Reducing Electricity Cost Through Virtual Machine Placement in High Performance Computing Clouds

Kien Le
Rutgers University
lekien@cs.rutgers.edu

Jingru Zhang
Rutgers University
jingru@eden.rutgers.edu

Jiandong Meng
Rutgers University
jiandong@eden.rutgers.edu

Ricardo Bianchini
Rutgers University
ricardob@cs.rutgers.edu

Yogesh Jaluria
Rutgers University
jaluria@jove.rutgers.edu

Thu D. Nguyen
Rutgers University
tdnguyen@cs.rutgers.edu

ABSTRACT

In this paper, we first study the impact of load placement policies on cooling and maximum data center temperatures in cloud service providers that operate multiple geographically distributed data centers. Based on this study, we then propose dynamic load distribution policies that consider all electricity-related costs as well as transient cooling effects. Our evaluation studies the ability of different cooling strategies to handle load spikes, compares the behaviors of our dynamic cost-aware policies to cost-unaware and static policies, and explores the effects of many parameter settings. Among other interesting results, we demonstrate that (1) our policies can provide large cost savings, (2) load migration enables savings in many scenarios, and (3) all electricity-related costs must be considered at the same time for higher and consistent cost savings.

Categories and Subject Descriptors

D.4.1 [**Operating Systems**]: Process Management

General Terms

Design, Performance

Keywords

Multi-data-center, computing cloud, cooling, energy.

1. INTRODUCTION

Cloud computing is rapidly growing in importance as increasing numbers of enterprises and individuals are shifting their workloads to cloud service providers. Services offered by cloud providers such as Amazon, Microsoft, IBM, and Google are implemented on thousands of servers spread across multiple geographically distributed data centers. There are at least three reasons behind this geographical distribution: the need for high availability and disaster tolerance, the sheer size of the computational infrastructure, and the desire to provide uniform access times to the infrastructure from widely distributed user sites.

The electricity costs involved in operating a large cloud infrastructure of multiple data centers can be enormous. In fact, cloud service providers often must pay for the peak power they draw, as well as the energy they consume. For example, some data centers are designed to consume a maximum of 50MW of power, which is equivalent to a city with 40K households. Depending on utilization, the energy costs alone (not including the peak power costs) of a *single* such data center can easily exceed $15M per year. This cost can almost double when peak power is considered.

Lowering these high operating costs is one of the challenges facing cloud service providers. Fortunately, the geographical distribution of the data centers exposes many opportunities for cost savings. First, the data centers are often exposed to different electricity markets, meaning that they pay different energy and peak power prices. Second, the data centers may be placed in different time zones, meaning that one data center may be consuming off-peak (cheap) electricity while another is consuming on-peak (expensive) electricity. Finally, the data centers may be located in areas with widely different outside temperatures, which have an impact on the amount of cooling energy used.

Given the different characteristics of the data centers' energy consumptions, energy prices, and peak power prices, it becomes clear that we can lower operating costs by intelligently placing (distributing) the computational load across the wide area. This is exactly the topic of this paper. In particular, we consider services such as Amazon's EC2, which implements "infrastructure as a service" (IaaS). In IaaS services, users submit virtual machines (including the user's entire software stack) to be executed on the physical machines of the provider. Users are responsible for the management of their virtual machines, whereas the provider can potentially decide where the virtual machines should execute. IaaS services can host different user applications, ranging from interactive services to large high-performance computations (HPC). In this paper, we focus on IaaS services that support HPC workloads (e.g., [2, 12]).

In this context, our study proposes policies for virtual machine placement and migration across data centers. The policies are executed by front-end devices, which select the destination for each job (possibly a collection of virtual machines) when it is first submitted. Later, a (possibly different) front-end may decide to migrate the (entire) job to another data center. We assume that users provide an estimate of the running time of each job. Using this estimate, the front-end can estimate the cost of executing the job at each data center. Specifically, our policies model the energy costs, peak power costs, the impact of outside temperature on cooling energy consumption, and the overhead of migration. For comparison, we also consider policies that are cost-unaware, such as Round Robin, and a policy that is cost-aware but static.

An important aspect of our load placement approach is that a change in electricity price at a data center (e.g., when the data center transitions from on-peak to off-peak prices) could cause a significant increase in its load in just a short time. As cloud service providers operate their data centers closer to the overheating point (when the servers' inlet air reaches an unsafe temperature) to reduce cost, such load increases can compromise the ability of the data centers to adjust their cooling before servers start to turn themselves off. This problem is exacerbated if providers turn off the water chillers (when not needed) to further reduce cost. Chillers typically incur significant delays to become fully effective for cooling when they are turned on. We study this effect and its implications for data center cooling and our policies. In particular, our policies predict changes in the amount of load offered to the data centers, and start "pre-cooling" them if necessary to prevent overheating.

Our evaluation uses detailed simulations. Our study of cooling effects relies on Computation Fluid Dynamics simulations of temperatures in realistic data centers. Increasingly, cloud service providers are running their data centers at higher temperatures (i.e., closer to the overheating point) to reduce cooling energy consumption [28]. Our study shows that under such scenarios, large and rapid changes in load can indeed produce overheating. The study also shows that certain cooling strategies are substantially worse than others, and require longer periods of pre-cooling to prevent overheating. Finally, the study illustrates the effect of outside temperature on the energy consumption of the cooling infrastructure.

Our study of the load placement policies relies on event-driven simulations of a network of data centers using real workload traces. The results of this study show that our policies achieve substantially lower costs than their competitors. Migration provides non-trivial cost benefits, as long as the amount of data to migrate is not excessive (in which case the policies decide not to migrate). Moreover, our results show that it is critical for our policies to consider their three cost components: server energy, peak power, and cooling energy. Finally, our results explore the impact of different values for our key simulation parameters, including different electricity prices and outside temperatures.

We conclude that cost-aware load placement policies can lower operational costs significantly for cloud service providers. However, these policies must properly account for the significant changes in load that they may cause and how those changes may affect the provider's cooling infrastructure. Our policies are a strong first step in these directions.

In summary, our main contributions are:

- A demonstration of the transient effect of large and rapid increases in load on the cooling of a data center;
- A study of the behavior of many cooling strategies and the effect of different levels of pre-cooling on them;
- Effective policies for load placement in HPC cloud services that employ multiple data centers. The policies consider all power-related costs, as well as transient cooling effects; and
- Extensive results on the behavioral, energy, and cost implications of those policies.

2. BACKGROUND AND RELATED WORK

State of the practice. As far as we know, current cloud providers do not aggressively manage their cooling systems (e.g., by turning chillers off) beyond running data centers at relatively high (e.g., 28°C) server inlet air temperatures [5]. Moreover, current providers do not employ sophisticated load placement policies. For example, in Amazon's EC2 the users themselves select the placement of their virtual machines.

Our purpose with this paper is to study the potential benefits (and pitfalls) of these techniques in terms of electricity costs, and encourage providers to apply them.

Peak power costs. For many data center operators, the electricity cost actually has two components: (1) the cost of energy consumed (energy price: \$ per KWh), and (2) the cost for the peak power drawn at any particular time (peak power price: \$ per KW). Even though it has frequently been overlooked, the second component can be significant because the peak power draw impacts generation capacity as well as the power distribution grid. Govindan *et al.* estimate that this component can grow to as high as 40% of the energy cost of a data center [23].

The peak power cost typically corresponds to the maximum power drawn within some *accounting period* (e.g., 1 month). Many utilities track peak power by tracking average power over 15-minute intervals. The interval with the highest average power determines the peak power cost for the accounting period.

For generality, we assume that there are two peak power charges, one for the on-peak hours (e.g., weekdays 8am to 8pm) and one for off-peak hours (e.g., weekdays 8pm to 8am and weekends). Thus, if the accounting period is a month, the data center would be charged for the 15 minutes with the highest average power draw across all on-peak hours in the month, and for the 15 minutes with the highest power draw across all off-peak hours.

Cost-aware wide-area load placement. There have been a few works on cost-aware load placement in grid and federated cloud scenarios [7, 18, 22]. In the context of a wide-area distributed file service, Le *et al.* [30] considered cost-aware request distribution. Unfortunately, these works did not consider energy prices, peak power costs, or any cooling issues.

In the context of interactive Internet services, [31, 32, 44] considered request distribution across data centers. Qureshi *et al.* [44] studied dynamic request distribution based on hourly energy prices. Le *et al.* [31] considered hourly energy prices, on-peak/off-peak energy prices, and green energy sources. Le *et al.* [32] also considered on-peak/off-peak energy pricing, but mainly focused on capping the brown energy consumption of the service. Liu *et al.* [34] focused on exploiting green energy. These works dealt with short-running requests, rather than longer running jobs that may comprise multiple virtual machines. This difference is relevant in that techniques such as load migration do not make sense for short-running requests. More importantly, these previous works did not consider the transient effect of the large and rapid increases in load they may cause for certain data centers. Moreover, they did not consider peak power costs, outside temperatures, or cooling-related costs.

Job migrations. There have also been some works on migration in grid environments [29, 35]. These efforts focused mainly on migrating a job to a site with available resources or to a site where execution can be shorter. They did not consider costs of any kind, energy consumption, or cooling.

In [6, 24, 52], techniques for migrating virtual machines over the wide area were outlined. These works focused mostly on the mechanics of virtual machine migration. We leverage their techniques for migrating jobs in our policies.

Data center thermal management. Prior work has considered two classes of thermal management policies for data centers: those that manage temperatures under normal operation [36, 37, 39, 48] and those that manage thermal emergencies [11, 17, 26, 45, 51].

The works that focused on normal operation target reducing cooling costs. However, they make the handling of significant increases in load even more difficult. The reason is that these works enable

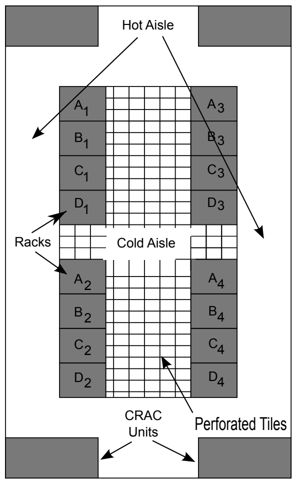


Figure 1: Data center with cooling. The water chiller is not shown.

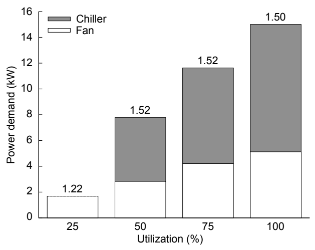


Figure 2: Impact of load on cooling power, assuming perfect LC; e.g., only 25% of the servers are on at 25% utilization. The numbers on top of the bars are the PUEs when counting only server and cooling energy. Outside temperature is 21°C and the maximum allowable temperature in the data center is 30°C.

increases of the inlet temperatures at the servers; i.e., there is less slack in the available cooling when the load increase comes. In contrast, the main goal of policies for managing thermal emergencies has been to control temperatures while avoiding unnecessary performance degradation. A large increase in load that causes temperatures to rise quickly can be considered a thermal emergency. However, none of these previous techniques can tackle such a significant increase in load and temperature without serious performance degradation.

Conserving energy in data centers. Many works have been done on conserving energy in data centers, e.g. [8–10, 16, 19, 25, 33, 43, 46]. Many of these works extend Load Concentration (LC) proposed in [43]. In LC, the system dynamically determines how many servers are required to serve the offered load, directs the load to this many servers, and turns the others off. In [21, 27], LC is applied in the context of virtual machines.

Our work differs from these efforts in that we consider load placement across data centers. In addition, our main goal is to lower costs, rather than conserve energy.

3. COOLING

The cooling system can be a significant source of energy consumption in a data center. For example, Barroso and Hölzle state that the cooling system can consume as much as 45% of the energy used by a typical raised-floor data center [5]. For a 10 MW facility, this would represent a cost of up to \$3.9M annually, assuming an electricity price of \$0.1 per kWh.

While many efforts have been expended to decrease this overhead as an aggregate, less attention has been paid to the dynamic behavior of cooling systems and how to leverage it to reduce cost and/or energy consumption. In this paper, we explore this dynamic behavior by simulating a cooling model for the data center shown in Figure 1. The data center contains 480 servers mounted in 16 racks in a 7m×8m×3m room.[1] Each server consumes a base power of 200W (power drawn when idle) and 300W when fully utilized.

Our modeled cooling system is typical of today's data centers, with computer room air conditioning (CRAC) units that take in hot air produced by servers, and blow cold air into the under-floor plenum. The cold air comes up through the perforated tiles on the floor and into the servers' air inlets to cool the servers. (Our modeled data center has four CRAC units.) If the outside temperature is sufficiently low, the CRACs discharge the hot air to and take cold air from outside. If the outside temperature is too high, the CRACs circulate the hot air through a water chiller for cooling before sending it back into the data center as cool air.

When the load is fixed, the total cooling energy consumed by the data center is the sum of the work performed by the CRACs and the chiller [47]. There are two main settings which affect the cooling system's energy consumption: (1) the CRAC fan speed, which determines the air flow rate through the data center; and (2) whether the chiller is turned on. Given an outside temperature and a data center utilization level, the cooling system can be designed to adjust the CRAC fan speed and chiller on/off setting to ensure reliable operation, as well as operate the data center in the most energy-efficient manner.

Our simulation uses a 3D model of the data center that includes all aspects of the air flow in the room and through the servers. Different utilization levels are modeled as racks being turned off in alternating sides. For example, we model a utilization of 50% with 4 fully on racks and 4 fully off racks on each side of the data center. We apply ANSYS Fluent 12.0 [3] turbulent k-epsilon with standard wall functions to obtain the results presented next.

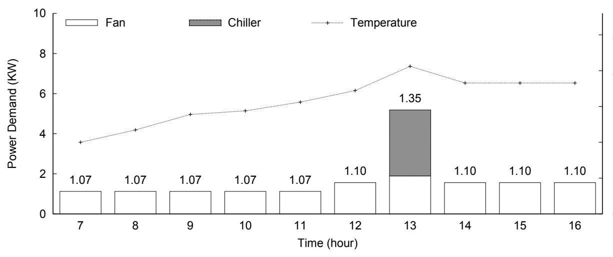
Figure 2 shows the power demand of the cooling system (using the most efficient setting), as a function of load when the outside temperature is 21°C. The number on top of each bar is the Power Usage Efficiency (PUE), i.e., the total energy consumed by the data center divided by the energy consumed by the servers and networking equipment. This figure shows that, for our scenario, cooling power demand can change by a factor of more than 7 as the load changes from 25% to 100%. The power demand of the chiller dominates when it has to be turned on. However, the power demand of both the fans and chiller increase with increasing load.

Figure 3 shows the outside temperature and cooling power demand as a function of time for two different locations during the same day. One can easily see that cooling energy consumption can change significantly throughout the day as the temperature changes.

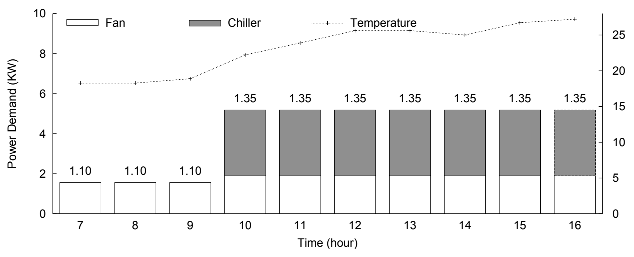
The significant variability of cooling energy consumption vs. outside temperature and load presents an interesting opportunity for dynamic load distribution to minimize energy consumption and/or cost. For example, if a service is replicated across two data centers, one in Northern California and one in Georgia, during the hottest and coolest periods of the day, it may make sense to direct load to Georgia because the cooling energy consumed is similar but the energy price may be lower in Georgia. On the other hand, when it is hot in Georgia but cool in Northern California, it may be beneficial to direct more load to Northern California because the cooling energy consumption is much smaller.

While it makes sense to explore dynamic load distribution (because of variable electricity prices as well as cooling), one has to carefully account for transient cooling effects. As service providers seek to reduce cost by operating data centers at higher temperatures, cooling systems have less time to react to load changes. This can lead to overheating if water chillers are dynamically turned off (when not needed) to further reduce cost. Specifically, while fan speeds can be changed almost instantaneously, there is a significant delay between when the chiller is turned on and when it becomes fully effective for cooling. Figure 4 shows what happens when the outside temperature is 23°C, the data center is operating at maximum inlet air temperature of 28°C, and the load rises rapidly from 25% to 75%. Such a large change in load may happen because a dynamic load distribution policy decides to migrate a large amount of

[1] We simulate a relatively small data center to keep simulation times reasonable. When scaling to larger data centers, efficiency of the cooling system can be improved in various ways, possibly leading to decreased energy consumption on the order of 30% [40].

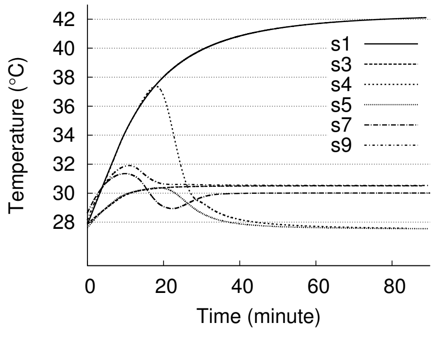


(a) Northern California

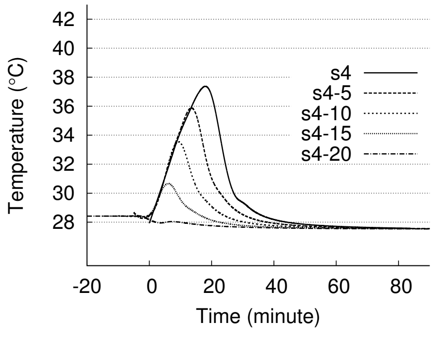


(b) Georgia

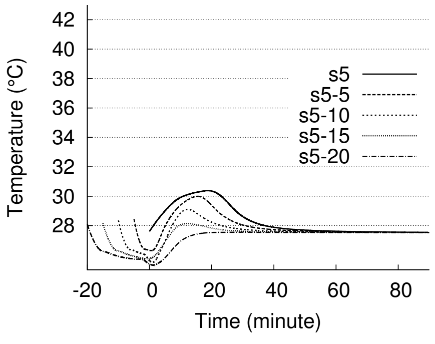
Figure 3: Impact of outside temperature on cooling power demand. The load at each data center is 50%. The maximum allowable temperature in the data center is 30°C.



(a)



(b)



(c)

Figure 4: Temperature in the data center vs. different cooling responses when utilization suddenly changes from 25% to 75%. Each line in (a) represents a cooling strategy listed in Table 1. (b) Strategy 4 with pre-cooling (taking each cooling action earlier) by 5 to 20 minutes. (c) Strategy 5 with pre-cooling by 5 to 20 minutes.

Strategy	Time (Minute)	Flow Rate (CFM)	Chiller	Through Chiller
1	$t > 0$	3350	OFF	NO
3	$t > 0$	10200	OFF	NO
4	$0 < t < 20$	3350	ON	NO
	$t > 20$	10200		YES
5	$0 < t < 20$	10200	ON	NO
	$t > 20$			YES
7	$0 < t < 5$	3350	OFF	NO
	$5 < t < 10$	5650		
	$10 < t < 15$	10200		
	$t > 15$	15200		
9	$0 < t < 5$	3350	OFF	NO
	$5 < t < 10$	5650	OFF	NO
	$10 < t < 15$	10200	OFF	NO
	$15 < t < 35$	10200	ON	NO
	$t > 35$	10200	ON	Yes

Table 1: Potential responses of a cooling system to sudden large load increase. Each scenario is a sequence of actions taken at different times. Actions include increasing flow rate (fan speed), turning on chiller, and circulating air through chiller.

load to the data center (because the electricity prices just dropped) or because a very large job (or batch of jobs) just arrived.

The figure plots the maximum inlet air temperature inside the data center against time, for many cooling responses (strategies) to the sudden load change occurring at time 0. The ultimate goal is to keep maximum inlet temperatures under 30°C [4]. We study many strategies because they produce different thermal behaviors as the load changes. (There has been little need to understand these response strategies until the recent trend of operating data centers at temperatures much closer to the overheating point; e.g., operating at 28°C when the maximum allowed temperature is 30°C.) Table 1 describes the cooling strategies in terms of flow rate and chiller activity. As their numbering suggests, these strategies are an interesting subset of those we studied, spanning the space of fan speed adjustments and chiller usage.

The figure shows that not adjusting the cooling settings (strategy 1) is not an option; the inlet temperatures increase significantly with the higher load. The other strategies lead to higher temperatures initially, but later bring them down. For example, strategy 3, which aggressively increases the flow rate but does not activate the chiller, stabilizes at above 30°C. Strategy 7, which ultimately uses a larger flow rate than strategy 3, but increases the flow rate more conservatively is even worse, allowing the temperature to rise to over 32°C before stabilizing just slightly above 30°C.

The only responses that eventually lead to a stable temperature under 30°C are those that use the chiller. Further, it is not sufficient to turn on the chiller when the load change is observed. In fact, the only strategies that do not lead to any overheating are the ones where the chiller was turned on early for pre-cooling (Figures 4(b) and (c)). How early pre-cooling must take place depends on the cooling strategy. Strategy 5, which immediately increases the flow rate when the chiller is turned on, only needs to pre-cool 5 minutes before the load change. On the other hand, strategy 4, which waits until the chiller is ready before increasing the flow rate, requires 20 minutes of pre-cooling.

As a result of the above transient effects, our cost-aware load distribution policy with migration has to predict future migrations, so that it can pre-cool the target data center if necessary (Section 4). To be safe, the policy looks ahead 20 minutes to ensure that the chiller will be fully effective before any load is migrated. We do not attempt to predict the arrival of very large jobs or large bursts of jobs because it is very difficult to accurately predict such events. Thus, arriving jobs may have to be delayed until they can be started without causing overheating.

4. LOAD DISTRIBUTION POLICIES

We now consider policies for distributing client load across multiple data centers to minimize electricity cost. We begin by discussing the assumptions and guidelines behind our policies. We then present our framework for computing the electricity cost and the estimation that our dynamic cost-aware policies will use. Finally, we present the dynamic cost-aware load distribution policies, which explicitly attempt to minimize cost, and several cost-unaware and static policies that will serve as comparison baselines.

4.1 Assumptions and Guidelines

Recall that jobs arrive to a front-end device, which may be located at one of the data centers used to serve the HPC workload or a separate location.[2] The front-end uses a load distribution policy to direct each job to one of the data centers for execution. Over time, if the policy includes job migration, the front-end may direct jobs to be migrated from one data center to another to reduce cost.

In addition to minimizing cost, the policies should also meet SLAs, and ensure that data centers are properly cooled and not loaded past their capacities. Our policies delay jobs to avoid overheating or overloading data centers. However, delaying jobs may lead to missed SLAs and possibly monetary penalties. Thus, the policies do not delay jobs past their SLAs except to preserve cooling and capacity constraints. Our policies lead to very few SLA violations (see Section 5.2), so we do not explicitly consider the SLA violation penalties here.

Each arriving job includes a specification of the amount of resources that it needs and an estimate of its run-time. It is quite easy to specify the amount of needed resources; this is typically under the direct control of the user (e.g., run this job on n VMs). Providing run-time estimates is also common practice in HPC systems [38, 49], where users often run the same applications many times and so can predict run-time based on experience. Currently, it is more challenging to accurately predict run-times on clouds, where the underlying hardware may be hidden from the user. However, we believe that cloud users will soon start requiring consistent resource allocation and/or stronger performance guarantees from cloud providers. With such features, users will be able to estimate run-times accurately and use those estimates in making informed performance and cost decisions. In systems like ours, providers can create incentives for users to provide accurate run-time estimates by: (1) lowering the priority or even suspending unfinished jobs that have exceeded their estimated run-times significantly, and (2) charging clients based on their resource use *and* the difference between their estimated and actual run times.

HPC jobs often need to access persistent data. User persistent data is typically replicated across (a few) geographically distributed data centers to guard against data unavailability or loss. This replication allows flexibility in but also constrains job placement. Specifically, if the data is stored at only a subset of the service provider's data centers, each user's jobs may only be placed at or moved to a data center that contains the user's persistent data. Updates to persistent data are streamed to replicas in the background when the corresponding jobs terminate. Furthermore, as in [52], the service can track all writes to the persistent store, enabling the transfer of only diffs when migrating VMs. Results are either small and easily transferable to clients, or are written to persistent storage and so are accessible from any replica.

Each job must be placed in a single data center to avoid requiring intra-job communication to cross data centers. (When migrating, the entire job has to be moved.) Services are typically over-provisioned such that there is always enough capacity to accommodate the offered workload (except for very rare events such as flash crowds). However, fragmentation of available resources across data centers may make it impossible to place an incoming job (that requires more resources than are currently available at any single data center). In this case, the job will be queued at the front-end until a data center becomes available.

The service's SLA with its customers involves each job completing within its specified run-time plus a slack that is a function of the resource requirement and run-time. Our policies attempt to place each arriving job at the data center that would lead to lowest cost. However, a low-cost data center may not be ready to accept a job because of the lag time to reach the needed level of cooling or because needed servers have been turned off (see below). In this case, all of our policies will place the job at a more expensive data center if waiting for the low-cost data center will lead to the job missing its SLA. If it is impossible to meet a job's SLA, then the job will be placed at the data center causing the smallest SLA violation.

To reduce energy consumption and cost, each data center only keeps as many servers active as necessary to service the current workload plus maintain a threshold percentage slack. The remaining servers are turned off to conserve energy [8–10, 25, 43] after they have been idle for a threshold amount of time. Server turn-on and turn-off both require time during which the server is consuming energy. A server cannot be used until turn-on has been completed. The slack allows arriving jobs requiring less than the threshold percentage of a data center's capacity to be started immediately, without waiting for servers to be turned on. Of course, when there is a burst of arrivals, even small jobs may have to wait for servers to be turned back on. Within each data center, we assume that data is stored on network-attached storage (and local Flash) so that turning servers off does not affect data availability; others have made the same assumption, e.g., [8].

The cooling system is always set to the lowest setting necessary to maintain a target maximum inlet temperature. As explained in Section 3, changing the fan speed does not incur significant delay. Thus, fan speed can be immediately adjusted as desired for job arrivals and exits. Turning on the chiller, however, does incur a significant delay. To avoid this delay, our policies only turn off the chiller after it has not been needed for a threshold amount of time.

Our policies currently do not dynamically consolidate jobs within a data center. Instead, they try to fully pack each server when jobs arrive so that additional consolidation makes very little difference.

Finally, we assume that data centers have homogeneous hardware; i.e., servers are identical. We make this simplifying assumption because we are most interested in load distribution across multiple data centers in this paper. To handle heterogeneous hardware (in future work), we envision a data center-level resource manager in addition to the front-end (which is a global resource manager). Each data center resource manager is responsible for managing VMs assigned to that data center, including assigning them to physical machines and consolidating them, thus hiding the details of the actual hardware from the front-end. When the front-end needs to decide where to place a batch of jobs, it would contact the manager of each usable data center to get the estimated cost of running the batch at that data center.

4.2 Cost Computing Framework

Our framework for computing the electricity cost of operating the service comprises the parameters listed in Table 2. Using these parameters, we formulate the following cost function:

[2] While we only consider one front-end in this paper for simplicity, it is possible to have multiple front-ends in general. The front-ends can either collaborate or statically divide the resources of the data centers among them for load distribution.

Symbol	Meaning
$Cost$	Total electricity cost (\$) of the service
DC	The set of data centers
$Cost_d^E$	Cost for energy consumed at data center d
$Cost_d^P$	Cost for peak power at data center d
$P_{d,t}$	Avg. power demand (KW) at data center d in epoch t
$P_{d,t}^B$	Base power demand at data center d in epoch t
$P_{d,t}^D$	Avg. dynamic power demand at data center d in epoch t
$P_{d,t}^C$	Avg. cooling power demand at data center d in epoch t
P_m^B	Base power demand of one idle server
P_m^D	Avg. dynamic power demand of one loaded server
$Price_{d,t}^E$	Energy price (\$/KWh) at data center d in epoch t
$PPeriods_d$	On-peak and off-peak time periods at data center d; each period is a set of time epochs
$PPower_{d,o}$	Peak power demand at data center d, o is *on-peak* or *off-peak*
$Price_{d,o}^P$	Peak power price (\$/KW) at data center d, o is *on-peak* or *off-peak*
$S_{d,t}^a$	Set of *active* servers at data center d in epoch t
$U_{s,t}$	Avg. utilization (%) of server s in epoch t
$Temp_{d,t}$	Avg. outside temperature at data center d in epoch t
$Cost_d^j$	Estimated cost of placing a job j at data center d
T_j	Run time of job j
M_j	The number of VMs comprising job j

Table 2: Parameters of cost computing framework.

$$Cost = \sum_{d \in DC} (Cost_d^E + Cost_d^P) \quad (1)$$

$$Cost_d^E = \sum_t P_{d,t}|t|Price_{d,t}^E \quad (2)$$

$$P_{d,t} = P_{d,t}^B + P_{d,t}^D + P_{d,t}^C \quad (3)$$

$$P_{d,t}^B = |S_{d,t}^a|P_m^B \quad (4)$$

$$P_{d,t}^D = \sum_{s \in S_{d,t}^a} U_{s,t}P_m^D \quad (5)$$

$$P_{d,t}^C = f(P_{d,t}^B + P_{d,t}^D, Temp_{d,t}) \quad (6)$$

$$Cost_d^P = \sum_{o \in PPeriods_d} PPower_{d,o}Price_{d,o}^P \quad (7)$$

$$PPower_{d,o} = \max_{t_p \in o} \frac{\sum_{t \in t_p} P_{d,t}|t|}{|t_p|} \quad (8)$$

The total electricity cost of the service (Equation 1) has two components, the cost for energy consumed by ($Cost_d^E$) and the cost for the peak power demand of ($Cost_d^P$) the service. $Cost_d^E$ is computed across time, where the time horizon is divided into discrete time epochs (Equation 2). Each epoch ends (and the next one starts) when any event affecting the energy consumption or cost of the system occurs. These events include changes in energy prices, changes in peak power prices, job arrivals and completions, server turn on and off, and change in cooling settings. The events are chosen to guarantee that prices and the number of jobs running at each data center (hence the number of active machines and their states) are constant within any single epoch.

For each time epoch, the cost of energy consumed is the product of the power demand ($P_{d,t}$), length of the time epoch ($|t|$), and energy price ($Price_{d,t}^E$). In turn, the power demand (Equation 3) is the sum of the base power demand of active machines ($P_{d,t}^B$), the dynamic power needed by the jobs currently running at the data center ($P_{d,t}^D$), and the power needed by the cooling system ($P_{d,t}^C$). Note that the power demand of an idle machine, i.e. one that is turned on but not running any jobs, is P_m^B, whereas the power demand of a fully loaded machine, i.e. one running a VM per each processor that it has, is $P_m^B + P_m^D$. We assume that the dynamic power demand of a machine is proportional to its load (e.g., $P_m^B + 0.25P_m^D$ when running 1 VM on a 4-processor machine). Thus, Equations 4 and 5 sum the power that is being drawn by all machines currently turned on in the data center.

As mentioned in Section 2, we assume two peak power charges, one for on-peak hours and one for off-peak hours, each with a potentially distinct peak power demand price (Equation 7). For each of the on-peak and off-peak periods, the cost is computed as the maximum power across all measurement intervals (e.g., 15 minutes each) in the period. Because each measurement interval may include many epochs, we compute the power of each interval as the average power over all the epochs within the interval (Equation 8).

4.3 Dynamic Cost-Aware Job Placement

Given the above framework for computing electricity cost, we are now faced with the problem of distributing jobs to minimize the overall cost (Equation 1) while still meeting their SLAs. In this section, we introduce two dynamic cost-aware, greedy distribution policies that attempt to place an arriving job j with minimal cost using the following estimates:

$$Cost_d^j = \delta(Cost_d^E) + \delta(Cost_d^P) \quad (9)$$

$$\delta(Cost_d^E) = \sum_{t \in T_j} \delta(P_{d,t})|t|Price_{d,t}^E$$

$$\delta(Cost_d^P) = \sum_{p \in PPeriods} \delta(PPower_{d,p})Price_{d,p}^P$$

where $\delta(Cost_d^E)$ is the change in cost due to additional energy consumption, i.e., additional base and dynamic server energy used to run j at data center d and any additional cooling required, and $\delta(Cost_d^P)$ is the change in cost due to increased peak power usage. Note that $\delta(Cost_d^P)$ may be 0 if running j at d does not increase the power demand above the peaks already reached in the past. On the other hand, if j crosses peak pricing periods, it may increase both peaks, which contributes to increasing the cost of the peak power demand. The energy consumed for cooling is computed using predicted outside temperatures.

The above estimates account for changes in energy price, peak power charge, and temperature; this is one reason why there may be multiple time epochs in T_j. We do not attempt to predict arriving load, however, so the estimates are only based on jobs currently in the system and j. The estimates can be computed either by accounting for jobs finishing and exiting the system (more accurate during low load periods) or by assuming that the workload at d will stay the same for all of T_j (more accurate during high load periods).

We do not show equations for $\delta(PPower_{d,p})$ and $\delta(P_{d,t})$ for brevity, since they are simple variations of Equations 3–6 and 8.

Cost-aware Distribution (CA). Consider a job j arriving at the start of an epoch t. Let D' be the subset of data centers that has sufficient unused capacity to accept j. If D' is empty, j is queued as explained above. Otherwise, CA orders the data centers in D' from least to most additional cost to run j using Equation 9. Then, CA places j at the first data center that can meet j's SLA. (Recall that a data center may not be able to meet a job's SLA because of delays necessary for ramping up cooling and/or turning on servers.) If no data center can meet j's SLA, CA places j at the data center that will violate j's SLA by the smallest amount.

Cost-aware Distribution with Migration (CAM). This policy places arriving jobs in the same way as CA. However, at the end of each epoch, it searches for job migrations that would lower the overall cost to complete the set of running jobs. We use the following simple search heuristic:

```
for $d_c$ from cheapest to most expensive data center:
  for $d_e$ from most expensive data center down to $d_c$:
    move jobs from $d_e$ to $d_c$ if have idle capacity at $d_c$
      and move would reduce cost
```

The cheapest to most expensive ordering of data centers can be computed using several metrics. Currently, we use an efficiency metric expressed as the energy cost (energy consumption of the servers and cooling system) divided by the number of active CPUs.

At the more expensive data center (d_e), jobs are ordered from highest to lowest using their remaining energy consumption. Consider jobs in this ordering and migrate a job j to the cheaper data center (d_c) if $C^j_{d_e} - C^j_{d_c} - C^j_M > 0$, where $C^j_{d_e}$ and $C^j_{d_c}$ are computed for the remaining run-time of j and C^j_M is the cost of migrating j from d_e to d_c. C^j_M comprises the cost of keeping the servers that j was running on at d_e and the servers that j will be running on at d_c active during the transfer time *T*. *T* is a function of the bandwidth between two data centers and the amount of data that needs to be transferred, which includes VM memory contents and any new persistent data generated since the job started running.

If single jobs cannot be moved profitably and there is still idle capacity at the cheaper data center, we consider migrating a batch of jobs because batch migration may improve the amortization of cooling cost (e.g., expense of turning on the chiller is amortized across many jobs instead of just one). The batching algorithm is simple. With the same job ordering as above, consider a batch of two up to a threshold batch size from the beginning of the ordering.

As described thus far, migration may increase the workload at a cheaper data center significantly. For example, consider the case when the electricity prices for a data center change from on-peak to off-peak. During on-peak hours, the data center may be mostly idle. At the start of off-peak pricing, however, it may be desirable to fully load the data center to leverage the lower prices. Recall from Section 3 that such a large shifting of load can result in overheating. Thus, CAM actually looks into the future for events that can lead to large-scale migration and pre-cools as necessary, so that migration can be done immediately when profitable. This prediction is relatively easy to implement because the look-ahead period is short (20 minutes delay for chiller turn-on) and only a few event types (currently only changes in energy and peak power prices) can result in large-scale migration.

4.4 Baseline Policies for Comparison

Round-Robin (RR). Arriving jobs are distributed in a round-robin fashion among the data centers for load balancing. This policy is completely oblivious to costs.

Worst Fit (WF). This policy selects the data center with the most available resources that will fit the arriving job. The policy is similar to RR but achieves better load balancing, because it explicitly considers the current load at each data center as well as the demand of the arriving job.

Static Cost-Aware Ordering (SCA). This policy accounts for differing costs in a gross, static manner. SCA considers the cost (including all three components: server energy consumption, peak power demand, and cooling) of supporting an average load (e.g., 50%) over a relatively long period of time (e.g., month or year) at each data center. It orders the data centers from cheapest to most expensive. It then directs each arriving job to the cheapest data center that has sufficient capacity to accept and run the job immediately. If no data center can run the job immediately, it chooses the data center with the best response time.

5. EVALUATION

5.1 Methodology

We use simulation to evaluate our framework and policies. Our multi-data center simulator takes as input an HPC workload trace, energy price traces, peak power charges, and temperature traces. Using these inputs, it simulates a load distribution policy and the resulting job completion times and data center energy consumptions. Our simulator accounts for the base energy of servers that are on (and turning-on/off), dynamic energy consumption, cooling levels needed to support specific data center loads vs. outside temperatures (cooling energy), delays in turning chillers on, and the resource usage (cores, memory, power demand, and migration footprints) of every job at a one-second granularity. Our evaluations are based on week-long traces, as well as sensitivity studies varying our main simulation parameters.

Workload. We use a trace of jobs submitted to an HPC cluster available from the Parallel Workloads Archive [14]. This trace contains approximately four months (December 1999 through April 2000) of data derived from the accounting records of the LSF software running on a 2048-node Origin 2000 cluster (Nirvana) at Los Alamos National Lab. The trace, LANL-O2K, contains a record for each job serviced by the cluster, with each record containing a job number, submitted time, actual run-time, and maximum number of cores and amount of memory used.

We extract a random week from this trace (January 24-31, 2000) and use it as the workload for all experiments discussed below. This week-long trace contains 6680 jobs, with a peak processing demand of 3867 cores. The maximum memory requirement of any one job is 4GB. Figure 5 shows the number of CPUs demanded by active jobs and Figure 6 shows the CDF of job run-times for our workload trace. There is a significant number of short running jobs (almost 30% run for less than 10 seconds) but there are also many long running jobs (>20% run for 1 hour or longer).

To map the above workload to our environment, we assume that each job can be split into VMs, one single-core VM per each core used by the job. The memory used by a job is equally divided among its VMs. When migrating a job, we assume that only the memory images of the VMs and a small amount of persistent data, expressed as the diffs of the data set [52], need to be transferred to the target data center.

As previously discussed, each arriving job specifies the number of VMs needed and an estimated run time. The service SLA is to complete each job within the estimated run-time plus 5% of the job's total estimated processing time (estimated run-time$\times$number of processors) plus 30 seconds. The latter part of the slack is to avoid missing the SLA for short running jobs when machines need to be turned on to accommodate them.

Data centers. We simulate a single front-end located on the East Coast of the US in most cases. The front-end distributes requests to three equal-size data centers located in the US West Coast (Northern California), US East Coast (Georgia), and Europe (Switzerland), respectively. As we focus on the cost savings enabled by geographical distribution, we assume that the data centers have the same PUE except for the cooling energy required by their outside temperatures. (Our framework can be easily extended to account for data centers with different PUEs and cooling technologies.) We

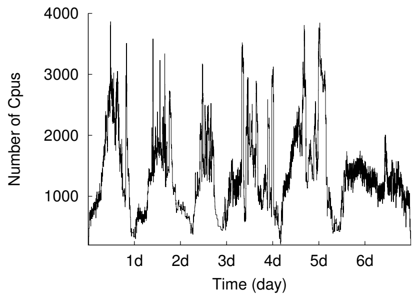


Figure 5: Number of CPUs demanded.

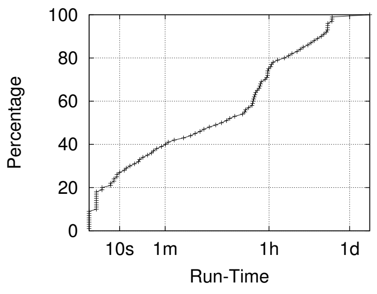


Figure 6: CDF of job run times.

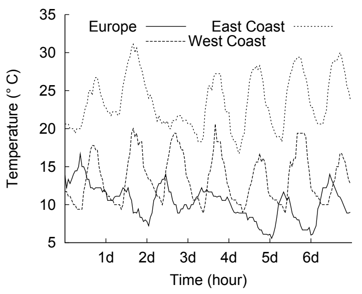


Figure 7: Outside temperatures.

Data Center	Energy	Peak Power
West Coast (SFO)	10.8 ¢/KWh	12 $/KW
East Coast (ATL)	10.7 ¢/KWh	12.83 $/KW
Europe (GVA)	11 ¢/KWh	13.3 $/KW

Table 3: Electricity prices [1,13,15,20,32,41,42].

assume that the users' persistent data are replicated across the three data centers, so that each job can be placed at any data center that has sufficient idle capacity to host the entire job. Similar to [52], we assume an inter-data-center bandwidth of 464 Mbps.

Provisioning each data center to be consistent with our modeled data center in Section 3 gives us a total of 5760 cores (1440 servers) across the three data centers. This corresponds to approximately 49% spare capacity at the peak load. This provisioning is consistent with current practices and allows the service to support the workload even if one of the data center fails.

Within a data center, each server has 4 cores and 4GB of memory. In our workload trace, no single VM requires more than 1GB of memory. Thus, job placement is essentially a core selection problem, i.e., a job can be placed at any data center that has enough idle cores. Within a data center, each VM of a job can be placed on any idle core. In our experiments, jobs are placed within a data center to minimize the number of active servers.

Each server's base power demand is 200W, with a peak power demand of 300W (corresponding to 100% utilization). Turning a machine on and off both require 30 seconds. Each data center turns off machines, if its has over 20% idle capacity. If not needed to maintain spare capacity, machines are turned off after they have been idle for 60 seconds.

Cooling. We use the cooling model developed in Section 3 to compute the energy consumed by the cooling system at each data center. Each data center adjusts its cooling system so that temperatures inside the data center never exceed 30°C. Jobs are never run before the cooling system can accommodate the increased heat. Specifically, if a chiller has to be turned on, arriving jobs are delayed until the chiller is ready (20 minutes). The front-end can ask data centers to pre-cool (by turning on the chiller early) in preparation to receive load; this is used in our cost-aware with migration policy. The chiller is turned off after it has not been used for 40 minutes.

Electricity prices. We simulate two sets of electricity prices at each data center, one for "on-peak" hours (weekdays from 8am to 8pm) and another for "off-peak" hours (weekdays from 8pm to 8am and weekends) [1, 13, 41]. The on-peak prices, listed in Table 3, are obtained either from previous publications [23, 32] or from information listed on power utility providers' Web sites [15,20,42]; by default, the off-peak prices are 1/3 of the on-peak prices.

Outside temperatures. We collected historical weather data from the Weather Underground Website [50] for the week of May 1-7, 2010. This period is interesting because not all locations are hot enough to require the chiller to be on all the time, and not all of them are cold enough to depend on just free cooling. Figure 7 shows the outside temperatures for our simulated period. Note that the East Coast location is quite a bit hotter than the other two locations but has slightly lower energy prices.

5.2 Comparing Policies

We begin our evaluation by comparing the cost of the service when using our dynamic cost-aware policies against the baseline policies. Figure 8 plots the total energy used by the service (left group of bars) and total cost (right group of bars). The energy usage is divided into three categories: server base energy, server dynamic energy, and cooling. The cost is divided into four categories: server base energy, server dynamic energy, cooling energy, and peak power. Both energy usage and cost are normalized against the results for CAM. SCA used a static data center ordering of Europe (most preferred), West Coast, and East Coast (least preferred). All SLAs were met by all policies.

We observe from these results that *dynamic* cost-aware load distribution can reduce cost significantly. Specifically, CAM outperforms the two load balancing policies, WF and RR, by 19% and 22%, respectively. CAM also outperforms SCA by 10%, showing that it is possible to leverage short-term differences in electricity prices and outside temperatures to reduce cost.

The ability to migrate jobs leads to a modest cost reduction; CAM outperforms CA by 3%. This is because CA makes placement decisions one-job at a time and never has a chance to correct its placement. In particular, it is difficult to predict the cost of cooling on a job-per-job basis. To see this, consider a set of jobs that arrive close to each other. When CA considers the placement of each job, placement to a particular data center d might seem expensive because the chiller would need to be turned on. CA would then place each job elsewhere, possibly at higher expense, even though placing the entire set of jobs at d would have allowed the cost of running the chiller to be amortized across the set, making d the lowest costing data center. CAM corrects this problem because it can consider the effect of migrating multiple jobs together.

The above differences can be observed in Figures 10 and 11, which show the reactions of CA and CAM, respectively, to changes in electricity prices and temperatures during a 4-hour period. CA uses the West Coast data center the most before hour 11 because of low off-peak electricity prices. However, it prefers to direct some load to the Europe data center rather than loading the West Coast data center to maximum capacity to avoid turning on the chiller. After hour 13, it gradually prefers Europe because of decreasing temperature (and increasing temperature in the West Coast). In contrast, CAM loads the West Coast data center to maximum capacity until hour 13; this difference in behavior is due exactly to CAM's ability to migrate batches of jobs as explained above. During the same time period, SCA uses the Europe data center almost exclusively, because it is the preferred data center.

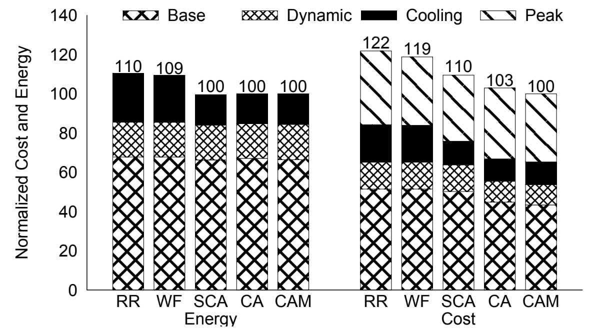


Figure 8: Energy used and total cost under different distribution policies.

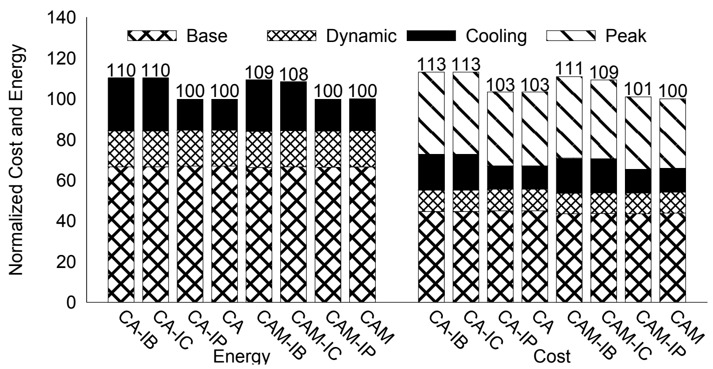


Figure 9: Impact of ignoring peak power and/or cooling cost. IP = ignore peak power, IC = ignore cooling cost, IB = ignore both.

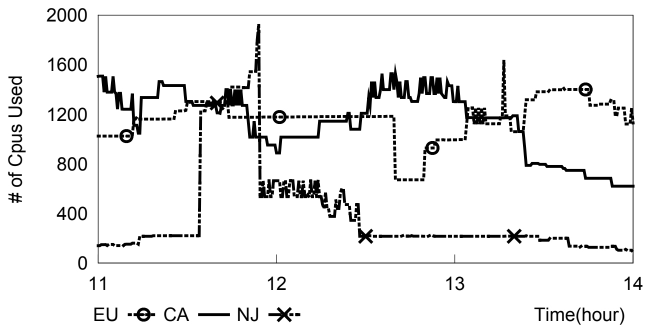


Figure 10: Load distribution under CA.

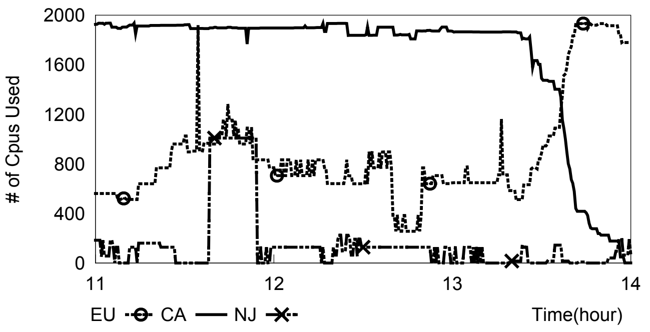


Figure 11: Load distribution under CAM.

Overall, CAM outperforms CA by trading-off slightly higher cooling cost for larger reductions in peak power and IT costs. Both CAM and CA lead to higher peak power cost than SCA because they may heavily load different data centers at different times. For example, Figure 11 shows CAM loading the West Coast data center to maximum capacity before hour 13, while loading the Europe data center to maximum capacity after hour 13. This causes both data centers to incur high peak power costs. In contrast, SCA always loads the same data center to maximum capacity before using the next. Thus, the second and third data centers may incur low peak power costs (because they are never highly utilized). CAM and CA outperform SCA because the savings in cooling and IT energy more than makes up for the higher peak power costs.

Our cost-aware policies (CAM, CA, SCA) lead to very few SLA violations: at most 5 violations out of 6679 job submissions, i.e., less than 0.075% of all SLAs. When a violation occurs, it lasts for only 9 minutes on average. All violations are caused by jobs with very short run-times (< 5s) that require large numbers of machines and arrive when chillers are off. These jobs' short run-times do not leave sufficient slack for turning on chillers.

Finally, our policies require a small amount of computation every epoch. However, this overhead is insignificant. For the workload we study, we observe an average epoch length of 25 seconds and an average overhead of 130ms per epoch on a single server, using unoptimized Java code for CAM, the dynamic policy with the highest computational overhead.

5.3 Peak Power and Cooling-Awareness

Next, we consider what happens if the dynamic cost-aware policies do not account for the cost of cooling and/or peak power demand. That is, we consider versions of the CA and CAM policies that use Equations 1 and 9 without the $Cost_d^P$ component and/or Equation 3 without the $P_{d,t}^C$ component. Figure 9 shows that such incomplete cost consideration can be expensive both in terms of cost and energy usage. Specifically, not accounting for cooling cost leads to 13% increased energy usage by CAM and 10% by CA because the policies place jobs on the East Coast even during the hottest periods to leverage slightly lower electricity prices. This leads to 8% increased cost for CAM and 13% for CA. On the other hand, not considering peak power demand has relatively little impact in our scenario because variations in peak power charges coincide with variations in energy prices: on-peak and off-peak periods are the same for energy prices and peak power charges. Moreover, the ratios of peak power charges and energy prices are close to each other across locations.

5.4 Sensitivity Analysis

In this section, we explore the sensitivity of CAM, CA, and SCA to various parameters.

Run-time predictions. In the above experiments, we assumed that the jobs' estimated run-times are exact. We also simulated scenarios in which estimated run-times differ from actual run-times according to a Poisson distribution with 10% average. This leads to inaccuracies ranging from 0 to 33% (0 to $\sim$7.9 hours). We found no significant changes to the results.

Migration time. The cost advantage gained by CAM depends on the amount of data that has to be transferred per migration. For the results presented above, each job migration required an average transfer of 50MB. We also considered scenarios where the transfer size is increased by 1-10GB per migration. As expected, the cost advantage of CAM decreases with increasing transfer size. However, CAM still outperformed SCA by approximately 8% when over 10GB must be transferred per migrated job. Further, the energy consumed under CAM only increased by 1%.

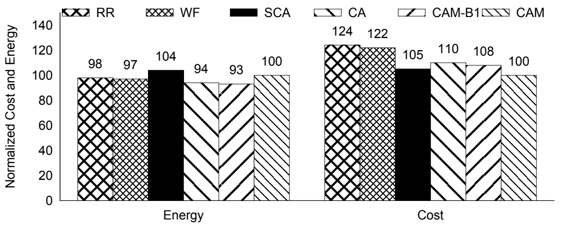


Figure 12: Energy used and total cost under different distribution policies when the East Coast data center is in North Carolina. CAM-B1 denotes CAM with batch size of 1.

Outside temperature. To gauge the impact of outside temperatures, we moved the data center in Northern California to Southern California, where the temperature is typically higher. This scenario favors CAM as it can start jobs at this data center when profitable to do so (e.g., off-peak electricity price) or when necessary because of capacity constraints (e.g., less expensive data centers are full), and then move the jobs to other data centers. This leads to an additional 1% cost advantage for CAM over both CA and SCA.

Energy prices. Figure 12 shows what happens when the energy price at the hottest data center is much cheaper: 5.23 cents/KWh on-peak. This setting represents an actual location on the East Coast (North Carolina) but with similar peak power charges and outside temperatures. In this case, the East Coast data center becomes the 2nd best for SCA, reducing the difference between SCA and CAM. In fact, the key difference between CAM and SCA is CAM's ability to leverage variable energy prices (e.g., off-peak on the West Coast is cheaper than on-peak on the East Coast).

In this scenario, SCA behaves slightly better than CA. This results from CA's mis-estimation of the per-job cost of cooling as discussed above. Figure 12 also shows the benefit of job batching for migration. CAM without batching (CAM-B1) costs 3% more than SCA and 8% more than CAM with batches of up to 10.

Relative data center sizes. Next, we consider what happens when the relative sizes of the data centers are changed. When we change the East Coast data center to twice as large as the other two, the cost advantage of CAM increases relative to all other policies. This is because the East Coast data center is, on average, the most expensive data center. CAM benefits from its ability to move jobs away from this data center when they were placed there because of capacity constraints. When we reduce the East Coast data center to 1/2 the size of the other two, the cost advantage of CAM decreases relative to the other policies, because it is easier for them to avoid this expensive data center with more capacity elsewhere.

Capacity over-provisioning. Finally, we investigate the effect of service capacity over-provisioning. Recall that the default capacity parameter in our experiments is $\sim$1.5 times the peak load. If we decrease the over-provisioning factor, the dynamic policies improve relative to SCA. Recall that CAM and CA reduce overall cost relative to SCA by trading off higher peak power costs for lower energy cost. With less over-provisioning, SCA is forced to utilize the less preferred data centers more, leading to increased peak power costs. CAM also improves relative to CA. The reason is that CA and CAM are forced to distribute jobs to more expensive data centers because of capacity constraints. However, CAM can move jobs back to the less expensive data centers when resources there are freed by exiting jobs, while CA cannot.

5.5 Migration and Pre-Cooling

Figure 13 shows three large migrations by CAM. At hour 8, electricity prices on the East Coast (North Carolina) went from off-peak to on-peak. This triggered a large migration from the East Coast to the West Coast. At time 11, electricity prices on the West Coast went from off-peak to on-peak. In response, CAM migrated a large number of jobs from the West Coast back to the East Coast. At time 14, electricity prices changed from on-peak to off-peak at the Europe data center. In response, CAM migrated a large number of jobs from the East Coast to Europe. In these instances, it was critical for the front-end to pre-cool the target data center by turning on the chiller 20 minutes before the migrations took place.

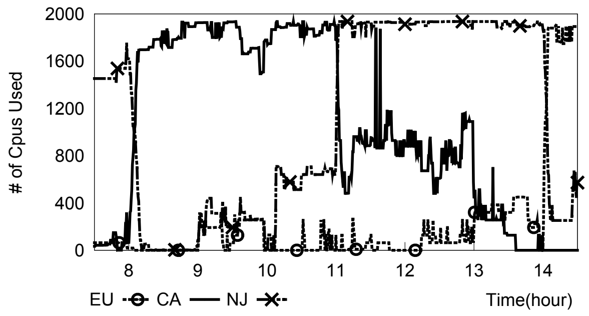


Figure 13: Large migrations by CAM at hours 8, 11, and 14.

5.6 Summary

Dynamic cost-aware load distribution can lead to significant cost savings. The actual savings depend on many parameters that determine how much dynamic cost diversity exists, and the flexibility with which policies can affect load distribution (e.g., if there is little spare capacity then there is almost no flexibility for load distribution). Intuitively, awareness of cooling is most important when the energy price and/or the peak power charge are only slightly lower at a hot location than at a cool location. In this case, not considering the cooling cost can lead to large cost and energy usage penalties when load distribution policies greedily choose the cheapest prices. Similarly, awareness of peak power charges is most important when the ratios of peak power charges to energy prices are widely different across time and/or data centers. However, to correctly account for all possible different scenarios, it is critical that the dynamic load distribution policy considers all three cost components.

6. CONCLUSIONS

In this paper, we have studied the possibility of lowering electricity costs for HPC cloud providers that operate multiple geographically distributed data centers. Specifically, we designed policies that intelligently place and migrate load across the data centers to take advantage of time-based differences in electricity prices and temperatures. Our policies account for three important electricity-related costs: energy price, peak power charge, and the energy consumed by the cooling system.

We developed a model of data center cooling for a realistic data center and cooling system. We simulated the model to obtain the cooling power as a function of data center load and outside temperature. This allowed us to investigate the impact of cooling on total cost, and explore whether cooling-aware load distribution can lead to cost savings. We also studied transient cooling effects resulting from abrupt, large changes in data center loads. We concluded that pre-cooling is necessary to prevent overheating in these scenarios. Our policies incorporate this pre-cooling.

Finally, we have shown that intelligent placement and migration of load can indeed lead to significant cost savings. Further, all electricity-related costs must be considered to maximize and ensure consistent cost savings.

7. REFERENCES

[1] Energy Information Administration. Average Retail Price of Electricity to Ultimate Customers by End-Use Sector, by State. `http://www.eia.doe.gov/cneaf/electricity/epm/table5_6_b.html`.

[2] Amazon. High Performance Computing Using Amazon EC2. `http://aws.amazon.com/ec2/hpc-applications`.

[3] ANSYS. ANSYS FLUENT. `http://www.ansys.com/Products/Simulation+Technology/Fluid+Dynamics/ANSYS+FLUENT`.

[4] ASHRAE. Environmental Guidelines for Datacom Equipment, 2008. American Society of Heating, Refrigeration, and Air-Conditioning Engineers.

[5] L.A. Barroso and U. Hölzle. The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines. *Synthesis Lectures on Computer Architecture*, 4(1), 2009.

[6] R. Bradford et al. Live Wide-Area Migration of Virtual Machines Including Local Persistent State. In *Proceedings of the 3rd International Conference on Virtual Execution Environments*, 2007.

[7] R. Buyya et al. Scheduling Parameter Sweep Applications on Global Grids: A Deadline and Budget Constrained Cost-Time Optimization Algorithm. *Software Practice and Experience*, 35(5), 2005.

[8] J. Chase et al. Managing Energy and Server Resources in Hosting Centers. In *Proceedings of the 8th Symposium on Operating Systems Principles*, 2001.

[9] G. Chen et al. Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services. In *Proceedings of the 5th Symposium on Networked Systems Design and Implementation*, 2008.

[10] Y. Chen et al. Managing Server Energy and Operational Costs in Hosting Centers. In *Proceedings of the International Conference on Measurement and Modeling of Computer Systems*, 2005.

[11] J. Choi et al. Modeling and Managing Thermal Profiles of Rack-mounted Servers with ThermoStat. In *Proceedings of the 13th International Symposium on High Performance Computer Architecture*, 2007.

[12] SARA Computing and Networking Services. High Performance Compute Cloud. `http://www.sara.nl/index_eng.html`.

[13] K. Coughlin et al. The Tariff Analysis Project: A Database and Analysis Platform for Electricity Tariffs, 2006. http://repositories.cdlib.org/lbnl/LBNL-55680.

[14] Dror Feitelson. Parallel Workloads Archive. `http://www.cs.huji.ac.il/labs/parallel/workload/`.

[15] Duke Energy. North Carolina Electric Rates. `http://www.duke-energy.com/north-carolina.asp`.

[16] E. N. Elnozahy, M. Kistler, and R. Rajamony. Energy Conservation Policies for Web Servers. In *Proceedings of the 4th Symposium on Internet Technologies and Systems*, 2003.

[17] A.P. Ferreira, D. Mosse, and J.C. Oh. Thermal Faults Modeling using a RC model with an Application to Web Farms. In *Proceedings of the 19th Euromicro Conference on Real-Time Systems*, 2007.

[18] S.K. Garg, R. Buyya, and HJ Siegel. Scheduling Parallel Applications on Utility Grids: Time and Cost Trade-off Management. In *Proceedings of the 32nd Australasian Conference on Computer Science*, 2009.

[19] R. Ge, X. Feng, and K.W. Cameron. Performance-Constrained Distributed DVS Scheduling for Scientific Applications on Power-aware Clusters. In *Proceedings of the 17th International Conference on High Performance Computing and Communications*, 2005.

[20] Georgia Power. Business Pricing - Georgia Power. `http://www.georgiapower.com/`.

[21] I. Goiri et al. Energy-aware Scheduling in Virtualized Datacenters. In *Proceedings of the International Conference on Cluster Computing*, 2010.

[22] I. Goiri, J. Guitart, and J. Torres. Characterizing Cloud Federation for Enhancing Providers' Profit. In *Proceedings of the 3rd International Conference on Cloud Computing*, 2010.

[23] S. Govindan, A. Sivasubramaniam, and B. Urgaonkar. Benefits and Limitations of Tapping into Stored Energy for Datacenters. In *Proceedings of the International Symposium on Computer Architecture*, 2011.

[24] Eric Harney et al. The Efficacy of Live Virtual Machine Migrations Over the Internet. In *Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing*, 2007.

[25] T. Heath et al. Energy Conservation in Heterogeneous Server Clusters. In *Proceedings of the 10th Symposium on Principles and Practice of Parallel Programming*, 2005.

[26] T. Heath et al. Mercury and Freon: Temperature Emulation and Management for Server Systems. In *Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2006.

[27] F. Hermenier et al. Entropy: A Consolidation Manager for Clusters. In *Proceedings of the 5th International Conference on Virtual Execution Environments*, 2009.

[28] Data Center Knowledge. Google: Raise Your Data Center Temperature. `http://www.datacenterknowledge.com/archives/2008/10/14/google-raise-your-data-center-temperature/`.

[29] K. Kurowski et al. Dynamic Grid Scheduling with Job Migration and Rescheduling in the GridLab Resource Management System. *Scientific Programming*, 12(4), 2004.

[30] K. Le et al. A Cost-Effective Distributed File Service with QoS Guarantees. In *Proceedings of the International Middleware Conference*, 2007.

[31] K. Le et al. Cost- And Energy-Aware Load Distribution Across Data Centers. In *Proceedings of the Workshop on Power Aware Computing and Systems*, 2009.

[32] K. Le et al. Capping the Brown Energy Consumption of Internet Services at Low Cost. In *Proceedings of the International Conference on Green Computing*, 2010.

[33] M.Y. Lim et al. Adaptive, Transparent Frequency and Voltage Scaling of Communication Phases in MPI Programs. In *Proceedings of the 18th International Conference on High Performance Computing and Communications*, 2006.

[34] Z. Liu et al. Greening Geographical Load Balancing. In *Proceedings of the International Conference on Measurement and Modeling of Computer Systems*, 2011.

[35] R.S. Montero, E. Huedo, and I.M. Llorente. Grid Resource Selection for Opportunistic Job Migration. *Proceedings of Euro-Par International Conference on Parallel and Distributed Computing*, 2004.

[36] J. Moore et al. Making Scheduling Cool: Temperature-Aware Workload Placement in Data Centers. In *Proceedings of the*

Annual USENIX Technical Conference, 2005.

[37] J. Moore et al. Weatherman: Automated, Online and Predictive Thermal Mapping and Management for Data Centers. In *Proceedings of the International Conference on Autonomic Computing*, 2006.

[38] A.W. Mu'alem and D.G. Feitelson. Utilization, Predictability, Workloads, and User Runtime Estimates in Scheduling the IBM SP2 with Backfilling. *IEEE Transactions on Parallel and Distributed Systems*, 12, 2001.

[39] T. Mukherjee et al. Software Architecture for Dynamic Thermal Management in Datacenters. In *Proceedings of the 2nd International Conference on Communication Systems Software and Middleware*, 2007.

[40] NREL. Best Practices Guide for Energy-Efficient Data Center Design, 2010. National Renewable Energy Laboratory (NREL), U.S. Department of Energy.

[41] Department of Energy and Climate Change. Quarterly Energy Prices. http://stats.berr.gov.uk/uksa/energy/sa20090625b.htm.

[42] Pacific Gas and Electric. ELECTRIC SCHEDULES. `http://www.pge.com/`.

[43] E. Pinheiro et al. Dynamic Cluster Reconfiguration for Power and Performance. *Compilers and Operating Systems for Low Power*, 2003. Earlier version published in COLP'01, September 2001.

[44] A. Qureshi et al. Cutting the Electric Bill for Internet-Scale Systems. In *Proceedings of SIGCOMM*, 2009.

[45] L. Ramos and R. Bianchini. C-Oracle: Predictive Thermal Management for Data Centers. In *Proceedings the 14th International Symposium on High Performance Computer Architecture*, 2008.

[46] B. Rountree et al. Adagio: Making DVS Practical for Complex HPC Applications. In *Proceedings of the 23rd International Conference on Supercomputing*, 2009.

[47] E. Samadiani et al. Adaptable Robust Design of Multi-Scale Convective Systems Applied to Energy Efficient Data Centers. *Numerical Heat Transfer*, 57(2), 2010.

[48] R. Sharma et al. Balance of Power: Dynamic Thermal Management for Internet Data Centers. In *Technical Report HPL-2003-5, HP Labs*, 2003.

[49] D. Tsafrir, Y. Etsion, and D. Feitelson. Modeling User Runtime Estimates. In *Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing*. Springer-Verlag, 2005.

[50] Weather Underground. Weather Underground. `http://www.wunderground.com/`.

[51] A. Weissel and F. Bellosa. Dynamic Thermal Management in Distributed Systems. In *Proceedings of the 1st Workshop on Temperature-Aware Computer Systems*, 2004.

[52] T. Wood et al. CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines. In *Proceedings of the 7th International Conference on Virtual Execution Environments*, 2011.