Reducing Energy Costs for IBM Blue Gene/P via Power-Aware Job Scheduling

Zhou Zhou[1], Zhiling Lan[1], Wei Tang[2], and Narayan Desai[2]

[1] Department of Computer Science
Illinois Institute of Technology
Chicago, Illinois, United States
{zzhou1,lan}@iit.edu

[2] Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, Illinois, United States
{wtang,desai}@mcs.anl.gov

Abstract. Energy expense is becoming increasingly dominant in the operating costs of high-performance computing (HPC) systems. At the same time, electricity prices vary significantly at different times of the day. Furthermore, job power profiles also differ greatly, especially on HPC systems. In this paper, we propose a smart, power-aware job scheduling approach for HPC systems based on variable energy prices and job power profiles. In particular, we propose a 0-1 knapsack model and demonstrate its flexibility and effectiveness for scheduling jobs, with the goal of reducing energy cost and not degrading system utilization. We design scheduling strategies for Blue Gene/P, a typical partition-based system. Experiments with both synthetic data and real job traces from production systems show that our power-aware job scheduling approach can reduce the energy cost significantly, up to 25%, with only slight impact on system utilization.

Keywords: energy, power-aware job scheduling, resource management, Blue Gene, HPC system

1 Introduction

With the vast improvement in technology, we are now moving toward exascale computing. Many experts predict that exascale computers will have millions of nodes, billions of threads of execution, hundreds of petabytes of inner memory, and exabytes of persistent storage [1]. Exascale computers will have unprecedented scale and architectural complexity different from the petascale systems we have now. Hence, many challenges are expected to emerge during the transition to exascale computing. Four major challenges—power, storage, concurrency, and reliability—are identified where current trends in technology are insufficient and disruptive technical breakthroughs will be needed to make exascale computing a reality [2]. In particular, the energy and power challenge is pervasive, affecting every part of a system. Today's leading-edge petascale systems consume between 2 and 3 megawatts per petaflop [2]. It is generally accepted that an exaflop system should consume no more than 20 MW; otherwise their operating costs would be prohibitively expensive.

High-performance computing generally requires a large amount of electricity to operate computer resources and to cool the machine room. For example, a high-performance computing (HPC) center with 1,000 racks and about 25,000 square feet requires 10 MW of energy for the computing infrastructure and an additional 5 MW to remove the dissipated heat [3]. At the Argonne Leadership Computing Facility (ALCF), our systems consume approximately $1 million worth of electricity annually. As of 2006, the data centers in the United States were consuming 61.4 billion kWh per year [4], an amount of energy equivalent to that consumed by the entire transportation manufacturing industry (the industry that makes airplanes, ships, cars, trucks, and other means of transportation) [5]. Since the cost of powering HPC systems has been steadily rising with growing performance, while the cost of hardware has remained relatively stable, it is argued that if these trends were to continue, the energy cost of a large-scale system during its lifetime could surpass the equipment itself [6].

Several conventional approaches to reducing energy cost have been adopted by organizations operating HPC systems. For instance, a popular and intuitive strategy is to manipulate the nodes within an HPC system through techniques such as dynamic voltage and frequency scaling, power state transitions, and the use of separation in hot and cold aisles. Meanwhile, new cooling technologies, load-balancing algorithms, and location-aware computing have been proposed as new ways to reduce the energy demand of HPC centers [7].

In this paper we develop and analyze a new method to reduce the energy cost of operating large-scale HPC systems. Our method relies on three key observations.

1. *Electricity prices vary.* In many districts in the United States with wholesale electricity markets, the price varies on an hourly basis. Sometimes the variation can be significant as much as a factor of 10 from one hour to the next [7]. HPC centers often make a contract with the power companies to pay variable electricity prices. A common arrangement is that HPC centers pay less for electricity consumed during an off-peak period (nighttime) than during an on-peak period (daytime)[4].
2. *Job power consumption differs.* Studies have shown that most HPC jobs have distinct power consumption profiles. For example, in [8] the authors analyzed the energy characteristics of the production workload at the Research Center Juelich (FZJ) and found that their jobs have a power consumption ranging from 20 kW to 33 kW per rack on their Blue Gene/P system. Usually, an application has relatively high power consumption during its computational phases, and its power consumption drops during the communication or I/O phase [8]. For example, I/O-intensive jobs and computation-intensive jobs have totally different energy-consuming behaviors leading to variation in their power consumption.

3. *System utilization cannot be impacted in HPC.* Most conventional power saving approaches focus on manipulating nodes by turning off some nodes or putting the system into an idle phase during the peak price time. An immediate consequence of these approaches is that they lower system utilization [4]. Lowering system utilization for energy saving is not tolerable for HPC centers, however. HPC systems require a significant capital investment; and hence making efficient use of expensive resources is of paramount importance to HPC centers. Unlike Internet data centers that typically run at about 10–15% utilization, systems at HPC centers have a typical utilization of 50–80% [9], and job queues are rarely empty because of the insatiable demand in science and engineering. Therefore, an approach is needed that can save energy cost while maintaining relatively high system utilization.

We argue that HPC systems can save a considerable amount of electric costs by adopting an intelligent scheduling policy that utilizes variable electricity prices and distinct job power profiles—without reducing system utilization. More specifically, we develop a power-aware scheduling mechanism that smartly selects and allocates jobs based on their power profiles by preferentially allocating the jobs with high power consumption demands during the off-peak electricity price period. Our design is built on three key techniques: a scheduling window, a 0-1 knapsack model, and an on-line scheduling algorithm. The scheduling window is used to balance different scheduling goals such as performance and fairness; rather than allocating jobs one by one from the wait queue, our scheduler makes decisions on a group of jobs selected from the waiting queue. We formalize our scheduling problem into a standard 0-1 knapsack model, based on which we apply dynamic programming to efficiently solve the scheduling problem. The derived 0-1 knapsack model enables us to reduce energy cost during high electricity pricing period with no or limited impact to system utilization. We use our on-line scheduling algorithm together with the scheduling window and 0-1 knapsack model to schedule jobs on Blue Gene/P.

By means of trace-based simulations using real job traces of the 40-rack Blue Gene/P system at Argonne, we target how much cost saving can be achieved with this smart power-aware scheduling. One major advantage of using real job traces from production systems of different architectures is to ensure that experimental results can reflect the actual system performance to the greatest extent. Experimental results show that our scheduling approach can reduce energy bills by 25% with no or slight loss of system utilization and scheduling fairness. We also perform a detailed analysis to provide insight into the correlation between power and system utilization rate, comparing our power-aware scheduling with the default no-power-aware scheduling. We also conduct a sensitivity study to explore how energy cost savings and utilization can be affected by applying different combinations of power ranges and pricing ratio.

The remainder of the paper is organized as follows. Section 2 discusses related studies on HPC energy issues. Section 3 describes our methodology, including the scheduling window, 0-1 knapsack model, and on-line scheduling algorithm.

Section 4 describes our experiments and results. Section 5 draws conclusion and presents future work.

2 Related Work

Research on power- or energy-aware HPC systems has been active in recent years. Broadly speaking, existing work has mainly focused on the following topics: hardware design, processor adjustment, computing nodes controlling and power capping.

Energy-efficient hardware is being developed so that the components consume energy more efficiently than do standard components [10]. Several researchers [11] and [12] argue that the power consumption of a machine should be proportional to its workload. According to [12]a machine should consume no power in idle state, almost no power when the workload is very light, and eventually more power when the workload is increased. However, power consumption does not strictly follow this because of the various job behaviors during runtime. In [8], the authors discuss these phenomena by presenting core power and memory power for a job on Blue Gene/P.

Dynamic voltage and frequency scaling (DVFS) is another widely used technique for controlling CPU power since the power consumption of processors occupies a substantial portion of the total system power (roughly 50% under load) [10]. DVFS enables a process to run at a lower frequency or voltage, increasing the job execution time in order to gain energy savings. Some research efforts on applying DVFS can be found in [13], [14], and [15]. Nevertheless, DVFS is not appropriate for some HPC systems. For example, DVFS is both less feasible and less important for the Blue Gene series because it does not include comparable infrastructure and already operates at highly optimized voltage and frequency ranges [8].

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. Some nodes are shut down or switched to a low-power state during the time of low system utilization [10, 16]. In [13] the authors designed an approach that can dynamically turn nodes on and off during running time. Jobs are concentrated onto fewer nodes, so that other idle nodes can be shut down to save energy consumption. Experiments on an 8-node cluster show about 19% energy savings. In their testbed, the time used to power on the server is 100 seconds, while the time to shut down is 45 seconds, causing approximately 20% degradation in performance.

Many large data centers use power capping to reduce the total power consumption. The data center administrator can set a threshold of power consumption to limit the actual power of the data center [6]. The total power consumption is kept under a predefined power budget so that an unexpected rise in power can be prevented. The approach also allows administrators to plan data centers more efficiently to avoid the risk of overloading existing power supplies. The idea of our scheduling borrows the idea of using power capping as a way to limit the power consumption of the system. However, our work is different from the conventional power capping approach in several aspects. First, we do not control the total power consumption through adjusting the frequency of CPU or power consumption of other components. Second, our goal is not to reduce the overall power consumption; instead, we aim at reducing energy cost by considering various job power ranges and dynamic electricity prices.

3 Methodology

In this section, we present the detailed scheduling methodology. As mentioned earlier, our scheduling design is built on three key techniques: scheduling window, 0-1 knapsack problem formulation, and on-line scheduling.

3.1 Problem Statement

User jobs are submitted to the system through a batch job scheduler. The job scheduler is responsible for allocating jobs in the wait queue to compute nodes. Before performing job scheduling, we have made two essential assumptions: (1) electricity prices keep changing during the day, with significant variation between low and high prices; (2) HPC jobs have different power profiles caused by different characteristics, which are available to the job scheduler at job submission. Also, our expected scheduling method should be based on the following design principles: (1) the scheduling method should save considerable energy cost by taking advantage of variable electricity prices and job power profiling, (2) there should be no or only minor impact to system utilization, and (3) job fairness should be preserved as much as possible. When electricity price is high (i.e., during the on-peak period), in order to save energy costs, we reduce the total amount of power consumed by the jobs allocated on the system. To limit the total power consumption, we set an upper bound denoted as the power budget that the system cannot exceed. Thus, our scheduling problem is as follows: How can we schedule jobs with different power profiles, without exceeding a predefined power budget and at the same time not affecting system utilization and not breaking the fairness of scheduling as much as possible?

Figure 1 illustrates a typical job-scheduling scenario on a 9-node cluster. Assume we have 5 jobs to be scheduled, which come in the order J1, J2, J3, J4, and J5. Job 1 stays at the head of the waiting queue and Job 5 at the tail. A 9-node atom cluster (one core per node) is ready to run these jobs. Each job is labeled with its requested number of nodes and total power consumption inside the rectangle. A job scheduler is responsible for dispatching jobs to its allocated nodes. Other unnecessary components are ignored because we focus only on the scheduling issue in terms of job size and power. We assume at this time that the electricity price is staying in the on-peak period and the power budget is 150 W for the whole system. The two rectangles in the right part of Figure 1 represent two potential scheduling solutions. The upper one stands for the typical behavior of the backfilling scheduler where the jobs' power is not a

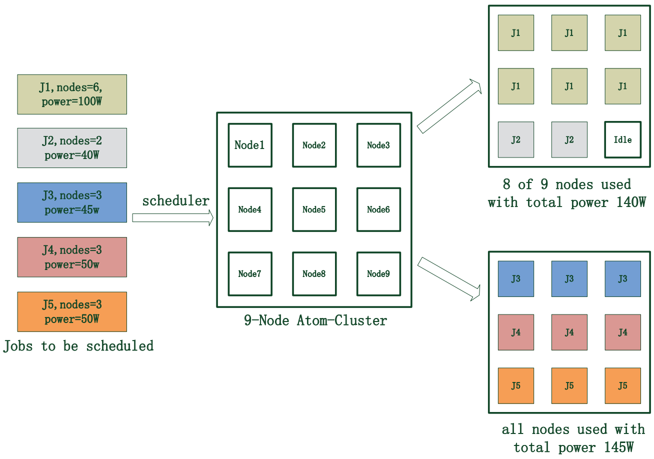


Fig. 1. Scheduling 5 jobs using traditional (top right) and power-aware scheduling (bottom right) separately.

concern. Once the scheduling decision for this time slot is made, there will be no changes unless some jobs cannot acquire needed resources. As shown in this figure, two jobs (J1 and J5) occupy eight nodes, leaving one node idle because J3, J4, and J5 all require more than one node. So at this time 8 out of 9 nodes are running jobs, with a total power of 140 kW. In contrast, the rectangle in the lower right corner shows another possible combination of jobs. Its aim is to choose jobs whose aggregated power consumption will not exceed the power budget and to try to utilize nodes as much possible. Instead of choosing jobs in a first com, first served (FCFS) manner, it searches the waiting queue for an optimal combination of jobs that can achieve the maximum system utilization and do not break the power budget constraint. As a consequence, we can see J3, J4, and J5 are picked up and put on the cluster. With their total size exactly equivalent to the cluster size, their total power is 145 W, which does not exceed the power budget.

3.2 Scheduling Window

Balancing fairness and system performance is a critical concern when developing schedulers. The simplest way to schedule jobs is to use a strict FCFS policy plus backfilling [17, 18]. It ensures that jobs are started in the order of their arrivals. FCFS plus EASY backfilling is widely used by many batch schedulers; indeed, it has been estimated that 90% to 95% of batch schedulers use this default configuration [19, 20]. Under FCFS/EASY, jobs are served in FCFS order, and subsequent jobs continuously jump over the first queued job as long as they do not violate the reservation of the first queued job.

In our design, we use a window-based scheduling mechanism to avoid breaking the fairness of job scheduling as much as possible. Rather than allocating jobs one by one from the front of the queue as adopted by existing schedulers, our method allocates a window of jobs at a time. The selection of jobs into the window is to guarantee certain fairness, while the allocation of the jobs in the window onto system resources is to meet our objective of maximizing system utilization without exceeding the predefined power budget. The job scheduler makes decisions on a group of jobs selected from the waiting queue. Jobs within the group are called to be in a scheduling window. To ensure fairness as much as possible, the job scheduler selects jobs in the scheduling window based the system's original scheduling policy. This can be seen as a variant of FCFS in that this window-based approach treats the group of jobs in the front of the wait queue with the same priority.

3.3 Job Scheduling

We now describe how to formalize the scheduling problem listed in Section 3.1 into a 0-1 knapsack model. We then present dynamic programming to efficiently solve the model.

0-1 Knapsack Model Suppose there are S available nodes in the system, J jobs as $\{j_i|1 \leq i \leq J\}$ to be scheduled, and a power budget denoted as PB. Hence we can formalize the problem into a classical 0-1 knapsack model as follows:

Problem 1. *To select a subset of* $\{j_i|1 \leq i \leq J\}$ *such that their aggregated power consumption is no more than the power budget, with the objective of maximizing the number of nodes allocated to these jobs.*

For each job j_i, we associate it with a gain value v_i and weight w_i. Here v_i represents the number of nodes allocated to the job, which will be elaborated in the next subsection, and w_i denotes its power consumption, which is usually measured in kilowatts per node or kilowatts per rack.

Problem 2. *To determine a binary vector* $X = \{x_i|1 \leq i \leq J\}$ *such that*

$$\begin{aligned} \mathit{maximize} \sum_{1 \leq i \leq J} x_i \cdot v_i,\ x_i = 0 \ \mathit{or}\ 1 \\ \mathit{subject\ to} \sum_{1 \leq i \leq j} x_i \cdot w_i \leq PB. \end{aligned} \tag{1}$$

Job Power Profiling To make an intelligent job allocation, we must precisely model the job power consumption. The IBM Blue Gene series is representative of contiguous systems, which means that only logically contiguous subsets of nodes can be grouped to serve a single job. For instance, in Blue Gene/P systems, the basic unit of job allocation is called midplane, which includes 512 nodes connected via a 3D torus network [21]. Two midplanes are grouped together to form a 1024-node rack. Hence a job can be allocated more nodes than it actually requests.

Calculating job power consumption of a job on a contiguous system is a bit complicated. Because of the existence of the basic allocation unit, some nodes in the group serving a job may stay idle. Therefore, we derive the job power consumption as follows.

$$w_i = P_{work\_nodes} + P_{idle\_nodes} \tag{2}$$

As shown in this equation, the total power consumption of job j_i is the sum of two parts: that of the working nodes P_{work_nodes} and that of the idle nodes P_{idle_nodes}.

To calculate P_{work_nodes} and P_{idle_nodes}, we take the Blue Gene/P system as a simple example. We get the following formulas.

$$\begin{cases} P_{work\_nodes} = \frac{N_i}{N_i^{alloc}} \cdot \frac{P_i}{1024} \\ \\ P_{idle\_nodes} = \frac{N_i^{alloc} - N_i}{N_i^{alloc}} \cdot \frac{P_{idle}}{1024} \end{cases} \tag{3}$$

In Equation 3, P_i is the power consumption of job j_i, which is measured in kW per rack; N_i is the number of nodes j_i requests; and N_i^{alloc} is the number of nodes j_i actually get allocated. Sometimes $N_i \neq N_i^{alloc}$ because there exists a basic allocation unit (512-node midplane). $\frac{P_i}{1024}$ and $\frac{P_{idle}}{1024}$ denote the power consumption in kW/node transformed from kW/rack with a rack consisting of 1,024 computing nodes. P_i and P_{idle} can be obtained by querying the historical data of a job recorded by a power monitor. Many HPC systems have been equipped with particular hardware and software to detect the running information (e.g., LLView for Blue Gene/P; see [8]).

Dynamic Programming After setting up the gain value and weight, the 0-1 knapsack model can be solved in pseudo-polynomial time by using a dynamic programming method [22]. 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 $\{j_i | 1 \leq i \leq k\}$ with no more than the power budget as w, where $1 \leq k \leq J$. $G[k, w]$ has the following recursive feature.

$$G[k, w] = \begin{cases} 0 & k = 0 \ or \ w = 0 \\ G[k-1, w] & w_i \geq w \\ max(G[k-1, w], v_i + G[k-1, w - w_i]) & w_i \leq w \end{cases} \tag{4}$$

The solution $G[J, PB]$ and its corresponding binary vector X determine the selection of jobs scheduled to run. The computation complexity of Equation 4 is $O(J \cdot PB)$.

3.4 On-Line Scheduling on Blue Gene/P

We apply the 0-1 knapsack model after the scheduling decision has been made. The detailed scheduling steps are as follows.

Step 1: Use a traditional scheduling method to select a set of jobs denoted as J.

Step 2: Decide whether it is an on-peak period. If so, go to Step 3. If not, set J as the optimal solution, and go to Step 5.

Step 3: Apply the 0-1 knapsack model to the job set J using weight and value functions, and get the optimal combination of jobs.

Step 4: Release allocated nodes of jobs that are not in the optimal set.

Step 5: Start jobs in the optimal set.

Table 1. Experiment Configuration

Workload	Intrepid (BG/P) at Argonne National Lab.
No. of Nodes	40,960 (40 racks)
No. of Jobs	March, 2009: 9709; Arpil, 2009: 10503 May, 2009: 7925; June, 2009: 8317 July, 2009: 8241; Aug, 2009: 7592

Price Oeriod	On-peak(9am-11pm)
	Off-peak(11pm-9am)

Pricing Ratio	On-peak:Off-peak = 1:3, 1:4, 1:5
Job Power Profile	20 to 33 kW per rack 30 to 90 kW per rack 30 to 120 kw per rack
Power Budget	50%, 60%, 70%, 80%, 90%

4 Evaluation

We evaluate our power-aware scheduling algorithm by using trace-based simulations. In particular, we use the event-driven simulator called Qsim [23], which supports simulation of the BG/P system and its partition-based job scheduling. We extend Qsim to include our power-aware scheduling method. In this section, we describe the experiment configuration and our evaluation metrics. We then present our experimental results by comparing our power-aware scheduling with the default power-agnostic scheduling. Table 1 shows the overall configuration of our experiment that will be described in the next subsection.

4.1 Experiment Configuration

Job trace. Our experimental study is based on real job traces from workloads collected from two different systems: one workload is from the 40-rack Blue Gene/P system called Intrepid at Argonne. Intrepid is a partitioned torus system, so nodes can be allocated in order to connect them into a job-specific torus network [24]. One major advantage of using real job traces is that experimental results from simulation are more convincing and can reflect the system performance to the greatest extent. For Intrepid, we use a six-month job trace from the machine (40,960 computing nodes) collected by Cobalt, a resource manager developed at Argonne [25]. It contains 52,287 jobs recorded from March 2009 to August 2009. We apply our power-aware scheduling algorithm on a monthly base to see the results for months. By doing so, we are able to examine our algorithm under diverse characteristics of jobs such as different numbers of jobs and various job arriving rate.

Dynamic electricity price. In our experiments, we assume the most common type of variable electricity price, namely, on-peak/off-peak pricing. Under this pricing system, electricity costs less during off-peak periods (from 11pm to 9pm) and more when used during on-peak periods (from 9am until 11pm) [4]. Here we are not concerned about the absolute value of electricity price. Instead we care only about the ratio of on-peak pricing to off-peak pricing because one of our goals is to explore how much energy cost can be saved under our smart power-aware scheduling as compared with a default job scheduler without considering power or energy. According to the study listed in [7], the most common ratio of on-peak and off-peak pricing varies from 2.0 to 5.0. Hence we use three different ratios, 1:3, 1:4, and 1:5, in our experiments. In the figures in the following sections, we use "PR" to denote "pricing ratio" for short.

Job power profile. Because we lack the power information directly related to the testing workloads, we use the field data listed in [8] to estimate the approximate power consumption of jobs for our workloads. For the Intrepid workload, job power ranges between 20 to 33 kW per rack. We assign each job a random power value within this range using normal distribution, which fits the observation in [8] that most jobs fall into the 22 to 24 kW per rack range. For definiteness and without loss of generality, we apply another two sets of ranges as 30 to 90 kW and 30 to 120 kW per rack. These numbers should not be taken too literally since they are used to represent a higher ratio between high-power and low-power jobs for the newest supercomputers. In the following sections, we use "power" to denote the meaning "power per rack" for short.

Power budget. We evaluate five power budgets in our experiments as follows. We first run the simulation using the default scheduling policy and monitor the runtime power of the system; we then calculate the average power and set it as the baseline value. Respectively, we set the power budget to 50%, 60%, 70%, 80%, and 90% of the baseline value.

4.2 Evaluation Metrics

Our power-aware scheduling has three targets: saving energy cost, impacting system utilization only slightly, and preserving a certain degree of fairness.

Energy cost saving. This metric represents the amount of the energy bill that we can reduce by using our power-aware scheduling, as compared with the default scheduling approach without considering power or energy. In detail, the energy cost is calculated by accumulation during runtime. Because the price changes at different time periods, a monitor is responsible for calculating the current energy cost as an extension to Qsim.

System utilization rate. This metric represents the ratio of the utilized node-hour compared with the total available node-hours. Usually it is calculated as an average value over a specified period of time.

Fairness. Currently, there is no standard way to measure job fairness. Previous work on fairness of scheduling includes using "fair start time" [26] and measuring resource quality [27, 28]. To study the fairness of our power-aware scheduling, we propose a new metric by investigating the temporal relationship between the start times of jobs. Because we adopt a scheduling window when apply the 0-1 knapsack algorithm, Aay job within this window can be selected for scheduling. Such scheduling may disrupt the execution order between jobs included in that window. Here we introduce a new metric called "inverse pair." This idea is borrowed from the concept of permutation inverse in discrete mathematics. In combinatorial and discrete mathematics, a pair of element of (p_i, p_j) is called an inversion in a permutation p if $i \geq j$ and $p_i \leq p_j$ [29]. Similarly, we build a sequence of jobs, S, based on their start time using a traditional job scheduling method. S_i denotes the start time of job J_i. Also we build another sequence of jobs, called P, using our power-aware job scheduling method. In the same way P_i denotes the start time of job J_i. In sequence S and P, for a pair of jobs J_i and J_j, if $S_i \leq S_j$ and $P_i \geq P_j$, we call jobs J_i and J_j an "inverse pair." We count the total number of inverse pairs to assess the overall fairness. This metric reflects the extent of disruption to job execution caused by using budget controlling.

4.3 Results

Our experimental results are presented in four different groups. First, we evaluate energy cost saving gained by using our power-aware scheduling policy. Second, we study the impact to system utilization after using our scheduling policy. Third, we study the extent to which scheduling fairness is impacted by our power-aware scheduling policy. We also conduct detailed analysis of how the average power and system utilization change within a day. In the three groups above, we use the power range 20 to 33 kW of real systems and a pricing ratio 1:3 to present detailed analysis. We then conduct a complementary sensitivity study on energy cost savings and system utilization using different combinations of power ranges and pricing ratios.

Energy cost savings. Figure 2 presents the energy cost saving of six months on BG/P. In this figure, each group of bars represents one month in our job log. In each group, one single bar represents a power budget value (i.e., "PB-50%" means power budget is set to 50% of the baseline value). Obviously, our power-aware scheduling approach can help reduce energy cost significantly for BG/P systems. We notice that a lower power budget can save more energy cost than can a higher power budget. Intuitively, a higher power budget limits the power usage during the on-peak period more than a lower power budget does. The bars within each group appear to have a decreasing trend with higher power budget, with only two exceptions: in June a power budget of 90% can save a little more than one of 80%, and similarly in May. The largest energy cost savings happen

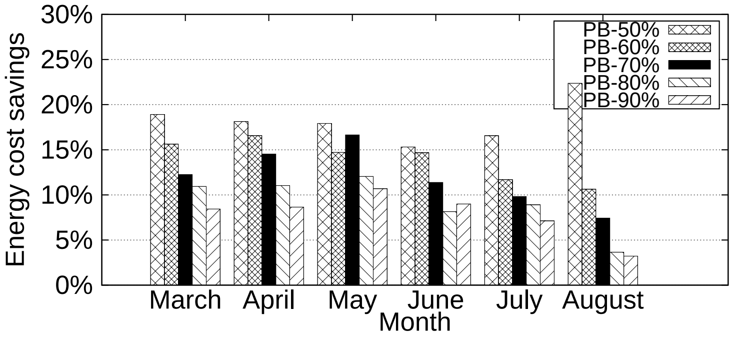


Fig. 2. Energy cost savings with power 20 to 33 kW per rack and pricing ratio 1:3.

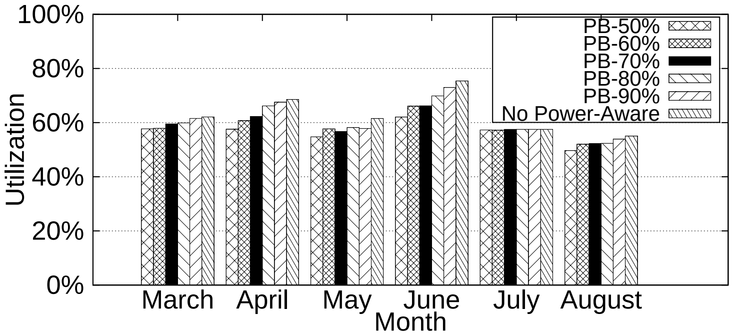


Fig. 3. Utilization with power 20 to 33 kW per rack and pricing ratio 1:3.

in August, as much as 23% with a power budget of 50%. For the other five months, using a power budget of 50% can achieve more than 15% cost savings. Even when using the highest power budget (PB-90%), most months can achieve more than 5% energy cost savings.

Impact on utilization. Figure 3 shows the system utilization of six months on BG/P. The bars are similar to those in Figure 2, and in each group there is an additional bar for the default scheduling method without power-aware scheduling. The figure shows that the system utilization is roughly 50% to 70%, which is comparable to many real scientific computing data centers and grids [30, 12]. Two features are prominent. First we observe that our power-aware scheduling approach only slightly affects the system utilization rate. Unlike energy cost savings, the disparity between high and low power budget is not large, compared with the base value. For example, the most utilization drop is in June when the original utilization rate using default scheduling policy is around 75%, while using a power budget of 50% results in a utilization rate of 62%. Second, whereas five power budgets are used in July, the utilization rates are almost the same, around 55%. This phenomenon is caused by our power-aware scheduling approach, which shifts some of the workload from the on-peak period to the off-peak period. We note that using a power budget would affect the utilization rate

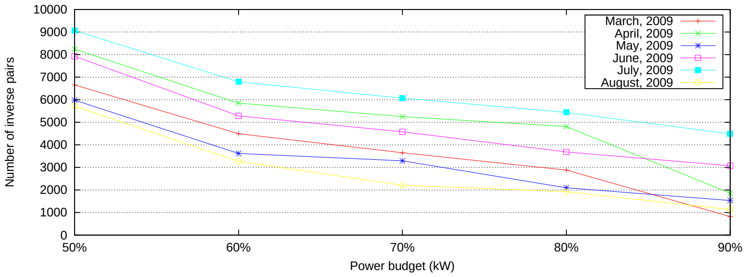
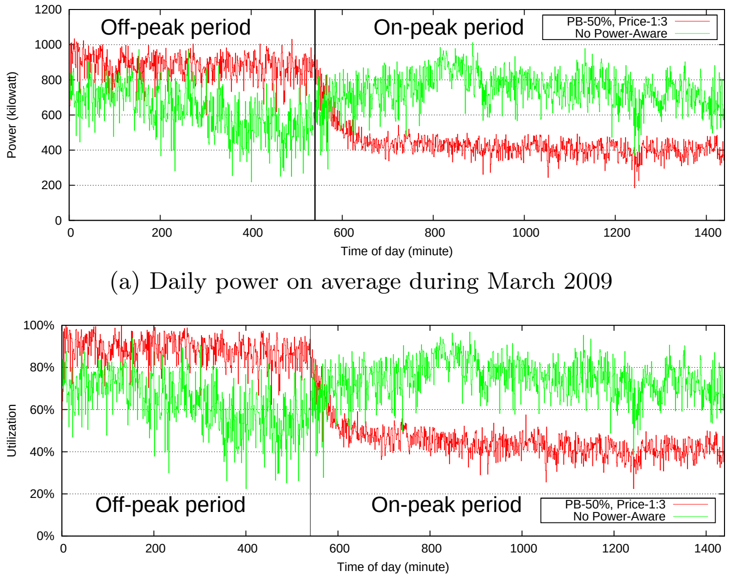


Fig. 4. Fairness of scheduling under power range 20 to 33 kW per rack and pricing ratio 1:3.

only during the on-peak period and would be compensated during the off-peak period.

Impact on fairness. We assess the fairness of scheduling using the metric "inverse pair." Her we evaluate how the power budget will affect the fairness of job scheduling. Figure 4 depicts the total number of inverse pairs on BG/P. The x-axis represents the power budget as introduced in the simulation configuration section. The y-axis denotes the total number of inverse pairs when we impose different power budgets. First we found that the number of inverse pairs occupies only a small portion of the total number of job pairs. For example, when we use a 50% power budget job log from March 2009 on BG/P, the number of inverse pairs is nearly 7,000, and the total number of job pairs in the job sequence is about 10^7 (9,709 jobs and $C^2_{9709} \approx 10^8$). From Figure 4 we can clearly see that the number of inverse pairs decreases as the power budget goes up; generally, the trend shows a linear decline as the power budget grows up. Even under the lowest power budget (50%) the total order of job executions is not affected too much, with the number of inverse pairs up to 8,000. Moreover, for all months, the number of inverse pairs dropped by nearly 50% when the power budget rose from 50% to 60%. We found that the number of inverse pairs is also related to the number of jobs in the job trace. Intuitively, a larger power budget is beneficial to scheduling more jobs and not disrupting their relatively temporal relationship.

Correlation between power and utilization. So far, we have studied the energy cost savings and system utilization rate under our power-aware scheduling approach. Here we present detailed results of the power and instant utilization within a day and how power-aware scheduling affects them. Figure 5(a) shows the average power in a day during March 2009 of a BG/P job trace. The x-axis represents the time in one day measured in minute. The y-axis represents the instant system power consumption recorded at each time point. The black line above minute=540 splits the time of a day into two periods of which the left part represents the off-peak period, whereas the right parts represents on-peak period. The green line represents a traditional system power pattern using the default scheduling policy. This reflects a common behavior of system power with relatively low power consumption early in the morning and higher during the rest of the day. Remember that in our simulation configuration, we set the on-peak



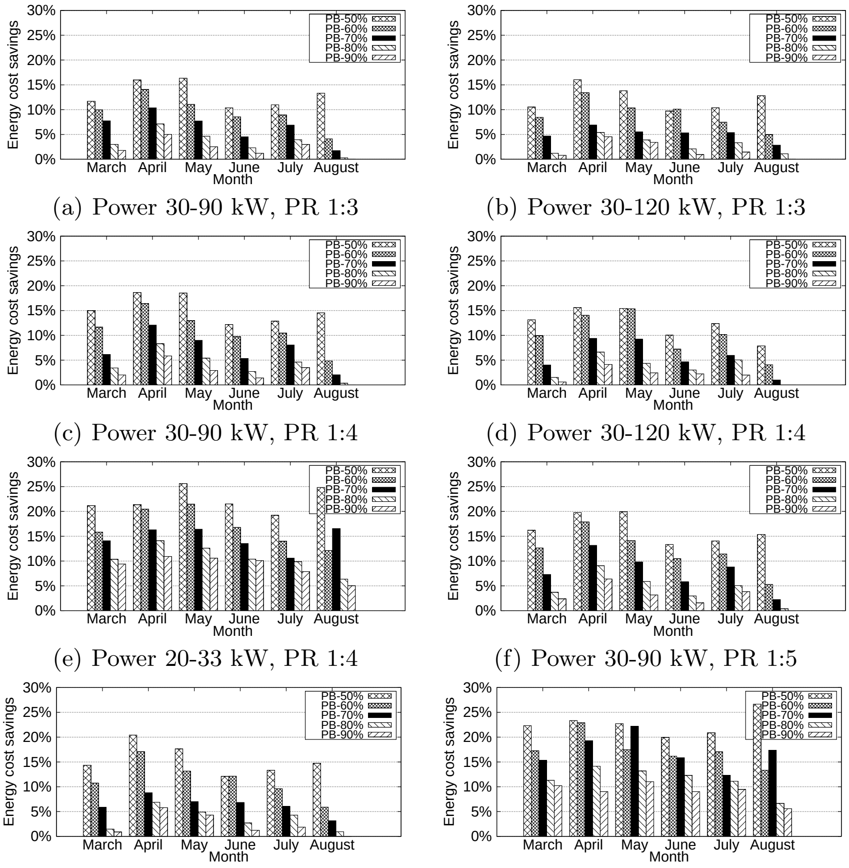
(a) Daily power on average during March 2009

(b) Daily utilization on average during March 2009

Fig. 5. Average power and utilization of BG/P using default no power-aware job scheduling (green line) power 20 to 33 kW per rack and pricing ratio 1:3 (red line).

period to be 9am to 11pm. Hence, we find that the system power with power-aware scheduling (red line) starts to go down around the time where minute=540. We can see that the red line is always below around 50% of the baseline value as half of the power of the green line. And we can also see that during the off-peak period where minute goes from 0 to 540, the system is running at a higher power rate indicated by the red line. The reason is that along with power-aware scheduling during on-peak period some jobs are delayed and get a chance to be run after on-peak period. These jobs can be seen as being "pushed" to off-peak period. This situation leads to more jobs in the waiting queue, causing more power consumption than the original scheduling method does. This fits what we expect, namely, that during the on-peak period less energy should be consumed and during the off-peak period more energy should be consumed.

Figure 5(b) shows the average utilization within a day for job log March 2009. The black dashed line where minutes=540 still acts as the splitting point between off-peak period and on-peak period. Obviously, we can see the curve of the utilization line conforms very closely to that in Figure 5(a). This fits the conclusion of multiple studies [6, 12] which have indicated that CPU utilization is a good estimator for power usage. First we examine the left part of the dashed line. Beginning from minute=540, the system utilization under default scheduling fluctuates around 80% and has a small drop off to 70% later on. This is typical in the real world because systems are always highly utilized in daytime where more



(a) Power 30-90 kW, PR 1:3

(b) Power 30-120 kW, PR 1:3

(c) Power 30-90 kW, PR 1:4

(d) Power 30-120 kW, PR 1:4

(e) Power 20-33 kW, PR 1:4

(f) Power 30-90 kW, PR 1:5

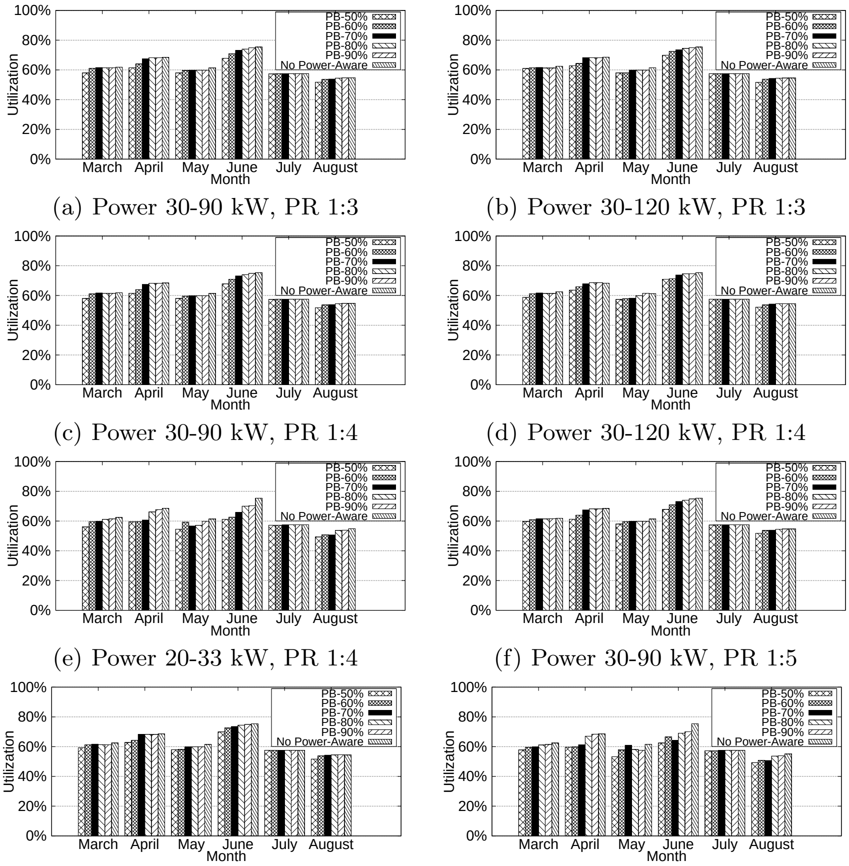
(g) Power 30-120 kW, PR 1:5

(h) Power 20-33 kW, PR 1:5

Fig. 6. Energy cost savings under various combinations; "Power" is the power range per rack, and "PR" is the pricing ratio.

jobs are submitted into the system. Also from minute=540, system utilization under power-aware scheduling policy starts to drop off and swiftly reaches a relatively stable value about 40% to 50%. Now we refer to the right part denoting the off-peak period. As shown in Figure 8, the green line and red line resemble the tendency in Figure 7. The utilization use power-aware scheduling is higher than scheduling without power budget most of the time. Overall, our power-aware scheduling approach reduces the system utilization during on-peak period and on the contrary raise it during off-peak period as compensation. Hence, it has only a slight impact on the whole system utilization, as we presented in the previous section.

Sensitivity study Our initial profiling experiments show that in future systems the difference of job power consumption is expected to be higher. Today's leadership-class supercomputers have wider power ranges, where the ratio be-



(a) Power 30-90 kW, PR 1:3

(b) Power 30-120 kW, PR 1:3

(c) Power 30-90 kW, PR 1:4

(d) Power 30-120 kW, PR 1:4

(e) Power 20-33 kW, PR 1:4

(f) Power 30-90 kW, PR 1:5

(g) Power 30-120 kW, PR 1:5

(h) Power 20-33 kW, PR 1:5

Fig. 7. Utilization under various combinations; "Power" is the power range per rack and "PR" is the pricing ratio.

tween peak and idle state can be as high as 1:4. Also with the smart grid project ramp-up and deployment in North America, more flexible electricity price policy will be provided in the future. So in this section, we conduct a sensitivity study to explore the potentiality of how different combinations of power and pricing ratio will affect our evaluation metrics. We use three power ranges, 20 to 30 kW, 30 to 90 kW, and 30 to 120 kW, along with three pricing ratios, 1:3, 1:4, and 1:5, in our experiments.

Figure 6 presents the energy cost savings under different combinations of power ranges and pricing ratios. In each small figure, "PR" is short for "pricing ratio". First we can see that our power-aware scheduling approach also works under these scenarios. For all months, a power budget of 50% can lead to no more than 10% energy cost savings. The largest energy cost savings can achieve more than 25%, as shown in Figure 6(e). We interpret these figures as follows.

First, we compare energy cost savings under variable power ranges and fixed pricing ratios. We observe that a narrower power range is likely to save more energy cost savings than is a wider power range. For example, when the pricing ratio is fixed to 1:4, the effect of a power range of 20 to 33 kW is greater than that of power range 30 to 90 kW and 30 to 120 kW in all months. Also under the same pricing ratio 1:4, a power range 30 to 90 kW results in greater energy cost savings than in almost all the months with a power range of 30 to 120 kW. The reason is that the majority of jobs in the BG/P log are small and thus require nodes less than 2 racks, which can have many more idle nodes. With many small jobs running in the systems, the system power tends to be lower than on systems where job sizes are more uniformly distributed. As a result, imposing a power budget cannot constrain the power of the whole system as much as that under a narrower power range.

Second, we focus on savings under a fixed power range and variable pricing ratio. Obviously, higher pricing ratios produce more energy cost savings in every month. Since some portion of the system power usage is shifted from the on-peak period to the off-peak period, higher pricing ratios lead to more difference in the cost of the amount of power transferred. Hence we believe one can saving save energy costs with a personalized electricity pricing policy in the future.

Figure 7 shows the system utilization rates under different combinations of power ranges and pricing ratios. Since the pricing ratio does not influence the utilization, we focus only on variable power ranges. First, similar to Figure 3, our power-aware job scheduling brings minor impact to system utilization rate. The largest utilization drop is in June in Figure 7(e), a drop of about 15%. Second, we observe that system utilization rates under wider power ranger are higher in some months. For example, in Figure 7(g) when the price is fixed at 1:5, using a power range of 30 to 120 kW gives rise to higher utilization in June than does a power range of 20 to 33 kW. This result reflects the role of our 0-1 knapsack algorithm. Under the same power budget, a wider power range can generate a more optimal solution because of the larger choosing space. But we notice that the effect is not that obvious because in BG/P the system resources released by jobs each time are relatively small compared with the whole system size, thus leaving little space for optimizing the system utilization rate.

4.4 Results Summary

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

- Our power-aware scheduling approach can effectively reduce the energy cost by up to 25%. For HPC centers such as the ALCF, this energy cost savings is translated into over $250,000 saving per year.
- This energy cost savings comes at the expense of a slight impact on system utilization during the on-peak price period. We also observe a modest increase in system utilization during the off-peak price period. In other words, the overall system utilization does not change much on a daily base.

- While our scheduling window preserves some degree of scheduling fairness, some jobs, especially those having high power consumption, will be delayed; but the delay is limited to a day.
- Based on our sensitivity study, we find that our power-aware job scheduling has a high potential to save energy costs and maintain system utilization.

5 Conclusion and Future Work

In this paper, we have presented a smart power-aware scheduling method to reduce electric bills for HPC systems under the condition of limiting the impact on system utilization and scheduling fairness. The design explores variable electricity prices and distinct job power profiles. Our approach contains three key techniques: a scheduling window, 0-1 knapsack, and on-line scheduling algorithm. Using real workloads from BG/P, we have demonstrated that our power-aware scheduling can effectively save energy costs with acceptable loss to system metrics such as utilization.

This is our first step in investigating power-aware job scheduling to address the energy challenge for HPC. We plan to extend our work in several ways. First, we will explore data analysis technology on historical data in order to examine power profiles for HPC jobs at various production systems; our aim is to enable us to predict job power profiles at their submission. Additionally, we are enhancing our power-aware scheduling with an adaptive self-tuning mechanism; the resulting job scheduler will be able to adjust its decision to external conditions such as electricity price automatically during operation. We also plan to integrate this work with our prior studies on fault-aware scheduling [23, 20, 31].

Acknowledgment

This work was supported in part by the U.S. National Science Foundation grants CNS-0834514 and CNS-0720549 and in part by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research under contract DE-AC02-06CH1135. We thank Dr. Ioan Raicu for generously providing high-performance servers for our experiments.

References

1. Z. Zhou, W. Tang, Z. Zheng, Z. Lan, and N. Desai, “Evaluating performance impacts of delayed failure repairing on large-scale systems,” in *2011 IEEE International Conference on Cluster Computing (CLUSTER)*, pp. 532–536, 2011.
2. K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Karp, S. Keckler, D. Klein, R. Lucas, M. Richards, A. Scarpelli, S. Scott, A. Snavely, T. Sterling, R. S. Williams, K. Yelick, K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Keckler, D. Klein, P. Kogge, R. S. Williams, and K. Yelick, “Exascale computing study: Technology challenges in achieving exascale systems,” 2008.

3. C. Patel, R. Sharma, C. Bash, and S. Graupner, “Energy aware grid: Global workload placement based on energy efficiency,” in *Proceedings of IMECE*, 2003.
4. I. Goiri, K. Le, M. Haque, R. Beauchea, T. Nguyen, J. Guitart, J. Torres, and R. Bianchini, “Greenslot: Scheduling energy consumption in green datacenters,” in *2011 International Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, pp. 1–11, 2011.
5. A. Jossen, J. Garche, and D. U. Sauer, “Operation conditions of batteries in pv applications,” *Solar Energy*, vol. 76, pp. 759–769, 2004.
6. X. Fan, W.-D. Weber, and L. A. Barroso, “Power provisioning for a warehouse-sized computer,” in *Proceedings of the 34th annual International Symposium on Computer Architecture*, ISCA ’07, (New York, NY, USA), pp. 13–23, ACM, 2007.
7. A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, “Cutting the electric bill for internet-scale systems,” in *Proceedings of the ACM SIGCOMM 2009 conference on data communication*, SIGCOMM ’09, (New York, NY, USA), pp. 123–134, ACM, 2009.
8. M. Hennecke, W. Frings, W. Homberg, A. Zitz, M. Knobloch, and H. Bttiger, “Measuring power consumption on IBM Blue Gene/P,” *Computer Science - Research and Development*, vol. 27, no. 4, pp. 329–336, 2012.
9. “Parallel workload archive.” http://www.cs.huji.ac.il/labs/parallel/workload/.
10. O. Mmmel, M. Majanen, R. Basmadjian, H. Meer, A. Giesler, and W. Homberg, “Energy-aware job scheduler for high-performance computing,” *Computer Science - Research and Development*, vol. 27, no. 4, pp. 265–275, 2012.
11. D. Meisner, C. Sadler, L. Barroso, W. Weber, and T. Wenisch, “Power management of online data-intensive services,” in *2011 38th Annual International Symposium on Computer Architecture (ISCA)*, pp. 319–330, 2011.
12. L. Barroso and U. Holzle, “The case for energy-proportional computing,” *Computer*, vol. 40, no. 12, pp. 33–37, 2007.
13. E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, “Load balancing and unbalancing for power and performance in cluster-based systems,” in *Proceedings of the Workshop on Compilers and Operating Systems for Low Power (COLP’01)*, 2001.
14. Y. Liu and H. Zhu, “A survey of the research on power management techniques for high-performance systems,” *Softw. Pract. Exper.*, vol. 40, pp. 943–964, Oct. 2010.
15. E. Lee, I. Kulkarni, D. Pompili, and M. Parashar, “Proactive thermal management in green datacenters,” *The Journal of Supercomputing*, vol. 60, no. 2, pp. 165–195, 2012.
16. J. Hikita, A. Hirano, and H. Nakashima, “Saving 200kw and $200k/year by power-aware job/machine scheduling,” in *IEEE International Symposium on Parallel and Distributed Processing, 2008, IPDPS 2008*, pp. 1–8, 2008.
17. Y. Etsion and D. Tsafrir, “A short survey of commercial cluster batch schedulers,” tech. rep., The Hebrew University of Jerusalem, Jerusalem, Israel, May 2005.
18. D. Feitelson and A. Weil, “Utilization and predictability in scheduling the ibm sp2 with backfilling,” in *Parallel Processing Symposium, 1998. IPPS/SPDP 1998. Proceedings of the First Merged International ... and Symposium on Parallel and Distributed Processing 1998*, pp. 542–546, 1998.
19. D. Tsafrir, Y. Etsion, and D. Feitelson, “Backfilling using system-generated predictions rather than user runtime estimates,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 6, pp. 789–803, 2007.
20. Y. Li, Z. Lan, P. Gujrati, and X.-H. Sun, “Fault-aware runtime strategies for high-performance computing,” *Parallel and Distributed Systems, IEEE Transactions on*, vol. 20, no. 4, pp. 460–473, 2009.

21. "Overview of the IBM Blue Gene/P project," *IBM Journal of Research and Development*, vol. 52, no. 1.2, pp. 199–220, 2008.
22. T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms.* McGraw-Hill Higher Education, 2nd ed., 2001.
23. W. Tang, Z. Lan, N. Desai, and D. Buettner, "Fault-aware, utility-based job scheduling on Blue Gene/P systems," in *IEEE International Conference on Cluster Computing and Workshops, 2009, CLUSTER '09*, pp. 1–10, 2009.
24. W. Tang, Z. Lan, N. Desai, D. Buettner, and Y. Yu, "Reducing fragmentation on torus-connected supercomputers," in *2011 IEEE International Parallel Distributed Processing Symposium (IPDPS)*, pp. 828–839, May 2011.
25. "Cobalt resource manager." http://trac.mcs.anl.gov/projects/cobalt.
26. G. Sabin, G. Kochhar, and P. Sadayappan, "Job fairness in non-preemptive job scheduling," in *International Conference on Parallel Processing, 2004, ICPP 2004*, pp. 186–194, vol. 1, 2004.
27. G. Sabin and P. Sadayappan, "Unfairness metrics for space-sharing parallel job schedulers," in *Job Scheduling Strategies for Parallel Processing* (D. Feitelson, E. Frachtenberg, L. Rudolph, and U. Schwiegelshohn, eds.), vol. 3834 of *Lecture Notes in Computer Science*, pp. 238–256, Springer, Berlin, 2005.
28. W. Tang, D. Ren, Z. Lan, and N. Desai, "Adaptive metric-aware job scheduling for production supercomputers," in *Parallel Processing Workshops (ICPPW), 2012 41st International Conference on*, pp. 107–115, 2012.
29. S. Pemmaraju and S. Skiena, *Computational Discrete Mathematics:Combinatorics and Graph Theory with Mathematica.* New York, NY, USA: Cambridge University Press, 2003.
30. I. Rodero, F. Guim, and J. Corbalan, "Evaluation of coordinated grid scheduling strategies," in *11th IEEE International Conference on High Performance Computing and Communications, 2009, HPCC '09*, pp. 1–10, 2009.
31. 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 *IEEE International Symposium on Parallel Distributed Processing (IPDPS), 2010*, pp. 1–11, 2010.

Government License