Integrating Dynamic Pricing of Electricity into Energy Aware Scheduling for HPC Systems

Xu Yang, Zhou Zhou, Sean Wallace, Zhiling Lan
Illinois Institute of Technology, Chicago, IL, USA
{xyang56, zzhou1, swallac6}@hawk.iit.edu, lan@iit.edu

Wei Tang, Susan Coghlan, Michael E. Papka
Argonne National Laboratory, Argonne, IL, USA
{wtang, smc, papka}@anl.gov

ABSTRACT

The research literature to date mainly aimed at reducing energy consumption in HPC environments. In this paper we propose a job power aware scheduling mechanism to reduce HPC's electricity bill without degrading the system utilization. The novelty of our job scheduling mechanism is its ability to take the variation of electricity price into consideration as a means to make better decisions of the timing of scheduling jobs with diverse power profiles. We verified the effectiveness of our design by conducting trace-based experiments on an IBM Blue Gene/P and a cluster system as well as a case study on Argonne's 48-rack IBM Blue Gene/Q system. Our preliminary results show that our power aware algorithm can reduce electricity bill of HPC systems as much as 23%.

Keywords

Power, Electricity Bill, Job Scheduling, System Utilization

1. INTRODUCTION

As HPC systems continue to grow so does their energy consumption. The electricity bill is now a leading component of total cost of ownership(TCO) of HPC systems. The typical current petascale system on average consumes 2-7 MW of power [21]. Case in point, the Argonne Leadership Computing Facility (ALCF) budgets approximately $1M annually for electricity to operate its primary supercomputing resource. Based on current projections, exascale supercomputers will consume 60-130 MW, which will prove to be an unbearable burden for any facility. Therefore, electricity savings are crucial for reducing the operational cost of extreme scale systems.

There is a significant number of research studies about improving energy efficiency for HPC systems and most of them focusing on the following topics: energy-efficient or energy proportional hardware, dynamic voltage and frequency scaling (DVFS) techniques, shutting down hardware components at low system utilizations, power capping, and thermal management.

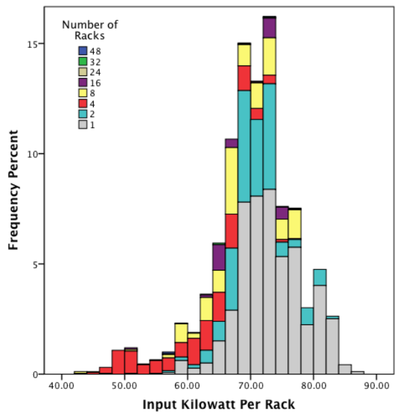

Figure 1: Job Power Distribution on BGQ

Being orthogonal to existing studies, this work focuses on *reducing the electricity bill of HPC systems* via a smart job scheduling mechanism. The rationale is based on a key observation of HPC jobs: *parallel jobs have distinct power consumption profiles.* Hennecke et al. [13] analyzed the energy characteristics of the production workload at Research Center Julich (FJZ) and found the power consumption of the jobs running on their Blue Gene/P(BGP) system range from 20 kW/rack to 33 kW/rack per. In our recent work [28], we provided analysis of a one-month workload on Mira, the 48-rack IBM Blue Gene/Q (BGQ) system at Argonne National Laboratory (Figure 1). The histogram displays percentages partitioned by size ranging from single rack jobs to full system runs. The power consumption of those jobs varies from around 40 kW/rack to 90 kW/rack. It can't tell from the figure that the size of these jobs affects their energy efficiency or not.

We hypothesize that it is possible to save a significant amount on an electric bill by exploiting a dynamic electricity pricing policy. To date, dynamic electricity pricing policies have been widely adopted in Europe, North America, Oceania, and parts of Asia. For example, in the U.S.A, wholesale electricity prices vary by as much as a factor of 10 from one hour to the next [23]. Under dynamic pricing, the power grid has on-peak time (when it bears a heavier burden and consequently the electricity price is higher) and off-peak time (when there is less demand for electricity and the price is lower) alternatively in a day.

In this work, we develop a job power aware scheduling mechanism. The novelty of this scheduling mechanism is that *it can reduce system's electricity bill by scheduling and dispatching jobs according to their power profiles and the real time electricity price, while causing negligible impact on the system's utilization and scheduling fairness.* Preferentially, it dispatches the jobs with higher power consumption during the off-peak period, and the jobs with lower power consumption during the on-peak period.

A key challenge in HPC scheduling is that system utilization should not be impacted. HPC systems require a tremendous capital investment, hence taking full advantage of this expensive resources is of great importance to HPC centers. Unlike the utilization of Internet data centers, which only fluctuate about 20%, systems at HPC centers are highly employed with a typical utilization around 50%-80% [3]. To address this challenge, we propose a novel window-based scheduling mechanism[26]. In this work, rather than allocating jobs one by one from the front of the wait queue as existing schedulers do, we schedule and dispatch a "window" of jobs at a time. Those jobs placed into the window are chosen to maintain job fairness, and the allocation of these jobs onto system resources is done in such a way as to minimize electricity bill. Two scheduling algorithms, namely a Greedy policy and a 0-1 Knapsack based policy, are presented in this work for decision-making.

We evaluate our job power aware scheduling design via extensive trace-based simulations and a case study of Mira. In this paper, we present a series of experiments comparing our design against the popular first-come, first-serve (FCFS) scheduling policy, with backfilling done in three different aspects (electricity bill saving, scheduling performance, and job performance). Our preliminary results demonstrate that our design can cut electricity bill by up to 23% without an impact on overall system utilization. Considering HPC centers often spend millions of dollars on even the least expensive energy contracts, such savings can translate into a reduction of hundreds of thousands in terms of TCO.

The structure of the paper is as follows. In section 2, we discuss the existing work about energy reduction for HPC systems. Section 3 describes the job power aware scheduling problem. Section 4 gives a detailed description of our job power aware scheduling design. Sections 5-7 present our evaluation methodology, trace-based simulations, and a case study by comparing our design against the widely-used FCFS scheduling policy under a variety of configurations. Our findings are presented in Section 8.

2. RELATED WORK

First, we give a brief survey of dynamic electricity pricing policies in different countries to demonstrate the applicability of this work. The European Exchange Market (EEX) in Germany, PowerNext in France, and APX in the Netherlands and Iberian market all vary their cost of electricity on an hourly basis[12]. In [20], the author stated dynamic electricity pricing policies are well adopted in Nordic countries as Norway, Finland, Sweden, and Denmark. Other power markets such as England, New Zealand, and Australia also have similar policies. Since the electricity crisis in 2011-2012, the Japanese Ministry of Economy, Trade and Industry (METI) has initiated the Smart Community Pilot Projects in four cities in Japan(Yokohama, Toyota, Kyoto, and Kitakyushu) to investigate the effect of dynamic pricing and smart energy equipment on residential electricity demand[15]. In China, major cities such as Beijing, Shanghai, Guangzhou have initiated dynamic electricity pricing for both domestic and industrial use since 2006. Dynamic electricity pricing has also been carried out in several provinces such as Zhejiang, Jiangsu, and Guangdong.

As we can see from the survey above, the dynamic electricity pricing policy has already been carried out in power markets in Europe, North America, Oceania, and China. Japan has initiated preliminary tests in some major cities to see the effect of this dynamic electricity pricing policy on the reduction of electricity consumption. While in this study we evaluate our design based on the on-/off-peak electricity pricing in U.S.A, we believe our design is applicable to other countries(e.g., those listed above) for cutting the electricity bill of their HPC systems.

Although there is no known effort to provide job power aware scheduling support in the field of HPC, there is a large body of related work. Due to space limitations, in this section we discuss some closely related studies and point out key differences among them.

From the hardware perspective, hardware vendors are dedicated to produce energy-efficient devices. For instance, Barroso and Holzle [5] argued that the power consumption of a machine should be proportional to its workload, i.e., it should consume no power in idle state, almost no power when the workload is very low, and eventually more power when the workload is increased. Ideally, an energy proportional system could save half of the energy used in data center operations. Li [18] optimized the power/ground grid to make the power supply more efficient for the chip.

Since processor power consumption is a significant portion of the total system power (roughly 50% under load [14]), DVFS is widely used for controlling CPU power [7]. By running a processor at a lower frequency/voltage, energy savings can be achieved at the expense of increased job execution time. In order to meet user's SLAs (Service Level Agreements), DVFS is typically applied at the period of low system activity. Some research studies on this topic can be found in [22] [19] [9] [16].

In a typical HPC system, nodes often consume considerable energy in idle state without any running application. For example, an idle Blue Gene/P rack still has a DC power consumption of about 13 kW [13]. During low system utilization, some nodes or their components could be shut down or switched to a low-power state. This strategy tries to minimize the number of active nodes of a system while still satisfying incoming application requests. Since this approach is highly dependent on system workload, the challenge is to determine when to shut down components and how to provide a suitable job slowdown value based on the availability of nodes.

Hikita et al.[14] performed an empirical study by implementing an energy-aware scheduler for an HPC system. The operation of the scheduler is simple: if a node is inactive for 30 minutes, it is powered off; when the node is required for job execution, it is powered on and moved to an active state. Powering up a node on their system takes approximately 45 minutes, which is substantial. This strategy can improve power efficiency by 39% at best. Because the rebooting of a node may consume significant time that will lead to a performance degradation if it happens to be peak job request period and more nodes are required than the active ones.

Pinheiro et al. [22] presented a mechanism that dynamically turns cluster nodes on and off. This approach uses load consolidation to transfer workload onto fewer nodes so that idle nodes can be turned off. The experimental tests on a static 8-node cluster indicate a 19% saving in energy. It takes about 100 seconds to power on a server and 45 seconds to shut it down on the cluster. The degradation in performance is approximately 20%.

Thermal management techniques are another method frequently discussed in the literature [19] [16]. The rationale is that higher temperatures have a large impact on system reliability and can also increase cooling costs. By using thermal management, system workload is adjusted according to a predefined temperature threshold: if the temperature on a server rises above that threshold, its workload is reduced. Disadvantages of thermal management are delayed response, high risk of overheating, excessive cooling and recursive cycling [16].

Many data centers use power capping or power budgeting to reduce the total power consumption. The operator can set a threshold of power consumption to ensure the actual power of the data center does not exceed it [10]. It prevents sudden rises in power supply and keeps the total power consumption under a predefined budget. Basically, the power consumption can be reduced by rescheduling tasks or CPU throttling, for example, Etinski et al. proposed a parallel job scheduling policy based on integer linear programming under a given power profile [9]. Lefurgy et al. presented a technique for high density servers that controls the peak power consumption by implementing a feedback controller [17].

This work has two major differences as compared to our previous work presented in [30]. First, our previous work targets Blue Gene/P systems, which have a special requirement on job scheduling, i.e., available nodes must be connected in a job specific shape before they can be allocated to a job[26] [24]. This work intends to provide a generic job power aware scheduling mechanism for various HPC systems. Second, our previous work relies on a power budget (similar to power capping) for energy cost saving, which degrades system utilization slightly during on-peak electricity price period. The scheduling policies presented in this work do not use power budget, and they minimize the electricity bill without impacting system utilization, during both on-peak and off-peak electricity pricing periods.

3. PROBLEM DESCRIPTION

Typically, user jobs are submitted to an HPC system through a batch scheduler, and then wait in a queue for the requested amount of system resources to become available. There may be one or multiple job queues with different priories. A job is generally defined by its arrival time, its estimated runtime, the amount of computing nodes requested, etc. The scheduler is responsible for assigning computing nodes to the jobs in the queues. FCFS with backfilling is a commonly used scheduling policy in HPC [11]. Under this policy, the scheduler picks a job from the head of the wait queue and dispatches it to the available system resources. The nodes assigned to a job become unavailable until the job is complete (i.e., space sharing).

As mentioned in Section 1, our work is based on two key observations in HPC: (1) electricity price is dynamically changing within a day; and (2) HPC jobs have distinct power consumption profiles. We shall point out that usually HPC jobs also tend to be repetitive. These repetitive jobs can be easily identified by user ID, project, expected runtime, etc. A batch scheduler can extract job power profile based on historical data and use it for power aware scheduling. For the simplicity of method description, we assume that a daily electricity price is divided into on-peak and off-peak periods, where on-peak period is referred to the time when more electricity is demanded (e.g., during the daytime). By exploring these two observations, the basic idea of our work is to allocate jobs with lower power consumption profiles during on-peak time and to allocate jobs with higher power consumption profiles during off-peak time. Furthermore, the allocation is made under the assumption that there will be no impact to system utilization, meaning that a situation where a job is waiting in the queue while there is a sufficient amount of idle/available computing nodes is not allowed.

Getting job power consumption profiles is feasible on today's supercomputers. Most production HPC systems are deployed with built-in sensors that monitor the health status of its hardware components. These sensors, deployed in various locations inside the system, report environmental conditions such as motherboard, CPU, GPU, hard disk and other peripherals for temperature, voltage, and/or fan speed. A number of software tools/interfaces are publicly available for users to access these sensor readings[6] [2] [4] [28].

Figure 2 pictorially illustrates an example to highlight the key idea of our design as compared to the conventional FCFS. Suppose five jobs J_0, J_1, J_2, J_3, J_4 are submitted to a 12-node system. Each job is associated with several parameters, such as the amount of nodes needed, the estimated runtime, etc. Further, each job is also associated with a power consumption profile p_i, which can be determined from historical data[28]. Suppose these jobs have the following parameters:

Job	Power Profile (W/node)	Job Size
J_0	50	6
J_1	20	3
J_2	40	3
J_3	30	3
J_4	10	6

Under the conventional FCFS policy, the scheduling sequence is always $< J_0, J_1, J_2 >$ in spite of the scheduling time. Our scheduling mechanism provides different scheduling sequences depending on the dynamic electricity price and job's power profile. More specifically, our scheduling algorithm allocates $< J_4, J_1, J_3 >$ during the on-peak period, and allocates $< J_0, J_2, J_3 >$ during the off-peak period. By comparing the total power consumptions of using FCFS to that of our design, it is clear that our design is able to reduce the accumulated power consumption during the on-peak time where as to increase the accumulated power consumption during the off-peak time, hence reducing the overall electricity bill.

4. METHODOLOGY

In Table 1, we list the nomenclature that will be used in the rest of this paper. Figure 3 gives an overview of our job

Table 1: Nomenclature

Symbol	Description
n_i	Job size, i.e., the number of nodes that are requested by Job i
p_i	Job power consumption profile, i.e., the average power consumption per node.
N_t	the amount of available nodes in the system at time t.
N	System size, i.e., the number of nodes in the system.
T	The time span from the start of the first job to the end of the last job.
w	Scheduling window size, i.e., the number of jobs in the window .

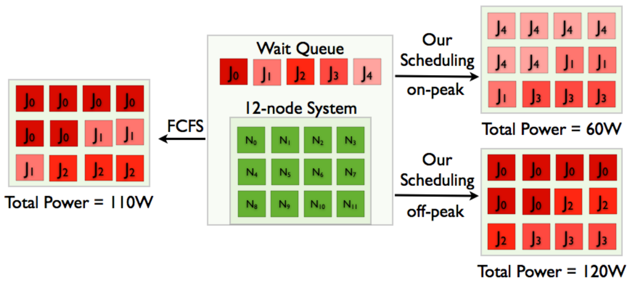

Figure 2: Job scheduling using FCFS(left) and our job power aware design at on-peak time(top right) and off-peak time (bottom right). For each job, its color represents its power profile, where dark color indicates power expensive and light color indicates power efficient.

power aware scheduling design. Our design contains two key techniques. One is the use of a scheduling window to take into consideration job features such as job fairness, and the other is the scheduling policy to balance energy usage and scheduling performance.

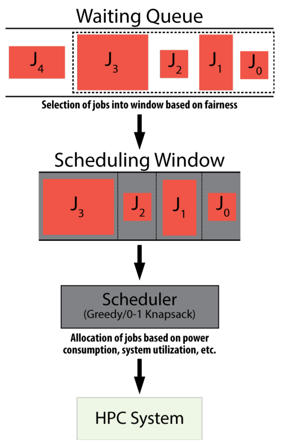
Waiting Queue
J4 J3 J2 J1 J0
Selection of jobs into window based on fairness
Scheduling Window
J3 J2 J1 J0
Scheduler
(Greedy/0-1 Knapsack)
Allocation of jobs based on power consumption, system utilization, etc.
HPC System

Figure 3: Overview of Job Power Aware Scheduling

4.1 Scheduling Window

Balancing fairness and system performance is always a big concern for a scheduling algorithm. The simplest way to ensure fairness and high system performance is to use a strict FCFS policy combined with backfilling, where jobs are started in the order of their arrivals [11].

In this work, rather than allocating jobs one by one from the front of the wait queue, we propose a novel window-based scheduling mechanism that allocates a window of jobs. The selection of jobs into the window is based on certain user-centric metrics, such as job fairness while the allocation of these jobs onto system resources is determined by certain system-centric metrics such as system utilization and energy consumption. By doing so, we are able to balance different metrics, representing both user satisfaction and system performance.

Given that FCFS is commonly used by production batch schedulers in HPC, we now describe how our window based scheduling works with FCFS. We maintain a scheduling window in front of the job queue, and the submitted jobs enter the job queue first and then into the scheduling window. The selection of jobs is based on job arrival times, thereby guaranteeing job fairness; the allocation of the jobs from the window to the available system nodes is based on job power profiles, which will be described later.

Typically, the window size should be determined based on system workload such that a large window is preferred in case of high workload. For typical workloads at production supercomputers, we find that a window size from 10 to 30 jobs can achieve reasonable electricity bill savings.

4.2 Job Power Aware Scheduling Policies

In this work, we develop two power aware scheduling policies. The first is a Greedy policy, where jobs are allocated entirely based on the values of their power profiles. The second is a 0-1 Knapsack based policy, where both job power profile and system utilization are taken into consideration during decision making.

4.2.1 *Greedy Policy*

All the jobs in the scheduling window are first sorted by their power profiles. During the on-peak electricity price period, all the jobs in the scheduling window are sorted in a decreasing order based on their power profiles; conversely, they are sorted in an increasing order during the off-peak period. After the sorting, the scheduler will dispatches the ordered jobs out of the scheduling window.

Greedy policy is simple and fast. Suppose the number of jobs in the window size is n, then the complexity of the algorithm is $O(nlgn)$.

4.2.2 *0-1 Knapsack based Policy*

The only difference between on-peak and off-peak scheduling selection is the aggregated power consumption: during the on-peak period, the goal is to minimize the value; during the off-peak period, the goal is to maximize the value.

In the following, we present the 0-1 Knapsack based policy works at off-peak time.

Suppose there are N_t available nodes in the system, the scheduling window size is w $\{J_i, |1 \leqslant i \leqslant w\}$, and each job J_i requires n_i nodes, with a power profile of p_i. Now, the scheduling problem can be formalized as follows:

Problem 1. To selecting a subset of $\{J_i, |1 \leqslant i \leqslant k\}$ from the scheduling window such that the aggregate nodes $\sum_{1 \leq i \leq k} n_i$ is no more than N_t, with the objective of maximizing the aggregated power consumption $\sum_{1 \leq i \leq k} n_i \cdot p_i$.

The above problem can be formalized into a 0-1 Knapsack problem. We set the available nodes N_t as the knapsack's size and consider the jobs in the scheduling window as the objects that we intend to put into the knapsack. For each job, its power profile (measured in W/node or kW/rack) is its value, the number of required node is considered as the weight. Hence we can further transform Problem 1 into a standard 0-1 Knapsack model.

Problem 2. To determine a binary vector $X = \{x_i, |1 \leqslant i \leqslant k\}$ such that:

$$\begin{aligned} &\text{maximize} \sum_{1 \leq i \leq k} x_i \cdot p_i, \quad x_i = 0 \text{ or } 1 \\ &\text{subject to} \sum_{1 \leq i \leq k} x_i \cdot n_i \leq N_t \end{aligned} \tag{1}$$

The standard 0-1 Knapsack model can be solved in pseudo-polynomial time by using dynamic programming[8]. To avoid redundant computation, when implementing this algorithm we use the tabular approach by defining a 2D table G, where $G[k, w]$ denotes the maximum gain value that can be achieved by scheduling jobs $\{j_i | 1 \leq i \leq k\}$ which require no more than N_t computing nodes, where $1 \leq k \leq J$. $G[k, w]$ has the following recursive feature:

$$G[k,w] = \begin{cases} 0 & kw = 0 \\ G[k-1,w] & w_i \geq w \\ max(G[k-1,w], v_i + G[k-1, w-w_i]) & w_i \leq w \end{cases} \tag{2}$$

The solution $G[J, N_t]$ and its corresponding binary vector X determine the selection of jobs scheduled to run. The computation complexity of Equation 2 is $O(J \cdot N_t)$.

During on-peak time, 0-1 Knapsack-based policy is modified by changing the selection criterion into minimizing the total value of the objects in the knapsack with the constraint of knapsack size.

5. EVALUATION METHODOLOGY

We conduct a series of experiments using trace-based simulations. In our experiments, we compare our design as against the well-known FCFS scheduling policy [11]. In the rest of the paper, we simply use Greedy, Knapsack, and FCFS to denote our scheduling policies and the conventional batch scheduling policy. This section describes our evaluation methodology, and the experimental results will be presented in the next section.

5.1 CQSim: Trace-based Scheduling Simulator

Simulation is an integral part of our evaluation of various scheduling policies as well as their aggregate effect on performance and power consumption. We have developed a simulator named CQSim to evaluate our design at scale. The simulator is written in Python, and is formed by several modules such as job module, node module, scheduling policy module, etc. Each module is implemented as a class. The design principles are reusability, extensibility, and efficiency. The simulator takes job events from a trace, and an event may be job submission, job start, job end, and other events. Based on these events, the simulator emulates job submission, allocation, and execution based on specific scheduling policy. CQsim is open source, and is available to the community [1].

5.2 Job Traces

In this work, we use two real workload traces collected from production supercomputers to evaluate our design. The objective of using multiple traces is to quantify the impact of different factors on electricity bill saving. The first trace we used is from a machine named Blue Horizon at the San Diego Supercomputer Center (denoted as SDSC-BLUE in the paper), which ran 14,4830 jobs in 2001.

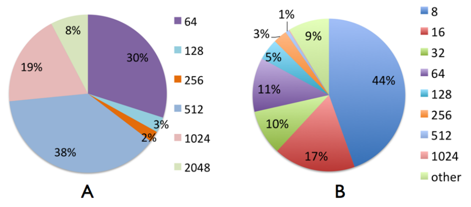

Figure 4: (A) Job size distribution of ANL-BGP, (B) Job size distribution of SDSC-BLUE

The second trace we used is from two racks of the IBM Blue Gene/P machine named Intrepid at Argonne (denoted as ANL-BGP in the paper)[25] [29]. This trace contains 26,012 jobs. Since this trace is extracted out of the original 40-rack workload, the utilization rate is relatively low. A well-known approach to remedy this problem is to decrease job arrival intervals by a certain rate [27]. After we decrease job arrival intervals by 40%, the trace becomes 5-month long with the utilization rate ranging between 39% and 88%. Figure 4 summarize job size distribution of these traces. ANL-BGP is used to represent capability computing where the computing power is explored to solve larger problems, whereas SDSC-BLUE is used to represent capacity computing where the computing power is utilized to solve a large number of small problems.

5.3 Dynamic Electricity Price

In our experiments, we set two different electricity prices: on-peak and off-peak pricing. We set the price in on-peak time (from 12pm to 12am) higher than the off-peak time (from 12am until 12pm). This is done to simplify our calculation and statistical analysis. Indeed, we are not concerned about the absolute value of electricity price; instead the ratio of on-peak price to off-peak price is more important. According to [23], the most common ratio of on-peak and off-peak pricing varies from 1:2 to 1:5. Hence we set the default ratio to 1:3.

5.4 Job Power Profile

Since job power profile is not included in the original traces, we assign each job with a power profile between 20 to 60W per node using a normal distribution according to the power profile presented in Figure 1. Similarly, we are not concerned about the absolute power profile value; instead the ratio of maximum power profile to minimal power profile is more important. The default ratio is set to 1:3.

5.5 Evaluation Metrics

In this work, we use three metrics to evaluate our design against the conventional FCFS.

Electricity Bill Saving. We calculate the relative difference between the electricity bill using our design and FCFS to measure the electricity bill savings achieved by our design. The simulator sums up electricity bill on a daily basis for the calculation of this metric.

System Utilization Rate. This metric denotes the ratio of the node-hours that are used for useful computation to the elapsed system node-hours. Specifically, let T be the total elapsed time for J jobs, c_i be the completion time for job i and s_i be its the start time, and n_i be the size of job i, then system utilization rate is calculated as

$$\frac{\sum_{0\leq i\leq J}(c_i - s_i)\cdot n_i}{N\cdot T} \tag{3}$$

Average Job Wait Time. For each job, its wait time refers to the time elapsed between the moment it is submitted to the moment it is allocated to run. This metric is calculated as the average across all the jobs submitted to the system. This metric is a user-centric metric, measuring scheduling performance from user's perspective.

6. EXPERIMENT RESULTS

We conduct four sets of experiments on the traces described in Section 5.2 to evaluate our design as against FCFS.

6.1 Baseline Results

Baseline results are presented in Figures 5 - 10, where we use the default setting described in Sections 5.3-5.4, meaning that job power profile ratio is 1:3 and the off-peak/on-peak pricing ratio is 1:3. On production supercomputers, the scheduling frequency is typically on the order of 10 to 30 seconds. Hence, the simulator is set to make a scheduling decision every 10 seconds.

Since our evaluation focuses on the relative reduction of electricity bills, so the absolute value of idle power consumption, which is set to 0 in our experiments, does not impact the results.

Figure 5 and Figure 6 compare system utilization rates achieved by using different job scheduling policies. It is clear that the utilization degradation introduced by our design is always less than 5%, no matter which scheduling policy we choose (Greedy or Knapsack). Moreover, for the 3rd and 5th month of SDSC-BLUE trace, both our scheduling policies may achieve higher system utilization compared to FCFS. These results clearly demonstrate that our scheduling design brings negligible impact on system utilization, which is critical to HPC systems.

Figure 7 and Figure 8 present electricity bill savings obtained by our designs as against FCFS. In general, the monthly electricity bill saving ranges from 0.5% to 10% by using Greedy, and it is from 2% to 10% by using Knapsack. The average electricity bill saving is 3.16%-5.53%. We also make two interesting observations. First is that Greedy achieves more electricity bill saving on SDSC-Blue, whereas Knapsack brings in more cost saving on ANL-BGP. Second, we can see that more energy saving is obtained from ANL-BGP. As we can see from Figure 4, these traces have distinctive job characteristics in term of job size. In ANL-BGP trace, 38% jobs request 512 nodes, 19% request 1024 nodes and 8% request 2048 nodes. Given the system size is 2,048, this means 65% jobs are relatively large in the sense that these

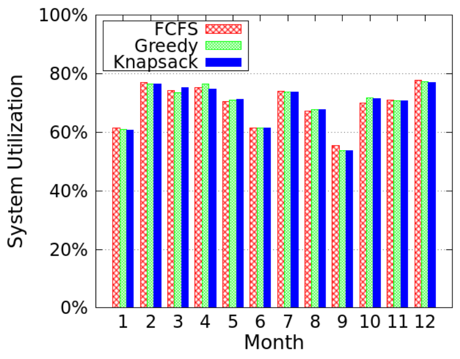

Figure 5: System Utilization of SDSC-BLUE

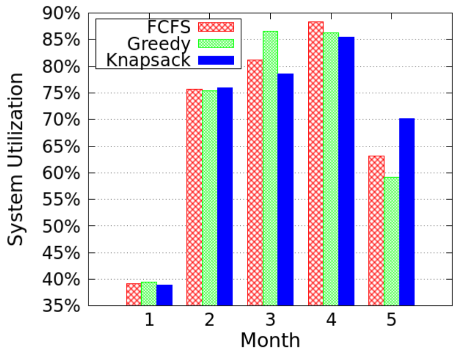

Figure 6: System Utilization of ANL-BGP

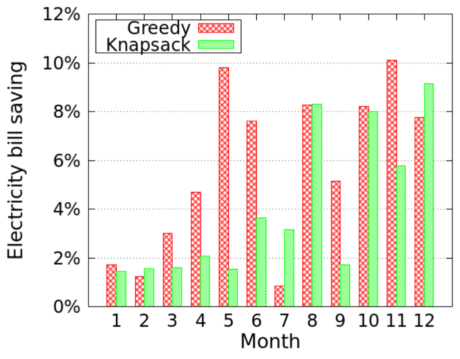

Figure 7: Electricity bill saving obtained on SDSC-BLUE using Greedy and Knapsack scheduling compared with FCFS. The average electricity bill saving obtained by using Greedy and Knapsack scheduling policy are 4.33% and 3.16% respectively.

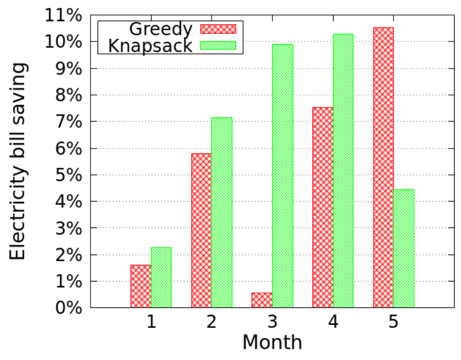

Figure 8: Electricity bill saving obtained on ANL-BGP using Greedy and Knapsack scheduling compared with FCFS. The average electricity bill saving obtained by using Greedy and Knapsack scheduling policy are 5.06% and 5.53% respectively.

jobs request more than a quarter of the system resources. In contrast, SDSC-BLUE has different characteristics, most jobs are relatively small: 71% of the jobs smaller than 32, whereas the system size is 1,152. In other words, ANL-BGP represents big capability computing and SDSC-BLUE represents small capability computing. The results indicate that our design provides more benefits for big capability computing.

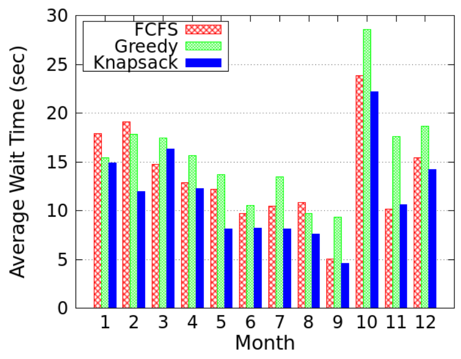

Figure 9: Job Wait Time of SDSC-BLUE

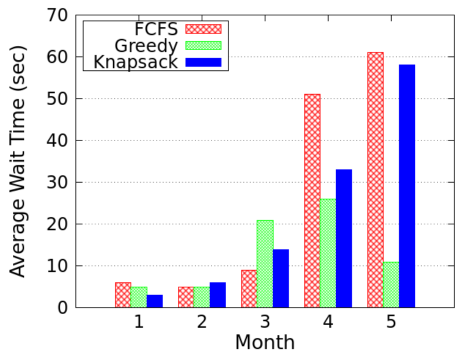

Figure 10: Job Wait Time of ANL-BGP

Figure 9 and Figure 10 show the average job wait times introduced by our design and FCFS. In general, job wait time is influenced by many factors, such as job arrival rate, job size, etc. Hence, the average job wait time varies from month to month. While our scheduling policies might impact the average job wait time, on both traces we observe that the maximum change on this metric caused by our design is less than 10 seconds as compared to FCFS. This implies that our design does not degrade the scheduling performance from user's perspective as compared to FCFS.

Due to space limitations, we will omit the presentation of job wait time in the following experiments.

6.2 Impacts of Electricity Prices and Job Power Profiles

In this set of experiments, we conduct a sensitivity study to investigate the amount of electricity bill savings that could be achieved by our design under different combinations of power and pricing ratios. We set three different job power consumption profiles, namely 1:2 (e.g., 20W per node as the lowest profile and 40W per node as the highest profile,) 1:3, and 1:4. We also set three off-peak versus on-peak pricing ratio (i.e., 1:3, 1:4, and 1:5). The results are summarized in Table 2 and 3.

Table 2 and 3 present the electricity bill savings obtained by our scheduling policies on ANL-BGP and SDSC-BLUE respectively, under different pricing and power profile combinations. As the job power profile ratio increases, so does the electricity bill saving obtained by both Greedy and Knapsack. The same situation happens as the pricing ratio goes up. The highest electricity bill saving is achieved in the case of when the job power profile ratio is set to 1:4 and the off-peak over on-peak pricing is set to 1:5.

This is quite reasonable, because the greater the job power profile ratio is, the more power consumption saving our design can obtain. With higher off-peak/on-peak price ratio, the same amount power consumption saving can yield more electricity bill saving.

Table 2: Electricity bill savings obtained by our scheduling policies on ANL-BGP. In each cell, the top number is the electricity bill saving obtained by Greedy and the bottom number is the electricity bill saving obtained by Knapsack

	Pricing Ratio		
Power Ratio	1:3	1:4	1:5
1:2	3.54% 4.18%	4.33% 5.07%	4.79% 5.64%
1:3	5.06% 5.35%	6.13% 6.48%	6.85% 7.25%
1:4	6.27% 7.21%	7.58% 8.52%	8.40% 9.86%

From both Table 2 and Table 3, we can observe that for the ANL-BGP trace, Knapsack outperforms Greedy under all power and pricing ratio combinations and for SDSC-BLUE trace the situation is just opposite. As mentioned earlier, the ANL-BGP trace contains a large percentage of large jobs. During on-peak period, the Greedy policy always selects a job with the least power profile, whereas the Knapsack policy often picks out a job with a small power

Table 3: Electricity bill savings obtained by our scheduling policies on SDSC-BLUE. In each cell, the top number is the electricity bill saving obtained by Greedy and the bottom number is the electricity bill saving obtained by Knapsack

	Pricing Ratio		
Power Ratio	1:3	1:4	1:5
1:2	3.84%	4.84%	6.19%
	2.39%	3.01%	3.85%
1:3	4.33%	5.46%	6.98%
	3.16%	3.98%	5.10%
1:4	5.55%	6.98%	8.95%
	3.05%	3.84%	4.92%

profile under the constraint of the available system resources. While the Knapsack policy may not schedule the job with the least power profile, it is capable of identifying the job which consumes the least amount of aggregated power consumption on all the nodes.

6.3 Impact of Scheduling Frequencies

Typically batch schedulers make allocation decisions periodically. On production supercomputers, the scheduling frequency is generally on the order of 10 to 30 seconds. Hence, in this set of experiments, we evaluate the impact of different scheduling intervals (i.e., 10 seconds, 20 seconds, and 30 seconds) on the amount of electricity bill savings.

Table 4 shows the average electricity bill savings obtained by our design compared to the conventional FCFS with three different scheduling periods on ANL-BGP and SDSC-BLUE. As we can see that the longer the scheduling period is, the more electricity bill savings our design can get. This is because with a relative high job arrival rate, a longer scheduling period means more system nodes can be accumulated for job allocation at a time. As such, our design is able to allocate more low power profiled jobs during on-peak period or more high power profiled jobs during off-peak period, resulting in more electricity bill savings.

Table 4: Electricity bill Savings obtained by our scheduling policies under different scheduling frequencies. In each cell, the top number is on ANL-BGP and the bottom number is on SDSC-BLUE.

	Scheduling Policy	
Frequency	Greedy	Knapsack
10-Second	7.49%	7.13%
	4.33%	3.16%
20-Second	10.07%	8.91%
	9.70%	9.80%
30-Second	17.52%	22.43%
	19.69%	23.07%

Table 5 shows the average system utilization of ANL-BGP and SDSC-BLUE trace under different scheduling frequencies by our design and FCFS. As we can see that both Greedy and 0-1 Knapsack scheduling policy employed in our design have almost no impact on the system utilization rate when the scheduling period is 10 seconds. When the scheduling period is increased to 30 seconds, some available nodes will have to wait for a relatively long time until they are assigned to new jobs. Thus the system utilization rate suffers slightly, however is always less than 3%. Combining the results shown in Table IV and V, it is observed that a longer scheduling frequency can bring in more electricity bill savings, at a cost of a slightly degraded system utilization (less than 3%).

Table 5: System utilization rate under different scheduling frequencies. In each cell, the top number is on ANL-BGP and the bottom number is on SDSC-BLUE.

	Scheduling Policy		
Frequency	FCFS	Greedy	Knapsack
10-Second	70.0%	69.70%	69.07%
	69.59%	69.53%	69.50%
20-Second	68.56%	69.03%	65.97%
	68.56%	69.25%	65.06%
30-Second	63.77%	60.42%	60.84%
	67.38%	68.85%	66.21%

6.4 Impact of Scheduling Window

The use of scheduling window, rather than one by one job scheduling adopted by the conventional batch scheduling, is a key technique of our design. Typically, an optimal window size is influenced by many factors, in particular job arrival rate. In general, a larger window means more opportunities for our design to make an optimal decision. However, large window size can result in high scheduling overhead, especially Knapsack policy, since window size is a dominant factor of its computational complexity.

We conduct a sensitivity study of scheduling window by varying its size from 10 to 200. For both traces, our results show that the variations of all three metrics are not substantial (e.g., within 5%). More importantly, our results indicate that when the window size is set to 10 to 30, for both traces, the variations of all three metrics are negligible. Given the high computation overhead introduced by large window size, a window size of 10-30 jobs is preferable for typical workload at production systems.

6.5 Result Summary

In summary, our trace-based experiments have shown the following:

- Workload characteristics can impact the performance of our design in terms of electricity bill savings. In particular, our design can achieve more savings on big capability computing systems than on small capacity computing systems.

- Both the Knapsack policy and the Greedy policy are capable of reducing electricity bill with little or no impact to system utilization, as compared to FCFS.

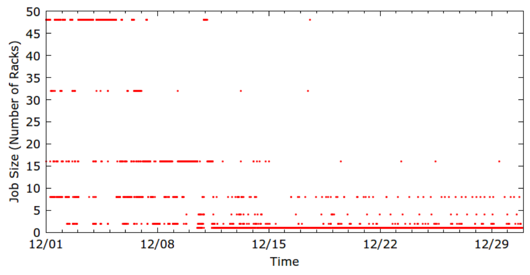

Figure 11: Job characteristics in December of 2012 on the 48-rack Mira machine. Each red point indicates a job submission

Further, the Knapsack policy seems to outperform the Greedy policy for capability computing.

- The amount of electricity bill savings is also influenced by scheduling frequency. The longer the scheduling frequency is, the more electricity bill savings is achieved by our job power profile aware scheduling.
- Higher power profile ratio and electricity pricing ration lead to greater electricity bill savings by using our design.
- For typical workload at production systems, a scheduling window of 10-30 jobs is sufficient.

7. CASE STUDY

In this section, we present a case study of using our job power aware scheduler on Mira. We collected the job trace from the machine in December 2012. During the month, the first half of the month were jobs for acceptance testing (hence most jobs are large jobs) and the second half was used for early science applications from users (hence most jobs are small sized such as single rack). There are totally 3,333 jobs executed on the machine during the month. A summary of these jobs is described in Figure 11. For each job, its power profile is extracted from the environmental log [28], and the distribution of job power profiles is presented in Figure 1.

In this case study, we compare the Knapsack policy to FCFS. We apply two scheduling frequencies, i.e. 10-second and 30-second. The scheduling window is set to 10-second.

Figure 12 presents the average utilization within a day. Here system utilization at each time point is calculated as the average over the month. During the off-peak time, system utilization achieved by our scheduler is higher than that achieved by FCFS. This is because during the off-peak time, our design attempts to allocate jobs with high power profiles, as many as possible, by taking advantage of low electricity pricing. During the on-peak time, FCFS achieves a slightly better system utilization over our design. This is because our design intends to schedule large jobs with low power profiles, leaving some idle nodes that are not sufficient for other jobs. Nevertheless, despite some minor variation at any time instant, the daily system utilization is not impacted or degraded by using our design.

Figure 13 presents the average power consumption within a day. Here power consumption at each time point is calculated as the average over the month. During the off-peak time, the amount of power consumed by our design is higher than that consumed by FCFS. This phenomenon is more obvious when we switch the scheduling frequency from 10 seconds to 30 seconds. This is reasonable as our design aims to increase power consumption by taking advantage of low electricity pricing during off-peak time. During the on-peak time,while our design is supposed to decrease the overall power consumption to avoid high electricity cost, the figure doesn't show such a pattern. As we mentioned earlier, the job trace was collected in the month in which the second half of the month was mainly used for early science testing (i.e., most jobs submitted are small sized such as single rack). As presented in Figure 1, small sized jobs have similar power profiles. Due to these unique characteristics of the job trace (i.e., the same sized jobs with similar power profiles), our design ends up with the same scheduling sequence as FCFS, hence giving the similar power consumption during the on-peak time. We believe our design is capable of providing more electricity bill saving when the machine is used in production.

The monthly electricity bill saving obtained by our design versus FCFS is 5.4% and 9.98% respectively by using 10-second scheduling frequency and 30-second frequency. This is substantial, given that approximately $1M annual electricity bill to power up this machine at Argonne.

8. CONCLUSIONS

In this paper, we have proposed an novel job power aware scheduling design, with the objective to reduce the electricity bill of HPC systems. Our design is based on the facts that HPC jobs have different individual power profiles and that electricity prices vary throughout a day. By scheduling jobs with high power profiles during low electricity pricing period and jobs with low power profiles during high electricity pricing period, our scheduler is capable of cutting the electricity bill of HPC systems by up to 23% without impacting system utilization, which is critical to HPC systems.

To our knowledge, this is the first electricity bill study of large-scale HPC systems using real job traces and job power profiles from production systems. Our key findings and contributions are: (1) a job power aware scheduling mechanism and two scheduling policies designed to cut the electricity bill of HPC systems; (2) a scheduling policy with real potential to substantially reduce the electricity bill for HPC systems by exploring distinct job power profiles and varying daily electricity prices; and (3) a trace-based scheduling simulator named CQSim for evaluating various scheduling policies at scale, which is available online.

Some avenues for future work include more experiments with real job traces and job power profiles from various HPC systems as well as integrating our design with the work on environmental data analysis tool as [28] for automatically obtaining job power profiles.

9. ACKNOWLEDGMENTS

We appreciate the valuable comments and suggestions from the anonymous reviewers. We would like to thank our paper shepherd, Dr.Taisuke Boku, for his time and guidance. This work at Illinois Institute of Technology was supported in part by US National Science Foundation grants CNS-0834514 and OCI-0904670. The work at Argonne is supported by the Office of Science of the U.S. Department

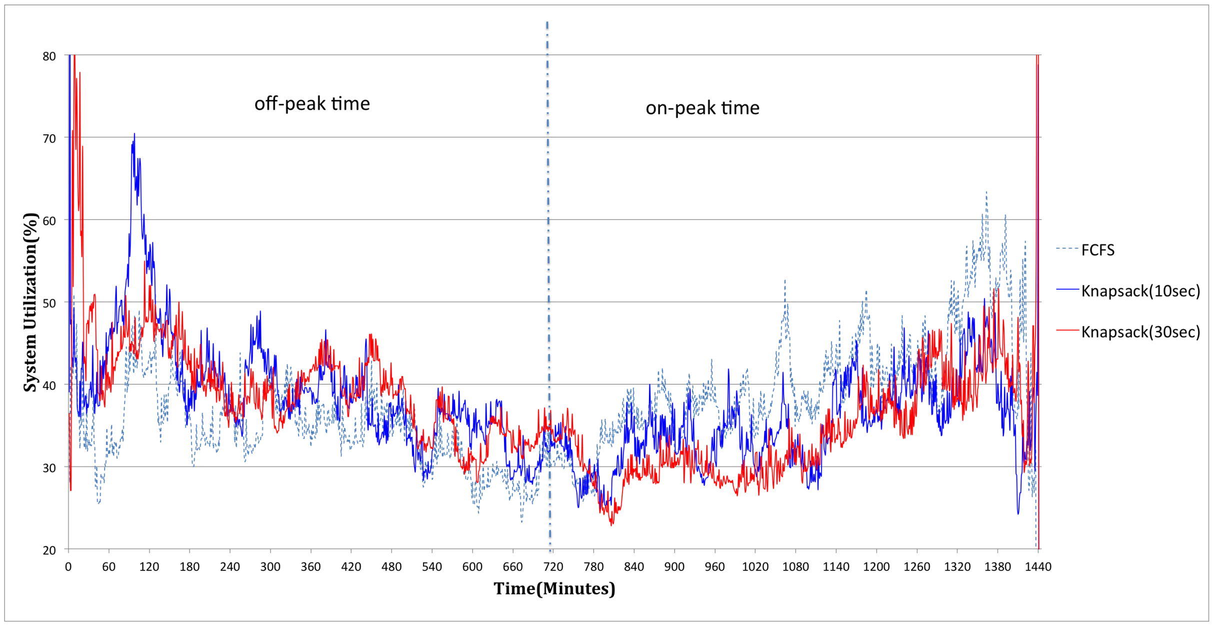

Figure 12: The average daily system utilization

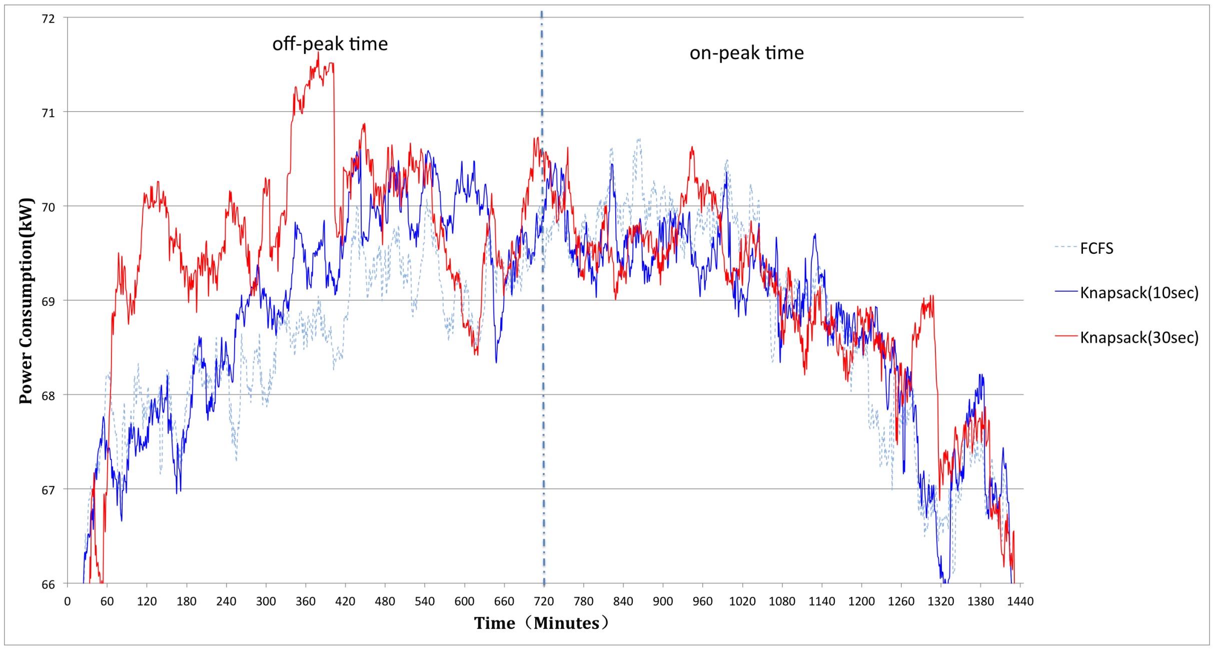

Figure 13: The average daily power consumption

of Energy under contract DEAC02-06CH11357.

10. REFERENCES

[1] Cqsim: An event-driven simultor. http://bluesky.cs.iit.edu/cqsim

[2] IBM redbooks publication, IBM system Blue Gene solution: Blue Gene/Q system administration.

[3] Parallel Workload Archive. http://www.cs.huji.ac.il/labs/parallel/workload

[4] Managing System Software for Cray XE and Cray XT Systems. Cray Document. 2012.

[5] L. Barroso and U. Holzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[6] J. Brandt, A. Gentile, D. J. Hale, and P. Pebay. OVIS: a tool for intelligent, real-time monitoring of computational clusters. In *International Symposium on Parallel and Distributed Processing*, 2006.

[7] Z. Cao, L. T. Watson, K. W. Cameron, and R. Ge. A power aware study for VTDIRECT95 using DVFS. In *Proceedings of the Spring Simulation Multiconference*, SpringSim '09. Society for Computer Simulation International, 2009.

[8] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2001.

[9] M. Etinski, J. Corbalan, J. Labarta, and M. Valero. Parallel job scheduling for power constrained HPC systems. *Parallel Comput.*, 38(12):615–630, 2012.

[10] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *34th ACM Annual International Symposium on Computer Architecture*, 2007.

[11] D. Feitelson and A. Weil. Utilization and predictability in scheduling the IBM SP2 with backfilling. In *International Parallel and Distributed Processing Symposium*, 1998.

[12] C. Garciia-Martos, J. Rodriiguez, and M. Sanchez. Forecasting electricity prices by extracting dynamic common factors: application to the Iberian market. *Generation, Transmission Distribution, IET*, 6(1):11–20, 2012.

[13] M. Hennecke, W. Frings, W. Homberg, A. Zitz, M. Knobloch, and H. Bottiger. Measuring power consumption on IBM Blue Gene/P. *Computer Science - Research and Development*, 27:329–336, 2012.

[14] J. Hikita, A. Hirano, and H. Nakashima. Saving 200kw and $200 k/year by power-aware job/machine scheduling. In *IEEE International Symposium on Parallel and Distributed Processing*, 2008.

[15] I. Koichiro, I. Takanori, and T. Makoto. Using dynamic electricity pricing to address energy crises evidence from randomized field experiments. 2013. http://www.stanford.edu/ itok/ItoIda-Tanaka-Dynamic-Pricing.pdf.

[16] E. Lee, I. Kulkarni, D. Pompili, and M. Parashar. Proactive thermal management in green datacenters. *The Journal of Supercomputing*, 60:165–195, 2012.

[17] C. Lefurgy, X. Wang, and M. Ware. Power capping: a prelude to power shifting. *Cluster Computing*, 11(2):183–195, June 2008.

[18] P. Li. Variational analysis of large power grids by exploring statistical sampling sharing and spatial locality. In *IEEE/ACM International Conference on Computer-Aided Design*, 2005.

[19] Y. Liu and H. Zhu. A survey of the research on power management techniques for high-performance systems. *Softw. Pract. Exper.*, 40(11):943–964, 2010.

[20] J. Lundgren, J. Hellstrom, and N. Rudholm. Multinational electricity market integration and electricity price dynamics. In *5th International Conference on European Electricity Market*, 2008.

[21] C. Patel, R. Sharma, C. Bash, and S. Graupner. Energy aware grid: Global workload placement based on energy efficiency. In *HPL Technical Report*, November 2002.

[22] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath. Load balancing and unbalancing for power and performance in cluster-based systems, 2001.

[23] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs. Cutting the electric bill for internet-scale systems. *SIGCOMM Comput. Commun. Rev.*, 39(4):123–134, Aug. 2009.

[24] W. Tang, N. Desai, D. Buettner, and Z. Lan. Analyzing and adjusting user runtime estimates to improve job scheduling on the Blue Gene/P. In *2010 IEEE International Symposium on Parallel Distributed Processing*, 2010.

[25] W. Tang, N. Desai, D. Buteener, and Z. Lan. Job scheduling with adjusted runtime estimates on production supercomputers. *Journal of Parallel and Distributed Computing (JPDC)*, 73(7):926–938, 2013.

[26] W. Tang, Z. Lan, N. Desai, D. Buettner, and Y. Yu. Reducing fragmentation on torus-connected supercomputers. In *2011 IEEE International Symposium on Parallel Distributed Processing Symposium*, 2011.

[27] D. Tsafrir, K. Ouaknine, and D. Feitelson. Reducing performance evaluation sensitivity and variability by input shaking. In *15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, 2007.

[28] S. Wallace, V. Vishwanath, S. Coghlan, Z. Lan, and M. Papka. Measuring power consumption on IBM Blue Gene/Q. In *The Ninth Workshop on High-Performance, Power-Aware Computing*, 2013.

[29] Z. Zheng, L. Yu, W. Tang, Z. Lan, R. Gupta, N. Desai, S. Coghlan, and D. Buettner. Co-analysis of RAS log and job log on Blue Gene/P. In *2011 IEEE International Symposium on Parallel Distributed Processing Symposium*, pages 840–851, 2011.

[30] Z. Zhou, Z. Lan, W. Tang, and N. Desai. Reducing energy costs for IBM Blue Gene/P via Power-Aware job scheduling. In *17th Workshop on Job Scheduling Strategies for Parallel Processing*, 2013.