces

Trace	Energy in MWh		Shutdown	Shutdown savings in k€				Saving
	Always on	Shutdown	Saved MWh	BaseP.	T1	T2	T3	in %
LRZ 14	12,941	12,090	851	123	123	123	112	6.6
LRZ 17	14,189	13,077	1,111	161	162	160	145	7.8
DKRZ 15	3,986	3,745	241	35	35	35	32	6.0
DKRZ 17	4,214	4,056	158	23	23	23	21	3.8

Table 5: Saving potential for the systems shutting down client nodes

150 k€ for LRZ and across the different price models and to 20-30 k€ for DKRZ which is significant. Note that the relative percentual benefit of saving when shutting down the clients stays the same independent of the power price model used as we use the average energy costs for each trace as this is an approximative model.

Infrastructure power. Next, we explore the impact of the infrastructure power consumption to the costs. Table 6 gives an overview of the saving potential when turning off the infrastructure when the nodes are idle. The saving potential in % is given relative to the absolute costs.

Optimal job scheduling. Finally, we explore the theoretic influence of the optimal scheduling algorithm. Therefore, we compute with the *optimal algorithm* the best and worst prices Table 7. In a nutshell, for the best price, we uses the hour with the cheapest energy costs first (with 100% utilization) until all computation is done. The mean for the remaining time intervals with higher cost is the worst price that we have to pay to upkeep the infrastructure. Note that the optimal oracle primarily serves to understand the

Trace	Energy in MWh		Shutdown	Shutdown savings in k€				Saving
	Always on	Shutdown	Saved MWh	BaseP.	T1	T2	T3	in %
LRZ 14	12,090	11,758	333	48.1	48.2	48.2	43.7	2.6
LRZ 17	13,077	12,643	434	62.8	63.2	62.4	56.8	3.1
DKRZ 15	3,745	3,452	293	42.3	42.4	42.4	38.4	7.3
DKRZ 17	4,056	3,864	192	27.8	27.9	27.8	25.2	4.6

Table 6: Extra saving potential turning off infrastructure (during client idle time)

Trace	Optimal price in €				Worst price in €			
	BasePeak	T1	T2	T3	BasePeak	T1	T2	T3
LRZ 14	0.138	0.129	0.137	0.115	0.167	0.206	0.173	0.190
LRZ 17	0.137	0.126	0.135	0.111	0.167	0.201	0.169	0.186
DKRZ 15	0.137	0.125	0.136	0.112	0.167	0.202	0.171	0.188
DKRZ 17	0.140	0.132	0.139	0.119	0.167	0.210	0.176	0.193

Table 7: Average optimal and worst price depending on the cost model and trace utilization

Trace	Client shutdown only				Client + infrastr. shutdown			
	BasePeak	T1	T2	T3	BasePeak	T1	T2	T3
LRZ 14	187	297	204	281	242	366	262	345
LRZ 17	251	388	262	369	323	475	336	450
DKRZ 15	55	86	58	82	104	145	108	137
DKRZ 17	36	61	41	57	68	102	75	94

Table 8: Cost savings in k€ with the optimal scheduling for the variable price models shutting down client nodes (and infrastructure) compared to Table 3

upper bound of any smart scheduling. As we can see, if we the optimal price is about 1 ¢ lower for T2 and about 2 ¢ for BasePeak, T1 and T3 – which is about 10-20% of the energy costs.

Now, we apply our optimal algorithm oracle, we obtain an extra saving that depends on the utilization, cost model, and energy consumption. Therefore, we consider the scenarios: turning-off-client nodes and turning of infrastructure, too. When we turn off the infrastructure as well, we yield the optimal price for the workload execution, it is computed as follows:

$$\begin{aligned} optPrice =& Utilization \cdot (InfrastructureEnergy + IdleNodeEnergy) && (5)\\ & \cdot BestPrice + JobEnergy && (6) \end{aligned}$$

Hence with infrastructure, we obtain

$$optPrice + (1 - Utilization) \cdot InfrastructureEnergy \cdot WorstPrice \quad (7)$$

Otherwise, we achieve the best price during the workload execution but have additionally to pay the worst price for the idling machine. Table 8 shows the cost saving potential for the price-aware strategy compared to our original costs.

Discussion of the theoretical benefit. An overview of the percentual saving of all aforementioned theoretical models compared to our base costs is given in Table 9. The table shows the percentual benefit compared to our baseline with average energy costs for the traces (no optimization), the benefit

Trace	Shutd. Clients	Client shutd. + opt. schedule BaseP.	T1	T2	T3	Shutd. Cl.+Inf.	Client/infr. shutd. + opt. schedule BaseP.	T1	T2	T3
LRZ 14	6.6	10.0	15.8	10.9	16.6	9.1	13.0	19.5	14.0	20.3
LRZ 17	7.8	12.2	18.8	12.9	19.9	10.9	15.8	23.0	16.5	24.2
DKRZ 15	6.0	9.5	14.9	10.0	15.7	13.4	18.0	25.2	18.7	26.2
DKRZ 17	3.8	5.9	10.1	6.7	10.3	8.3	11.2	16.7	12.3	17.0

Table 9: Cost savings in k€ for the various models in percent compared to Table 3

for shutting down clients and infrastructure but with the default schedule (average costs), and finally the application of the optimal algorithm.

Given these numbers, it becomes apparent, that shutting down nodes (e.g., to memory) is an important aspect of reducing the power costs. Knowing fluctuating prices for a year in advance, an optimal schedule would be able to reduce costs further by 5-10%. Particularly, the day-night cycle in T3 provides room for exploitation. However, this benefit can never be achieved in practice as we do not know the hourly energy price of the stock market a year in advance. The base-peak model reduce costs sufficiently as well and is a practicable model as prices for base and peak are well known in advance and negotiated with the power supplier. The simulation will allow to quantify these numbers more accurately as it can consider the energy costs at a given time.

Shutting down the infrastructure additionally when not needed would reduce the costs further by 3% and 5% for LRZ and DKRZ, respectively. In the latter case, the infrastructure costs are a higher fraction, therefore the gain is higher.

However, as mentioned before, this benefit is in practice not achievable as restarting the infrastructure is extraordinary time consuming and may increase the hardware failures.

By assuming constant power consumption of the system, a naive estimate of cost saving would be (1 - utilization) – hence about 25% for our utilization figures. In general, the overall benefit is a bit lower than suggested, due to the fact that CPUs need more power when under load. However, the cost benefit of using an optimal schedule for the hourly cost models is relatively close to this assumption.

6.5. Simulation Results

First, we discuss what is shown in the tables. An example for DKRZ with the trace from 2017 is shown in Table 10. For different energy-cost models, the table shows the scheduler configuration and the resulting performance

in terms of days the jobs run in the simulation (run days) – as the trace covers roughly a year worth of workload, the overall utilization, the energy consumption, energy costs, the energy saving relative to BackFillDelay[11], and various waiting time statistics for pending jobs in minutes/days (the quartiles[12], Q1, Q2 (Median), Q3, quantile 90 and 99, and the maximum). The cells of the table have been colored, green is close to the best value, yellow a neutral (worse) result and red usually a bad result.

The table contains a selection of data albeit we have produced most data. Some aggressive schedulers that aim to conserve energy cannot be run successfully because their simulated runtime for the one year workload exceeded 500 days – after which the hourly cost model has no data available. In analysis, we pick FIFO and BackFillDelay as baselines for comparing the performance of alternative scheduling strategies. FIFO is expected to deliver a suboptimal waiting time for the users and should be considered as a upper bound for waiting. In a subset of conducted experiments, the FIFOBackFill and BackFillDelay performed similarly, so that we omit presentation of FIFOBackFill.

6.6. Discussion of the Schedulers

Comparing non energy-aware schedulers. The schedulers BackFillShutdownDelay and Backfill, LongestFirstBackfill, and BiggestFirstBackfill perform similarly to FIFO in respect to energy saving (look at fixed price). BiggestFirstBackfill and LongestFirstBackfill are able to reduce the waiting time as they are expected to pack the jobs tighter than FIFO. Since Backfill and BackFillShutdownDelay perform identically in respect to waiting times but differ only in energy consumption, we only show BackFillShutdownDelay. Also, their waiting time is not different between energy-cost models, hence we only show them for the fixed price model. The reason is that the overall utilization of the system is sufficiently low to have only few pending jobs. This also confirms that a backfill scheduler is typically sufficient to schedule jobs on a HPC system particularly. The BackFillDelay scheduler achieves generally significantly lower waiting times for most jobs (up to q99) com-

[11] BackFillDelay serves as baseline for the cost saving, the column saving is computed relative to BackFillDelay and shows how much money can be saved (in k€).

[12] Q1: The quickest dispatched 25% of all jobs must wait up to this time. Q2 is the median. Q3 is 75%. q90/q99 indicates the time 90% and 99% of all jobs finish before, respectively.

Configuration	Run days	Util.. %	Energy MWh	Energy k€	Saving k€	Waiting time in minutes/days (d) Q1	Q2	Q3	q90	q99	max
Fixed price											
FIFO	366.8	83.2	4,214	611.1	0.0	0	18	74	155	522	1 d
BackfillDelay	366.8	83.2	4,214	611.0	0.0	0	0	2	14	112	3 d
BiggestFirstBackfill	366.8	83.2	4,214	611.0	0	0	0	8	102	575	2 d
LongestFirstBackfill	366.8	83.2	4,214	611.0	0	0	0	2	54	466	7 d
BackfillShutdownDelay	366.8	83.2	4,056	588.1	22.9	0	0	2	14	112	3 d
Base-peak											
FIFO	366.8	83.2	4,214	611.0	0.1	0	18	74	155	522	1 d
BackfillDelay	366.8	83.2	4,214	611.1	0.0	0	0	2	14	112	3 d
BackfillShutdownDelay	366.8	83.2	4,056	589.1	22.0	0	0	2	14	112	3 d
EnforcePriceAware-12	700.7	43.6	5,098	694.6	-83.5	86 d	175 d	250 d	307 d	331 d	336 d
EnforcePriceAware-24	1115.8	27.4	6,393	822.0	-210.9	198 d	396 d	563 d	687 d	739 d	751 d
EnforcePriceAware-36	1144.7	26.7	6,483	834.9	-223.8	203 d	413 d	585 d	714 d	768 d	780 d
PriceAware-12	450.6	67.7	4,317	619.4	-8.3	20 d	39 d	59 d	74 d	83 d	84 d
PriceAware-24	476.4	64.1	4,398	628.1	-17.0	27 d	57 d	78 d	98 d	108 d	111 d
PriceAware-36	476.4	64.1	4,398	628.1	-17.0	27 d	57 d	78 d	98 d	108 d	111 d
PriceAwareDeadline-12	367.3	83.1	4,057	584.9	26.2	700	722	777	853	1129	2 d
PriceAwareDeadline-24	367.8	83.0	4,059	587.4	23.7	1390	1440	1 d	1 d	1 d	2 d
PriceAwareDeadline-36	368.3	82.9	4,061	585.3	25.8	1 d	2 d	2 d	2 d	2 d	3 d
Hourly cost model: T1											
FIFO	366.8	83.2	4,214	607.7	0.0	0	18	74	155	522	1 d
BackfillDelay	366.8	83.2	4,214	607.6	0.0	0	0	2	14	112	3 d
BackfillShutdownDelay	366.8	83.2	4,056	583.5	24.2	0	0	2	14	112	3 d
PriceAware-12	428.1	71.3	4,247	599.3	8.3	15 d	26 d	42 d	54 d	61 d	62 d
PriceAware-24	432.5	70.6	4,261	601.4	6.3	15 d	27 d	44 d	58 d	65 d	66 d
PriceAware-36	432.5	70.6	4,261	601.4	6.3	15 d	27 d	44 d	58 d	65 d	66 d
PriceAwareDeadline-12	367.3	83.1	4,057	583.9	23.8	688	721	774	848	1107	2 d
PriceAwareDeadline-24	367.8	83.0	4,059	584.1	23.5	1348	1440	1 d	1 d	1 d	2 d
PriceAwareDeadline-36	368.3	82.9	4,061	584.6	23.0	1 d	2 d	2 d	2 d	2 d	2 d
PriceAwareDeadline-48	368.8	82.7	4,062	584.7	22.9	2 d	2 d	2 d	2 d	2 d	2 d
PriceAwareDeadline-96	370.8	82.3	4,068	585.8	21.8	3 d	4 d	4 d	4 d	4 d	4 d
Hourly cost model: T2											
FIFO	366.8	83.2	4,214	609.8	0.0	0	18	74	155	522	1 d
BackfillDelay	366.8	83.2	4,214	609.8	0.0	0	0	2	14	112	3 d
BackfillShutdownDelay	366.8	83.2	4,056	586.6	23.1	0	0	2	14	112	3 d
PriceAware-12	418.6	72.9	4,217	602.2	7.5	12 d	23 d	36 d	46 d	51 d	52 d
PriceAware-24	418.6	72.9	4,217	602.2	7.5	12 d	23 d	36 d	46 d	51 d	52 d
PriceAware-36	418.6	72.9	4,217	602.2	7.5	12 d	23 d	36 d	46 d	51 d	52 d
PriceAwareDeadline-12	367.3	83.1	4,057	586.9	22.8	687	721	774	847	1092	2 d
PriceAwareDeadline-24	367.8	83.0	4,059	587.0	22.8	1345	1440	1 d	1 d	1 d	2 d
PriceAwareDeadline-36	368.3	82.9	4,061	587.4	22.4	1 d	2 d	2 d	2 d	2 d	2 d
PriceAwareDeadline-48	368.8	82.7	4,062	587.6	22.2	2 d	2 d	2 d	2 d	2 d	2 d
PriceAwareDeadline-96	370.8	82.3	4,068	588.5	21.3	3 d	4 d	4 d	4 d	4 d	4 d
Hourly cost model: T3											
FIFO	366.8	83.2	4,214	547.6	0.1	0	18	74	155	522	1 d
BackfillDelay	366.8	83.2	4,214	547.7	0.0	0	0	2	14	112	3 d
BackfillShutdownDelay	366.8	83.2	4,056	524.9	22.8	0	0	2	14	112	3 d
PriceAware-12	476.4	64.1	4,398	605.8	-58.2	25 d	48 d	81 d	100 d	109 d	110 d
PriceAwareDeadline-12	367.3	83.1	4,057	530.4	17.2	703	722	778	855	1115	2 d
PriceAwareDeadline-24	367.8	83.0	4,059	526.7	20.9	1408	1 d	1 d	1 d	1 d	2 d
PriceAwareDeadline-36	368.3	82.9	4,061	530.5	17.2	1 d	2 d	2 d	2 d	2 d	3 d
PriceAwareDeadline-48	368.8	82.7	4,062	527.3	20.3	2 d	2 d	2 d	2 d	2 d	3 d
PriceAwareDeadline-96	370.8	82.3	4,068	528.5	19.2	3 d	4 d	4 d	4 d	4 d	4 d

Table 10: Statistics for DKRZ 17

Configuration	Run days	Util.. %	Energy MWh	Energy k€	Saving k€	Waiting time in minutes/days (d) Q1	Q2	Q3	q90	q99	max
Fixed price											
FIFO	366.5	74.4	3,986	578.0	0.0	0	0	13	108	587	733
BackfillDelay	366.5	74.4	3,986	578.0	0.0	0	0	0	1	31	6 d
BiggestFirstBackfill	366.5	74.4	3,986	578.0	0	0	0	0	4	1121	12 d
LongestFirstBackfill	366.5	74.4	3,986	578.0	0	0	0	0	1	1 d	16 d
BackfillShutdownDelay	366.5	74.4	3,745	543.0	34.9	0	0	0	1	31	6 d
Base-peak											
FIFO	366.5	74.4	3,986	578.3	0.1	0	0	13	108	587	733
BackfillDelay	366.5	74.4	3,986	578.3	0.0	0	0	0	1	31	6 d
BackfillShutdownDelay	366.5	74.4	3,745	544.4	33.9	0	0	0	1	31	6 d
EnforcePriceAware-12	586.2	46.5	4,431	606.3	-28.0	51 d	76 d	138 d	179 d	213 d	220 d
EnforcePriceAware-24	923.2	29.5	5,482	709.3	-131.0	147 d	220 d	368 d	469 d	543 d	557 d
EnforcePriceAware-36	951.9	28.6	5,572	721.9	-143.6	157 d	235 d	390 d	495 d	571 d	585 d
PriceAware-12	396.2	68.8	3,838	550.2	28.1	4 d	5 d	15 d	19 d	28 d	30 d
PriceAware-24	419.4	65.0	3,910	557.6	20.7	8 d	12 d	31 d	42 d	51 d	53 d
PriceAware-36	419.4	65.0	3,910	557.6	20.7	8 d	12 d	31 d	42 d	51 d	53 d
PriceAwareDeadline-12	366.9	74.3	3,746	540.4	37.9	265	640	719	749	999	1189
PriceAwareDeadline-24	367.4	74.2	3,748	541.6	36.8	429	898	1436	1 d	1 d	1 d
PriceAwareDeadline-36	367.9	74.1	3,749	540.6	37.8	475	1204	1 d	2 d	2 d	2 d
Hourly cost model: T1											
FIFO	366.5	74.4	3,986	578.4	0.0	0	0	13	108	587	733
BackfillDelay	366.5	74.4	3,986	578.4	0.0	0	0	0	1	31	6 d
BackfillShutdownDelay	366.5	74.4	3,745	543.8	34.7	0	0	0	1	31	6 d
PriceAware-12	385.4	70.7	3,804	561.2	17.3	428	2 d	5 d	12 d	18 d	19 d
PriceAware-24	387.6	70.4	3,811	562.2	16.2	434	3 d	5 d	13 d	19 d	21 d
PriceAware-36	387.6	70.4	3,811	562.2	16.2	434	3 d	5 d	13 d	19 d	21 d
PriceAwareDeadline-12	366.9	74.3	3,746	544.6	33.8	187	586	718	745	967	1167
PriceAwareDeadline-24	367.4	74.2	3,748	545.8	32.6	260	686	1430	1 d	1 d	1 d
PriceAwareDeadline-36	367.9	74.1	3,749	547.0	31.4	295	961	1 d	2 d	2 d	2 d
PriceAwareDeadline-48	368.4	74.0	3,751	547.6	30.8	296	1033	2 d	2 d	2 d	2 d
PriceAwareDeadline-96	370.4	73.6	3,757	550.0	28.4	350	1 d	3 d	4 d	4 d	4 d
Hourly cost model: T2											
FIFO	366.5	74.4	3,986	577.2	0.0	0	0	13	108	587	733
BackfillDelay	366.5	74.4	3,986	577.2	0.0	0	0	0	1	31	6 d
BackfillShutdownDelay	366.5	74.4	3,745	542.3	35.0	0	0	0	1	31	6 d
PriceAware-12	378.6	72.0	3,783	546.3	31.0	356	1 d	4 d	7 d	11 d	12 d
PriceAware-24	378.6	72.0	3,783	546.3	31.0	356	1 d	4 d	7 d	11 d	12 d
PriceAware-36	378.6	72.0	3,783	546.3	31.0	356	1 d	4 d	7 d	11 d	12 d
PriceAwareDeadline-12	366.9	74.3	3,746	542.3	34.9	164	563	717	744	967	1167
PriceAwareDeadline-24	367.4	74.2	3,748	542.2	35.0	222	658	1424	1 d	1 d	1 d
PriceAwareDeadline-36	367.9	74.1	3,749	542.2	35.0	250	735	1 d	2 d	2 d	2 d
PriceAwareDeadline-48	368.4	74.0	3,751	542.3	34.9	261	858	2 d	2 d	2 d	2 d
PriceAwareDeadline-96	370.4	73.6	3,757	542.8	34.4	319	1326	3 d	4 d	4 d	4 d
Hourly cost model: T3											
FIFO	366.5	74.4	3,986	527.3	0.0	0	0	13	108	587	733
BackfillDelay	366.5	74.4	3,986	527.3	0.0	0	0	0	1	31	6 d
BackfillShutdownDelay	366.5	74.4	3,745	497.2	30.1	0	0	0	1	31	6 d
PriceAware-12	425.0	64.2	3,928	518.2	9.0	8 d	11 d	34 d	45 d	56 d	59 d
PriceAwareDeadline-12	366.9	74.3	3,746	490.4	36.8	312	652	720	760	1012	1240
PriceAwareDeadline-24	367.4	74.2	3,748	492.7	34.5	512	1044	1438	1 d	1 d	1 d
PriceAwareDeadline-36	367.9	74.1	3,749	490.5	36.8	568	1 d	1 d	2 d	2 d	2 d
PriceAwareDeadline-48	368.4	74.0	3,751	493.2	34.0	598	1 d	2 d	2 d	2 d	2 d
PriceAwareDeadline-96	370.4	73.6	3,757	493.5	33.8	665	2 d	4 d	4 d	4 d	4 d

Table 11: Statistics for DKRZ 15

Configuration	Run days	Util.. %	Energy MWh	Energy k€	Saving k€	Waiting time in minutes/days (d) Q1	Q2	Q3	q90	q99	max
Fixed price											
FIFO	394.7	73.1	14,242	2,065.1	-7.5	0	156	884	1 d	5 d	7 d
BackfillDelay	391.3	73.8	14,190	2,057.6	0.0	0	0	0	4	99	71 d
BiggestFirstBackfill	391.3	73.8	14,190	2,057.6	0.0	0	0	0	33	1 d	72 d
LongestFirstBackfill	391.3	73.8	14,190	2,057.6	0.0	0	0	0	2	525	72 d
BackfillShutdownDelay	391.3	73.8	13,077	1,896.2	161.4	0	0	0	4	99	71 d
Base-peak											
FIFO	394.7	73.1	14,242	2,059.4	-7.5	0	156	884	1 d	5 d	7 d
BackfillDelay	391.3	73.8	14,190	2,051.9	0.0	0	0	0	4	99	71 d
BackfillShutdownDelay	391.3	73.8	13,077	1,891.6	160.3	0	0	0	4	99	71 d
EnforcePriceAware-12	398.6	72.4	13,160	1,877.3	174.6	3 d	5 d	8 d	10 d	11 d	11 d
EnforcePriceAware-24	454.5	63.5	13,401	1,874.2	177.7	22 d	37 d	43 d	54 d	64 d	65 d
EnforcePriceAware-36	462.5	62.4	13,432	1,877.9	174.0	26 d	43 d	52 d	62 d	72 d	73 d
PriceAware-12	395.6	73.0	13,095	1,882.9	169.0	258	806	1 d	2 d	6 d	7 d
PriceAware-24	406.5	71.0	13,142	1,865.6	186.3	4 d	8 d	11 d	12 d	16 d	17 d
PriceAware-36	434.3	66.5	13,259	1,869.7	182.2	16 d	21 d	25 d	35 d	43 d	44 d
PriceAwareDeadline-12	394.8	73.1	13,092	1,887.4	164.5	40	404	997	1 d	5 d	7 d
PriceAwareDeadline-24	394.9	73.1	13,093	1,886.2	165.7	56	520	1226	1 d	5 d	7 d
PriceAwareDeadline-36	394.9	73.1	13,093	1,885.9	166.0	86	612	1417	2 d	5 d	7 d
Hourly cost model: T1											
FIFO	394.7	73.1	14,242	2,063.6	-0.7	0	156	884	1 d	5 d	7 d
BackfillDelay	391.3	73.8	14,190	2,062.9	0.0	0	0	0	4	99	71 d
BackfillShutdownDelay	391.3	73.8	13,077	1,900.6	162.3	0	0	0	4	99	71 d
PriceAware-12	397.9	72.5	13,105	1,908.0	154.9	223	1284	2 d	5 d	8 d	10 d
PriceAware-24	409.4	70.5	13,154	1,920.3	142.6	2 d	6 d	8 d	14 d	19 d	20 d
PriceAware-36	414.1	69.7	13,174	1,916.8	146.1	4 d	9 d	12 d	17 d	24 d	25 d
PriceAwareDeadline-12	394.8	73.1	13,092	1,897.0	165.9	38	398	1015	1 d	5 d	7 d
PriceAwareDeadline-24	394.9	73.1	13,092	1,896.0	166.9	45	516	1263	1 d	5 d	7 d
PriceAwareDeadline-36	394.9	73.1	13,093	1,897.2	165.7	51	606	1 d	2 d	5 d	7 d
PriceAwareDeadline-48	395.1	73.0	13,093	1,900.5	162.4	85	797	1 d	2 d	5 d	7 d
PriceAwareDeadline-96	395.1	73.0	13,094	1,902.8	160.2	119	1019	2 d	3 d	5 d	7 d
Hourly cost model: T2											
FIFO	394.7	73.1	14,242	2,052.2	-4.5	0	156	884	1 d	5 d	7 d
BackfillDelay	391.3	73.8	14,190	2,047.8	0.0	0	0	0	4	99	71 d
BackfillShutdownDelay	391.3	73.8	13,077	1,890.6	157.2	0	0	0	4	99	71 d
PriceAware-12	395.1	73.0	13,094	1,889.9	157.8	33	362	1169	2 d	5 d	7 d
PriceAware-24	395.1	73.0	13,094	1,889.3	158.5	42	489	1302	2 d	5 d	7 d
PriceAware-36	395.1	73.0	13,094	1,889.0	158.8	54	573	1416	2 d	5 d	7 d
PriceAwareDeadline-12	394.8	73.1	13,092	1,889.8	157.9	26	276	947	1 d	5 d	7 d
PriceAwareDeadline-24	394.8	73.1	13,092	1,889.9	157.9	28	327	1102	1 d	5 d	7 d
PriceAwareDeadline-36	394.8	73.1	13,092	1,890.0	157.8	29	344	1144	1 d	5 d	7 d
PriceAwareDeadline-48	394.8	73.1	13,092	1,890.0	157.8	30	350	1155	1 d	5 d	7 d
PriceAwareDeadline-96	394.9	73.1	13,092	1,889.5	158.3	31	359	1199	1 d	5 d	7 d
Hourly cost model: T3											
FIFO	394.7	73.1	14,242	1,843.9	-9.3	0	156	884	1 d	5 d	7 d
BackfillDelay	391.3	73.8	14,190	1,834.6	0.0	0	0	0	4	99	71 d
BackfillShutdownDelay	391.3	73.8	13,077	1,680.6	154.0	0	0	0	4	99	71 d
PriceAware-12	401.3	71.9	13,120	1,708.5	126.1	1336	2 d	4 d	6 d	11 d	12 d
PriceAware-24	474.0	60.9	13,428	1,808.2	26.4	27 d	36 d	55 d	74 d	83 d	84 d
PriceAwareDeadline-12	394.8	73.1	13,092	1,691.7	142.9	119	535	1131	1 d	5 d	7 d
PriceAwareDeadline-24	395.1	73.0	13,094	1,688.6	146.0	288	973	1 d	2 d	5 d	7 d
PriceAwareDeadline-36	395.3	73.0	13,094	1,695.5	139.0	398	1343	1 d	2 d	5 d	7 d
PriceAwareDeadline-48	396.1	72.9	13,098	1,693.8	140.7	621	1 d	2 d	3 d	6 d	8 d
PriceAwareDeadline-96	398.1	72.5	13,106	1,702.8	131.8	1104	2 d	4 d	5 d	8 d	10 d

Table 12: Statistics for LRZ 17

Configuration	Run days	Util.. %	Energy MWh	Energy k€	Saving k€	Waiting time in minutes/days (d) Q1	Q2	Q3	q90	q99	max
Fixed price											
FIFO	366.9	78.6	12,941	1,876.5	0.0	0	570	2 d	5 d	7 d	9 d
BackfillDelay	366.9	78.6	12,941	1,876.5	0.0	0	0	0	29	695	84 d
BiggestFirstBackfill	366.9	78.6	12,090	1,753.1	0.0	0	0	7	492	2 d	85 d
LongestFirstBackfill	366.9	78.6	12,090	1,753.1	0.0	0	0	0	119	2 d	111 d
BackfillShutdownDelay	366.9	78.6	12,090	1,753.1	123.5	0	0	0	29	695	84 d
Base-peak											
FIFO	366.9	78.6	12,941	1,872.5	3.9	0	570	2 d	5 d	7 d	9 d
BackfillDelay	366.9	78.6	12,941	1,876.4	0.0	0	0	0	29	695	84 d
BackfillShutdownDelay	366.9	78.6	12,090	1,756.0	120.5	0	0	0	29	695	84 d
EnforcePriceAware-12	368.0	78.4	12,106	1,727.3	149.1	7 d	8 d	12 d	14 d	15 d	18 d
EnforcePriceAware-24	416.1	69.3	12,313	1,729.3	147.1	32 d	44 d	51 d	52 d	53 d	54 d
EnforcePriceAware-36	430.0	67.0	12,373	1,737.1	139.3	37 d	54 d	64 d	64 d	66 d	68 d
PriceAware-12	367.0	78.6	12,090	1,741.8	134.7	306	1 d	6 d	8 d	9 d	11 d
PriceAware-24	381.0	75.7	12,150	1,724.7	151.7	16 d	18 d	19 d	20 d	21 d	23 d
PriceAware-36	414.0	69.7	12,290	1,733.4	143.0	28 d	41 d	49 d	50 d	50 d	51 d
PriceAwareDeadline-12	366.9	78.6	12,090	1,747.7	128.7	28	696	2 d	5 d	7 d	9 d
PriceAwareDeadline-24	366.9	78.6	12,090	1,746.4	130.0	40	791	2 d	5 d	7 d	9 d
PriceAwareDeadline-36	366.9	78.6	12,090	1,745.7	130.8	61	961	2 d	5 d	7 d	9 d
Hourly cost model: T1											
FIFO	366.9	78.6	12,941	1,848.4	5.2	0	570	2 d	5 d	7 d	9 d
BackfillDelay	366.9	78.6	12,941	1,853.6	0.0	0	0	0	29	695	84 d
BackfillShutdownDelay	366.9	78.6	12,090	1,727.4	126.1	0	0	0	29	695	84 d
PriceAware-12	366.9	78.6	12,090	1,715.1	138.4	987	2 d	6 d	8 d	9 d	11 d
PriceAware-24	366.9	78.6	12,090	1,721.3	132.3	5 d	6 d	11 d	14 d	17 d	19 d
PriceAware-36	387.2	74.5	12,176	1,761.9	91.6	18 d	22 d	24 d	25 d	26 d	27 d
PriceAwareDeadline-12	366.9	78.6	12,090	1,719.1	134.4	61	716	2 d	5 d	7 d	9 d
PriceAwareDeadline-24	366.9	78.6	12,090	1,718.5	135.0	94	841	2 d	5 d	7 d	9 d
PriceAwareDeadline-36	366.9	78.6	12,090	1,717.5	136.0	120	937	2 d	5 d	7 d	9 d
PriceAwareDeadline-48	366.9	78.6	12,090	1,716.8	136.7	241	1200	2 d	5 d	7 d	9 d
PriceAwareDeadline-96	366.9	78.6	12,090	1,715.8	137.7	543	1 d	3 d	5 d	7 d	9 d
Hourly cost model: T2											
FIFO	366.9	78.6	12,941	1,876.3	-2.7	0	570	2 d	5 d	7 d	9 d
BackfillDelay	366.9	78.6	12,941	1,873.6	0.0	0	0	0	29	695	84 d
BackfillShutdownDelay	366.9	78.6	12,090	1,748.9	124.7	0	0	0	29	695	84 d
PriceAware-12	366.9	78.6	12,090	1,753.9	119.7	20	848	2 d	6 d	8 d	10 d
PriceAware-24	366.9	78.6	12,090	1,754.1	119.5	32	1097	3 d	7 d	8 d	10 d
PriceAware-36	367.2	78.5	12,091	1,754.6	119.0	91	1 d	5 d	8 d	9 d	11 d
PriceAwareDeadline-12	366.9	78.6	12,090	1,752.6	121.0	14	625	2 d	5 d	7 d	9 d
PriceAwareDeadline-24	366.9	78.6	12,090	1,752.6	121.0	16	665	2 d	5 d	7 d	9 d
PriceAwareDeadline-36	366.9	78.6	12,090	1,752.4	121.2	17	709	2 d	5 d	7 d	9 d
PriceAwareDeadline-48	366.9	78.6	12,090	1,752.3	121.3	18	728	2 d	5 d	7 d	9 d
PriceAwareDeadline-96	366.9	78.6	12,090	1,752.3	121.3	19	801	2 d	5 d	7 d	9 d
Hourly cost model: T3											
FIFO	366.9	78.6	12,941	1,685.6	6.6	0	570	2 d	5 d	7 d	9 d
BackfillDelay	366.9	78.6	12,941	1,692.2	0.0	0	0	0	29	695	84 d
BackfillShutdownDelay	366.9	78.6	12,090	1,582.2	110.1	0	0	0	29	695	84 d
PriceAware-12	366.9	78.6	12,090	1,572.4	119.8	1 d	2 d	8 d	10 d	11 d	14 d
PriceAware-24	424.0	68.0	12,332	1,589.2	103.0	30 d	46 d	58 d	59 d	60 d	61 d
PriceAwareDeadline-12	366.9	78.6	12,090	1,570.6	121.7	175	797	2 d	5 d	7 d	9 d
PriceAwareDeadline-24	366.9	78.6	12,090	1,571.9	120.4	279	1117	2 d	5 d	7 d	9 d
PriceAwareDeadline-36	366.9	78.6	12,090	1,571.3	121.0	365	1 d	2 d	5 d	7 d	9 d
PriceAwareDeadline-48	366.9	78.6	12,090	1,573.3	118.9	702	1 d	2 d	5 d	7 d	9 d
PriceAwareDeadline-96	366.9	78.6	12,090	1,571.6	120.6	1389	2 d	4 d	6 d	8 d	10 d

Table 13: Statistics for LRZ 14

pared to any other scheduling policy. Even the waiting time for 90% of jobs (q90) is with a few minutes acceptable and 99% jobs are started within 2 hours. The LRZ 14 trace, however, has with up to 700 minutes a substantial waiting time for 9% of the jobs. However, a very few (big) jobs are delayed for several weeks. This is a bit unrealistic, as in practise, the priority of such a job would enforce execution at some point. Neither of these schedulers are able to save actual runtime of the workload as jobs are submitted until the end of the trace, and, thus, energy saving is comparable. In some cases, FIFO is a bit worse or better than the others. The latter is a coincidence and particularly may happen for the hourly energy-cost models when it accidentally schedulers jobs on cheaper energy prices.

Shutdown scheduler. The shutdown scheduler is effective to reduce the costs for both data centers and workloads. Again, this is due to the fact that less than 80% of nodes are utilized throughout the year, hence 20% of nodes can be turned of saving their power. As suggested by our theoretic model, the saving for LRZ is about 120k€ and 160k€ for all energy models for the trace of 2014 and 2017, respectively. For DKRZ it is 35k€ and 23k€ for 2015 and 2017, respectively. Therefore, shutdown of nodes, particularly with suspend to RAM would be effective for both data centers.

Enforcing energy saving. As the EnforceCheapEnergy scheduler delays execution of jobs to cheaper prices without considering the infrastructure energy costs, it leads to longer runs and actually increases the energy consumption and costs for DKRZ. For LRZ, it was effective to reduce the costs even more than the BackfillShutdown scheduler for the BasePeak model saving an extra of 10-20k€. Hence, this naive strategy is only applicable, when the utilization is sufficiently low and a predictable energy cost pattern is applied. However, even in that case it did increase the runtime by 10-20% and, hence, this is not a practicable approach. Note that any extension of runtime means we have to operate the data center longer without users submitting new jobs which is unrealistic. Generally, the more lookahead it has, the more it stalls jobs and the worse its performance.

Price-Aware scheduler. The PriceAware scheduler is typically able to reduce the costs (see, e.g., T1 for DKRZ 17) but may increase the overall energy consumption considerably. In most cases, the jobs are considerable delayed and, hence, the runtime of the traces increases but there is considerable cost

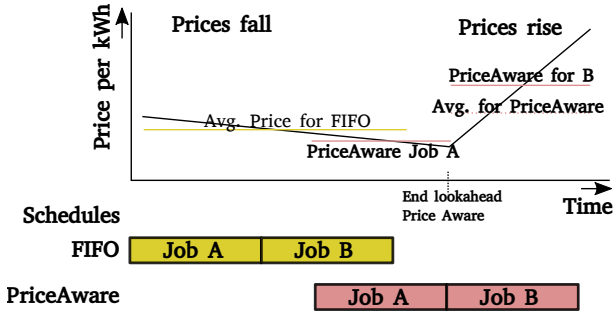


Figure 6: Example for FIFO achieving better energy costs for dispatching two jobs on a single node. The lower part shows the actual execution of jobs.

saving compared to BackfillDelay. Still, compared to the BackfillShutdown-Delay scheduler, the PriceAware scheduler yields worse cost saving. The look-ahead window has some influence but longer look-ahead does not increase energy saving and may even be worse, which is counter intuitive.

The reason for the worse results is due to the heuristic nature of the scheduling algorithm; the algorithm in combination with the pending jobs tries to make smart choices, but future jobs are penalized more than we win in the short term. We give an example in the following. Assume we have only one node and two jobs that are submitted at t=0; the given cost function and runtimes as illustrated in Figure 6. The price first drops continuously then rises rapidly. The FIFO scheduler dispatches both jobs immediately, achieving an average cost for the energy as marked with the vertical bar. The PriceAware scheduler would compare the energy costs for leaving the nodes idle vs. the gain when starting late. We assume due to the falling price the gain is still a bit better, hence, the scheduler would dispatch the job as late as possible just before the price rises. Job A would get the lowest energy costs. However, then it must schedule Job B in a period of rapidly rising costs, at best it schedules it immediately. As you can see the mean energy costs (of both jobs) scheduler are now actually higher than for FIFO and we even delayed the user jobs further! Now, one may argue that the scheduler should be able to postpone both jobs as much as possible in the phase with decreasing costs, but the lookahead window may just end right before the prices rise again. Therefore, the heuristics is unable to exploit the potential gain.

Price-Aware-Deadline scheduler. While the scheduler works similarly to the previous scheduler, its main difference is that it enforces to dispatch jobs when the submission time has passed the lookup window size. While this still leads to considerable delay times particularly during peak times, this strategy actually leads to a reduction of costs while it may increase the energy consumption slightly. Generally, the algorithm does what it intended to do, aiming to reduce costs.

Table 14 shows the cost savings compared to the BackfillDelay algorithm, and shows the cost benefit of the PriceAware algorithm over the BackFill-ShutdownDelay algorithm. As suggested by the enforcement of the algorithms, the additional cost savings are minimal and often the shutdown scheduler is still better. Consequently, the presented algorithms are unable to exploit the theoretic optimal gain of an optimal scheduling.

Trace	BackFillShutdownDelay cost saving				PriceAwareDeadline cost saving				benefit			
	BasePk	T1	T2	T3	BasePk	T1	T2	T3				
DKRZ 14	33.9	34.7	35.0	30.1	36.8	32.6	35.0	34.5	2.9	-2.1	0.0	4.4
DKRZ 17	22.0	24.2	23.1	22.8	23.7	23.5	22.8	20.9	1.7	-0.7	-0.3	-1.9
LRZ 15	33.9	34.7	35.0	30.1	36.8	32.6	35.0	34.5	2.9	-2.1	0.0	4.4
LRZ 17	160.3	162.3	157.2	154.0	165.7	166.9	157.9	146.0	5.4	4.6	0.7	-8.0

Table 14: Simulated cost savings in k€ compared to the BackfillDelay algorithm and between the two cost-aware schedulers

6.7. TCO Considerations

This section aims to compare the cost savings to the Total Cost of Ownership (TCO). TCO of a given Data Center (DC) is the aggregated sum of all costs spent on using and acquiring DC's assets.

We compare three scenarios:

- **Current**: This is the current configuration, where we purchase the machine with the characteristics in Table 2. In this case, we assume to pay the average price according to the trace.
- **Optimal energy costs**: Here, we use the theoretically computed values when we 1) shutdown the client nodes; 2) use the optimal scheduling; 3) shutdown infrastructure while it is not needed (see Table 8).
- **Optimal procurement**: In this scenario, we are able to dynamically buy exactly the fraction of the system according to its utilization. We

assume the system costs are proportional to the utilization[13] but the annual costs stay the same. In this case, we assume to pay the average price according to the trace.

Note that the following analysis is highly approximative due to the inaccuracy of the publicly available data about system costs and annual costs, and the assumption under the scenarios. An additional assumption is that the procurement costs (CAPEX) are equally shared among 5 years of system life time. The results of the three scenarios are summarized in Table 15. The table shows for each scenario the expenses for the system procurement and the annual upkeep, the consumed energy of the strategy, the implied energy costs, and the TCO assuming that staff and infrastructure serves the sole purpose of operating and supporting a machine.

Trace	System	Energy	Energy costs in k€				TCO in k€			
	Ex. in k€	in MWh	BasePeak	T1	T2	T3	BasePeak	T1	T2	T3
LRZ 14	32,600	12,941	1,870	1,876	1,874	1,699	34,470	34,476	34,474	34,299
LRZ 17	32,600	14,189	2,050	2,064	2,039	1,857	34,650	34,664	34,639	34,457
DKRZ 15	7,300	3,986	576	578	577	523	7,876	7,878	7,877	7,823
DKRZ 17	7,300	4,214	609	611	610	553	7,909	7,911	7,910	7,853

(a) Current situation – the baseline

Trace	System	Energy	Energy costs in k€				TCO in k€			
	Ex. in k€	in MWh	BasePeak	T1	T2	T3	BasePeak	T1	T2	T3
LRZ 14	32,600	11,758	1,628	1,511	1,612	1,354	34,228	34,111	34,212	33,954
LRZ 17	32,600	12,643	1,727	1,589	1,703	1,407	34,327	34,189	34,303	34,007
DKRZ 15	7,300	3,452	472	433	470	386	7,772	7,733	7,770	7,686
DKRZ 17	7,300	3,864	541	509	535	459	7,841	7,809	7,835	7,759

(b) Optimal energy consumption (client/infra. shutdown) and scheduling

Trace	System	Energy	Energy costs in k€				TCO in k€			
	Ex. in k€	in MWh	BasePeak	T1	T2	T3	BasePeak	T1	T2	T3
LRZ 14	29,048	11,758	1,699	1,705	1,703	1,543	30,747	30,752	30,750	30,590
LRZ 17	28,251	12,643	1,827	1,840	1,660	1,833	30,078	30,090	29,911	30,084
DKRZ 15	6,455	3,452	499	501	501	500	6,954	6,956	6,956	6,955
DKRZ 17	6,746	3,864	558	560	559	507	7,304	7,306	7,305	7,253

(c) Optimal procurement

Table 15: Theoretical TCO comparison for the three different scenarios and one year

Observations: As the energy costs account for only a fraction of the TCO, the saving of the **optimal energy costs** strategy is about 1-2% of

[13] In practise, the scaling is not linear due to the economy of scale, but as we still keep more than 75% of the system, this is a rough estimate.

the TCO. As expected, the dynamic and **optimal procurement** leads to substantial cost savings, as 10% of the procurement costs could be saved. However, this scenario is highly unlikely as it would increase the waiting time for users significantly. Therefore, we consider a slightly overprovisioned machine is optimal, that means that typical user jobs can be handled by the local supercomputer, while bursts of user jobs with high priority can be offloaded to the cloud to minimize waiting time.

7. Summary & Conclusion

In this paper, we discussed the benefit of using alternative energy and cost-saving strategies in data centers and quantified the estimated benefit for DKRZ and LRZ data centers. We explored the advantage of using billing modalities for a BasePeak and three different hourly pricing models. While the theoretic and best-case assessment suggests cost-savings up to 19% and 15% for LRZ and DKRZ, respectively, the more realistic simulation showed that these theoretical benefits can be barely achieved. This may be partially due to the fact that FIFO schedulers are used as baselines. Besides the low gain of cost-aware schedulers, they substantially increase the waiting time of users' jobs. For the assessment, we used four different cost models, including a base/peak model, and three synthetically generated traces. The models based on the stock market hourly prices prove to be not beneficial when assuming the average price is comparable to the other cost models. In addition, we observed almost no cost-saving with our scheduling strategies compared to the client node shutdown. However, the theoretical analysis of the TCO shows where to conduct further research.

Still, this article discussed various viable alternatives. In terms of practicability, we believe a base/peak model is a viable option since it provides a more predictable turn-around-time as the machine is highly utilized during night time while pending and interactive jobs can still be dispatched during day time[14]. Additionally, the significance of cost-saving by turning off client nodes is such as to motivate the deployment in a productive operation; the gain is in the order of two full-time-equivalent researchers at LRZ[15] and $\frac{1}{2}$ for DKRZ. While existing schedulers support shutdown/restart of nodes, we

[14]This model can still be applied when buying energy from the stock market, however, we just classify the price into base/peak classes.

[15]Salary about 60 k€ per year.

claim suspending to memory might be an optimal strategy to reduce wear-down of the components and improve the response time. In respect of TCO, an optimal and timely procurement of hardware upon the need provides a significantly lower cost even in comparison with the best energy-savings methods. As it is expected, a high utilization of data centers is obviously the key to efficient science. A model presumably would prove cost-efficient by applying a slight overprovisioning of hardware and using methods to dynamically overflow workloads into the cloud while imposing a low waiting-time on users' jobs - particularly whenever bursts of jobs are rarely submitted. However, this requires a seamless experience at users' side, in terms of capabilities such as data access and visualization.

Based on the results described in this paper, we will explore the cost-benefit of further billing and scheduling strategies that were mentioned earlier in the text. However, since energy is only a fraction of the costs with a limited potential, as shown, the main focus of research is to gain a deeper insight into TCO and its implication on workflows and producing scientific outputs using supercomputers.

8. Acknowledgment

Thanks to Gerald Vogt for providing us with the energy price lists of DKRZ and his support to explain the details. The authors would like to also thank Dr. Detlef Labrenz (LRZ) for his comments and suggestions.

References

[1] A. Shehabi, S. Smith, N. Horner, I. Azevedo, R. Brown, J. Koomey, E. Masanet, D. Sartor, M. Herrlin, W. Lintner, United States Data Center Energy Usage Report, Lawrence Berkeley National Laboratory, Berkeley, California. LBNL-1005775 Page 4.

[2] Fraunhofer Institute for Solar Energy Systems, Energy charts (2017). URL https://www.energy-charts.de/index.htm

[3] Y. Joshi, P. Kumar, Energy efficient thermal management of data centers, Springer Science & Business Media, 2012.

[4] M. K. Patterson, S. W. Poole, C.-H. Hsu, D. Maxwell, W. Tschudi, H. Coles, D. J. Martinez, N. Bates, Tue, a new energy-efficiency metric

applied at ornl's jaguar, in: International Supercomputing Conference, Springer, 2013, pp. 372–382.

[5] D. Azevedo, M. Patterson, J. Pouchet, R. Tipley, Carbon usage effectiveness (cue): a green grid data center sustainability metric, White Paper 32.

[6] T. Wilde, A. Auweter, M. K. Patterson, H. Shoukourian, H. Huber, A. Bode, D. Labrenz, C. Cavazzoni, Dwpe, a new data center energy-efficiency metric bridging the gap between infrastructure and workload, in: High Performance Computing & Simulation (HPCS), 2014 International Conference on, IEEE, 2014, pp. 893–901.

[7] A. Damodaran. [link].
URL http://pages.stern.nyu.edu/~adamodar/New_Home_Page/AppldCF/derivn/ch5deriv.html

[8] Ofgem (2017). [link].
URL https://www.ofgem.gov.uk/system/files/docs/2017/01/guide_to_riioed1.pdf

[9] tco. [link].
URL https://en.wikipedia.org/wiki/Total_cost_of_ownership#cite_note-1

[10] G. L. Valentini, W. Lassonde, S. U. Khan, N. Min-Allah, S. A. Madani, J. Li, L. Zhang, L. Wang, N. Ghani, J. Kolodziej, et al., An overview of energy efficiency techniques in cluster computing systems, Cluster Computing 16 (1) (2013) 3–15.

[11] J. Gao, Machine-learning-applicationsfor-datacenter-optimization.

[12] H. Shoukourian, T. Wilde, D. Labrenz, A. Bode, Using machine learning for data center cooling infrastructure efficiency prediction, in: IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2017, pp. 954–963. doi:https://doi.org/10.1109/IPDPSW.2017.25.

[13] H. Li, J. Li, W. Yao, S. Nazarian, X. Lin, Y. Wang, Fast and energy-aware resource provisioning and task scheduling for cloud systems,

in: 2017 18th International Symposium on Quality Electronic Design (ISQED), 2017, pp. 174–179. doi:10.1109/ISQED.2017.7918312.

[14] S. K. Garg, C. S. Yeo, A. Anandasivam, R. Buyya, Energy-efficient scheduling of hpc applications in cloud computing environments, arXiv preprint arXiv:0909.1146.

[15] D. Aikema, C. Kiddle, R. Simmonds, Energy-cost-aware scheduling of HPC workloads, in: World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a, IEEE, 2011, pp. 1–7.

[16] O. Mämmelä, M. Majanen, R. Basmadjian, H. De Meer, A. Giesler, W. Homberg, Energy-aware job scheduler for high-performance computing, Computer Science-Research and Development 27 (4) (2012) 265–275.

[17] E. Roloff, M. Diener, A. Carissimi, P. O. Navaux, High performance computing in the cloud: Deployment, performance and cost efficiency, in: Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, IEEE, 2012, pp. 371–378.

[18] H. Shoukourian, Adviser for Energy Consumption Management: Green Energy Conservation, Ph.D. thesis, München, Technische Universität München (TUM) (2015).

[19] H. Shoukourian, T. Wilde, A. Auweter, A. Bode, D. Tafani, Predicting energy consumption relevant indicators of strong scaling hpc applications for different compute resource configurations, in: Proceedings of the Symposium on High Performance Computing, HPC '15, Society for Computer Simulation International, San Diego, CA, USA, 2015, pp. 115–126.
URL http://dl.acm.org/citation.cfm?id=2872599.2872614

[20] H. Shoukourian, T. Wilde, A. Auweter, A. Bode, Power Variation Aware Configuration Adviser for Scalable HPC Schedulers, in: Proceedings of the 13 International Conference on High Performance Computing & Simulation, HPCS, IEEE, 2015, pp. 71–79. doi:10.1109/HPCSim.2015.7237023.
URL http://dx.doi.org/10.1109/HPCSim.2015.7237023

[21] H. Chen, X. Zhu, H. Guo, J. Zhu, X. Qin, J. Wu, Towards energy-efficient scheduling for real-time tasks under uncertain cloud computing environment, Journal of Systems and Software 99 (2015) 20 – 35. doi:https://doi.org/10.1016/j.jss.2014.08.065.
URL http://www.sciencedirect.com/science/article/pii/S0164121214001903

[22] M. Demirci, A survey of machine learning applications for energy-efficient resource management in cloud computing environments, in: Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on, IEEE, 2015, pp. 1185–1190.

[23] J. L. Berral, R. Gavalda, J. Torres, Adaptive scheduling on power-aware managed data-centers using machine learning, in: Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing, IEEE Computer Society, 2011, pp. 66–73.

[24] T. Miyazaki, I. Sato, N. Shimizu, Bayesian optimization of hpc systems for energy efficiency, in: International Conference on High Performance Computing, Springer, 2018, pp. 44–62.

[25] M. Cheng, J. Li, S. Nazarian, Drl-cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers, in: 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), 2018, pp. 129–134. doi:10.1109/ASPDAC.2018.8297294.

[26] D. B. Jackson, H. L. Jackson, Q. O. Snell, Simulation based hpc workload analysis, in: Parallel and Distributed Processing Symposium., Proceedings 15th International, IEEE, 2001, pp. 8–pp.

[27] J. Brennan, I. Kureshi, V. Holmes, Cdes: an approach to hpc workload modelling, in: Distributed Simulation and Real Time Applications (DS-RT), 2014 IEEE/ACM 18th International Symposium on, IEEE, 2014, pp. 47–54.

[28] Leibniz Supercomputing Centre (LRZ) of the Bavarian Academy of Sciences, Humanities (2018). [link].
URL http://www.lrz.de/

[29] Top500 (2018). [link].
URL http://top500.org/

[30] Leibniz Supercomputing Centre, Supermuc Petascale System, https://www.lrz.de/services/compute/supermuc/ (2018).

[31] Gauss Centre for Supercomputing (2018). [link].
URL http://www.gauss-centre.eu

[32] Partnership for Advanced Computing in Europe (2018). [link].
URL http://www.prace-ri.eu/

[33] American Society of Heating, Refrigerating and Air-Conditioning Engineers (2018).
URL https://www.ashrae.org/home

[34] H. Shoukourian, T. Wilde, H. Huber, A. Bode, Analysis of the efficiency characteristics of the first high-temperature direct liquid cooled petascale supercomputer and its cooling infrastructure, Journal of Parallel and Distributed Computing 107 (2017) 87 – 100. doi:https://doi.org/10.1016/j.jpdc.2017.04.005.
URL http://www.sciencedirect.com/science/article/pii/S0743731517301272

[35] IBM (2018). [link].
URL http://www.ibm.com

[36] Green500 (2018). [link].
URL http://www.green500.org/

[37] Intel (2018). [link].
URL http://www.intel.com

[38] IBM LoadLeveler (2017).
URL https://www-03.ibm.com/systems/power/software/loadleveler/

[39] T. Wilde, PhD Dissertation: Assessing the Energy Efficiency of High Performance Computing (HPC) Data Centers, Ph.D. thesis (2018).

[40] A. Auweter, A. Bode, M. Brehm, L. Brochard, N. Hammer, H. Huber, R. Panda, F. Thomas, T. Wilde, A Case Study of Energy Aware Scheduling on SuperMUC, Springer International Publishing, Cham, 2014, pp.

394–409. doi:10.1007/978-3-319-07518-1_25.
URL https://doi.org/10.1007/978-3-319-07518-1_25

[41] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano, L. Benini, Scheduling-based power capping in high performance computing systems, Sustainable Computing: Informatics and Systems 19 (2018) 1 – 13. doi:https://doi.org/10.1016/j.suscom.2018.05.007.
URL https://www.sciencedirect.com/science/article/pii/S2210537917302317

[42] A. Bode, Energy to Solution: A New Mission for Parallel Computing., in: Euro-Par, Springer, 2013, pp. 1–2.

[43] C. Loken, D. Gruner, L. Groer, R. Peltier, N. Bunn, M. Craig, T. Henriques, J. Dempsey, C.-H. Yu, J. Chen, et al., Scinet: lessons learned from building a power-efficient top-20 system and data centre, in: Journal of Physics: Conference Series, Vol. 256, IOP Publishing, 2010, p. 012026.

[44] Stadtwerke München GmbH (2018). [link].
URL https://www.swm.de/

[45] E. Energy Germany (2018). [link].
URL https://www.eon.de/

[46] TenneT European electricity transmission system (2018). [link].
URL https://www.tennet.eu/

[47] N. Bates, G. Ghatikar, G. Abdulla, G. A. Koenig, S. Bhalachandra, M. Sheikhalishahi, T. Patki, B. Rountree, S. Poole, Electrical grid and supercomputing centers: An investigative analysis of emerging opportunities and challenges, Informatik-Spektrum 38 (2) (2015) 111–127. doi:10.1007/s00287-014-0850-0.
URL https://doi.org/10.1007/s00287-014-0850-0

[48] D. Hackenberg, T. Ilsche, J. Schuchart, R. Schöne, W. E. Nagel, M. Simon, Y. Georgiou, Hdeem: high definition energy efficiency monitoring, in: Proceedings of the 2nd International Workshop on Energy Efficient Supercomputing, IEEE Press, 2014, pp. 1–10.

[49] H. Vogel, D. Bäumer, M. Bangert, K. Lundgren, R. Rinke, T. Stanelle, COSMO-ART: Aerosols and reactive trace gases within the COSMO model, in: Integrated Systems of Meso-Meteorological and Chemical Transport Models, Springer, 2010, pp. 75–80.

[50] H. Shoukourian, T. Wilde, A. Auweter, A. Bode, Monitoring Power Data: A first step towards a unified energy efficiency evaluation toolset for HPC data centers, Elsevier, 2013.
URL http://dx.doi.org/10.1016/j.envsoft.2013.11.011