Online Resource Management for Data Center with Energy Capping

A. S. M. Hasan Mahmud
Florida International University

Shaolei Ren
Florida International University

Abstract

The past few years have been witnessing a surging demand for cloud computing services, resulting in a huge carbon footprint and making energy cost one of the top operational costs of data centers. Meanwhile, as sustainable computing has become increasingly important, data centers are constantly pressured to cap the long-term usage of their energy produced from carbon-intensive sources (a.k.a., "brown" energy). In this paper, we study *energy budgeting* and propose a novel online resource management algorithm, called ORM, to control the number of active servers for delay-sensitive workloads in a data center for minimizing the operational cost while satisfying the energy capping constraint. We rigorously prove that ORM achieves a close-to-minimum operational cost compared to the optimal offline algorithm with future information, while bounding the potential violation of energy capping, in an almost arbitrarily random environment. We also perform a trace-based simulation study to complement the analysis and validate the effectiveness of ORM.

1 Introduction

With the increasing popularity of cloud computing services, data centers are growing continuously in both number and scale, resulting in hundreds of thousands of server in one data center and requiring many megawatts of electricity. Recent studies have shown that the combined electricity consumption of the data centers is 623 billion kWh globally and would rank 5th among countries if the data centers were a country [8]. Consequently, for sustainable computing, large IT companies have been increasingly pressured to power their data centers using green energy while capping their brown energy usage, either mandated by governments in the form of Kyoto-style protocols or required by utility companies [7, 11, 20, 24].

While on-site renewable energy has been available in modern data centers, the intermittent nature makes it prohibitive to use renewables as the only energy source for large data centers. In practice, a significant portion of data center energy supply still comes from electricity which is often produced from carbon-intensive sources [7], making capping the brown (electricity) usage one of the top priorities for data center operators. In addition to highly-desired sustainability, capping the electricity usage also enables other remarkable benefits for data centers such as tax reduction, favorable accreditation, lower-cost contracts with utilities. Achieving energy capping clearly involves deciding the energy usage over a long term (e.g., 6 months or a year), and hence, we call this process *energy budgeting*, in comparison with power budgeting which allocates the peak power to different servers in a data center [12]. Nonetheless, energy budgeting fundamentally differs from power budgeting and faces a significant challenge: the data center operator needs to decide its energy usage in an online manner that cannot possibly foresee the far future time-varying workloads or intermittent on-site green/renewable energy supplies. Recently, some initial efforts have been made to achieve energy capping for data centers [11, 20, 24], but they require accurate prediction of long-term future information (e.g., workloads, renewable energy availability) that is typically unavailable in practice (e.g., due to business upgrades and/or unpredicted traffic spikes [5, 22]). Thus, to cope with the lack of accurate future information, online algorithms that can be applied based on the currently available information are required to achieve long-term energy capping.

In this paper, we study the long-term energy budgeting and propose a provably-efficient online resource management algorithm, called ORM, to control the number of active servers for minimizing the data center operational cost (incorporating both electricity and delay costs) while satisfying the energy capping constraint. As the foundation of ORM, a virtual energy budget deficit queue is constructed and used to guide data center decisions: a large queue length will prohibit the data center operator from continuing using a lot of electricity in order not to deviate too much from the energy capping constraint. Essentially, the budget deficit queue is updated as a feedback loop that takes the current electricity consumption as the input and aims at pushing the queue length to zero (i.e., satisfying the energy capping constraint). We conduct a rigorous performance analysis and

prove that ORM can achieve a parameterized operational cost that is arbitrarily close to the minimum cost achieved by the optimal offline algorithm with T-step lookahead information, while bounding the maximum energy budget deficit for almost any workload and renewable energy availability trajectories over the course of operation. We also perform a trace-based simulation study to complement the analysis.

The rest of this paper is organized as follows. Related work is reviewed in Section 2. Section 3 describes the model. In Section 4 and 5, we present the problem formulation and develop our online algorithm, ORM. Section 6 provides a simulation study and finally, concluding remarks are offered in Section 7.

2 Related Work

We provide a snapshot of the related work from the following aspects.

Data center optimization. There has been a growing interest in optimizing data center operation from various perspectives such as cutting electricity bills [9, 19, 21] , minimizing response times [4, 13] and reducing the brown energy usage [6, 15]. In particular, power proportionality via dynamically turning on/off servers has been extensively studied for energy saving [9, 13]. However, none of these studies have addressed the long-term energy capping or "energy budgeting".

Energy capping. The increasing pressure on sustainable cloud computing has recently catapulted *energy capping* as one of the most significant challenges for data center operators. The existing studies, e.g., [3, 11, 20], rely on long-term prediction of the future information, which may not be feasible in practice. Similarly, [24] utilizes the prediction of long-term future workloads to cap the monthly energy cost. These studies only use empirical evaluations without providing any performance guarantees. In comparison, ORM offers provable analytical guarantees on the average cost while bounding the deviation from long-term energy capping, and our simulation results also demonstrate the benefits of ORM over the existing method empirically.

3 Model

We consider a discrete-time model by dividing the entire budgeting period (e.g., 6 months or a year) into K time slots. Each time slot has a duration that matches the timescale of prediction window for which the data center operator can *accurately* predict the future information (including the workload arrival rate, renewable energy supply, and electricity price). For example, if the operator can only predict the hour-ahead future information, then each time slot corresponds to one hour and the operator can update its resource management decisions at the beginning of each hour. In the following analysis, we mainly focus on hour-ahead prediction for the convenience of presentation, while noting that the model applies as well to longer-term prediction. Throughout the paper, we also use *environment* to collectively refer to electricity price, on-site renewable energy supplies and workloads.

3.1 Workloads

We consider J types of workloads (or jobs, as interchangeably used in the paper). Different types of jobs differ in terms of the relative importance in the total cost function as well as their service rates (i.e. a server may process one type of jobs faster than another type). We denote by $\lambda_j(t) \in [0, \lambda_{\max,j}]$ the arrival rate of type-j jobs and by μ_j the service rate of a server for type-j jobs. We assume that $\lambda_j(t)$ is available at the beginning of each time slot t, as widely considered in prior work [2, 11, 15]. Nonetheless, even though $\lambda_j(t)$ is not accurately known, our results (in Section 6) show that ORM only incurs a very marginal additional cost. In our study, we focus on delay-sensitive interactive jobs (as in [15]), whereas delay-tolerant batch jobs can be easily maintained by a separate batch job queue as in [21, 23]. All jobs first arrive at a load distributor before they are distributed to servers for processing.

We denote the *delay cost* for type-j jobs by $d_j(\lambda_j, m_j)$, which is intuitively increasing in λ_j and decreasing in m_j where m_j is the number of (homogeneous) servers allocated to type-j jobs [15]. As a concrete example, we model the service process at each server as an M/M/1 queue and use the average response time (multiplied by the arrival rate) to represent the delay cost. Specifically, it is well known that the average response time for the M/M/1 queue at a server processing type-j jobs is $\frac{1}{\mu_j - \lambda_j/m_j}$ [19] and hence, the total delay cost at time t can be written as

$$\begin{aligned} d(\lambda(t), \mathbf{m}(t)) &= \sum_{j=1}^{J} w_j d_j(\lambda_j(t), m_j(t)) \\ &= \sum_{j=1}^{J} w_j \cdot \frac{\lambda_j(t)}{\mu_j(t) - \frac{\lambda_i(t)}{m_j(t)}}, \end{aligned} \tag{1}$$

where $\lambda(t) = (\lambda_1(t), \lambda_2(t), \cdots, \lambda_J(t))$, $\mathbf{m}(t) = (m_1(t), m_2(t), \cdots, m_J(t))$, and $w_j \geq 0$ is the weight indicating the relative importance of type-j jobs (i.e., a larger weight means the delay performance is more important). Note that we ignore the network delay cost, which can be approximately modeled as a certain constant [15] and added into (1) without affecting our approach of analysis. It should be made clear that the M/M/1 queueing model may not capture the exact response time in practice because: (1) workload service time often follows a heavy-tailed distribution such as Pareto distribution [10]; and (2) the inter-arrival time may not be exponentially distributed [18]. However, as queuing models with non-exponentially distributed inter-arrival and service times are difficult to analyze, the M/M/1 model has been widely used as an analytic vehicle to provide a reasonable approximation for the actual service

process [4, 15]. In addition, our analysis is not restricted to the specific delay cost given by (1). Finally, we note that the considered delay cost model is mainly intended to characterize the overall data center performance: if the overall delay cost is very high (e.g., too few servers are turned on), we will expect that the tenants/applications residing in the data center also experience a significant delay because of server overloading.

3.2 Data center

We consider a data center with on-site renewable energy plants (e.g., solar panels) as a complementary energy source and with M homogeneous servers, while noting that heterogeneous servers can also be easily incorporated.[1] Given a specific job type, processing speed of a server is quantified according to the service rate (rather than the actual clock rates), i.e., how many jobs can be processed *on average* in a unit time. Specifically, the service rate of a server for processing type-j jobs is μ_j. We denote by $m_j(t)$ the number of servers allocated to process type-j jobs at time t. In our study, we focus on server power consumption, while the power consumption of other parts such as power supply system and cooling system are captured by the power usage effectiveness (PUE) factor which, multiplied by the server power consumption, gives the total data center power consumption [14]. Mathematically, we denote the total power consumption[2] during time t by $p(\lambda(t),\mathbf{m}(t))$, which can be expressed as

$$p(\lambda(t),\mathbf{m}(t)) = \gamma\cdot\sum_{j=1}^{J} m_j(t)\cdot\left[e_0 + e_c\frac{\lambda_j(t)}{m_j(t)\mu_j}\right], \quad (2)$$

where $\gamma > 1$ is the PUE, e_0 is the static server power regardless of the workloads (as long as a server is turned on) and e_c is the computing power incurred only when a server is processing workloads. As in [14, 15], we ignore the possible *toggling* costs incurred when changing the resource management decisions (i.e., turning a server off or into deep sleep) that can be dealt with using techniques as developed in [13], since the decisions are updated infrequently (i.e., hourly in our study).

We denote the electricity price at time t by $u(t)$, which is known to the data center no later than the beginning of time t and may change over time if the data center participates in a real-time electricity market (e.g., hourly market [15]). Assuming that the available on-site renewable energy supply during time t by $r(t) \in [0, r_{\max}]$, we can express the incurred electricity cost during time t as

$$e(\lambda(t),\mathbf{m}(t)) = u(t)\cdot[p(\lambda(t),\mathbf{m}(t)) - r(t)]^+, \quad (3)$$

where $[\,\cdot\,]^+ = \max\{\cdot, 0\}$ indicating that no electricity will be drawn from the power grid if on-site renewable energy is already sufficient. While we use Eqn. (3) to represent the electricity cost (as considered by [15]), our analysis is not restricted to a linear electricity cost function and can also model other electricity cost functions such as nonlinear convex functions (e.g., data centers are charged at a higher price if it consumes more power).

[1] If multiple types of heterogeneous servers are considered, we need to decide how many servers of each type are allocated to the workloads.

[2] This is equivalent to energy consumption, since the length of each time slot is the same.

4 Problem Formulation

In this section, we specify the optimization objective as well as constraints, and present an offline formulation for our resource management problem. We also introduce an offline algorithm as a benchmark to compare ORM with.

4.1 Objective and constraints

Objective. We focus on operational costs rather than capital costs (e.g., building data centers, installing renewal energy generators), although capital costs are also significant and need to be minimized using separate techniques (e.g., optimizing the energy portfolio [20]). In data center operation, both electricity cost and delay cost are important, as the former takes up a dominant fraction of the operational cost while the later affects the user experiences and revenues [13]. Our study incorporates both costs by considering a parameterized cost function as follows

$$g(\lambda(t),\mathbf{m}(t)) = e(\lambda(t),\mathbf{m}(t)) + \beta\cdot d(\lambda(t),\mathbf{m}(t)), \quad (4)$$

where $\beta \geq 0$ is the weighting parameter adjusting the importance of delay cost relative to the electricity cost [15].

$$\bar{g} = \frac{1}{K}\sum_{t=0}^{K-1} g\left(\lambda(t),\mathbf{m}(t)\right), \quad (5)$$

where K is the total number of time slots over the entire budgeting period.

Constraints. To avoid server overloading and overprovisioning, resource management decisions need to satisfy

$$\lambda_j(t) \leq \theta\cdot\mu_j\cdot m_j(t),\ \forall j,t, \quad (6)$$

$$\sum_{j=1}^{J} m_j(t) \leq M,\ \forall t, \quad (7)$$

where $\theta \in (0,1)$ is the maximum utilization constraint for each server (i.e., $\frac{\lambda_j(t)}{\mu_j\cdot m_j(t)} \leq \theta$).

Next, assuming that electricity energy is "brown", we specify the brown energy capping constraint as follows

$$\frac{1}{K}\sum_{t=0}^{K-1}\left[p\left(\lambda(t),\mathbf{m}(t)\right) - r(t)\right]^+ \leq \frac{Z}{K}, \quad (8)$$

where Z is the capping constraint over the entire budgeting period. While we mathematically express the energy capping in (8) as as a *hard* constraint, it is in fact a desired

energy capping goal/target (e.g., as specified by utility companies and/or governments). If the energy cap is exhausted, mechanisms such as "cap-and-trade" (i.e., purchasing renewable energy credits to offset the exceeded electricity usage) may be adopted as proposed in [11], but they are orthogonal to our study. Note that our study can also address the scenario in which part of the electricity is produced by green energy sources: by multiplying the electricity usage with a certain factor that indicates the percentage of "brown" electricity, (8) specifies the constraint on the actual "brown" electricity usage.

4.2 Offline problem formulation

This subsection presents an offline problem formulation for resource management as follows.

$$\mathbf{P1}: \quad \min_{\mathscr{A}} \bar{g} = \frac{1}{K}\sum_{t=0}^{K-1} g(\lambda(t), \mathbf{m}(t)) \tag{9}$$

$$s.t., \quad \text{constraints (6), (7), (8),} \tag{10}$$

where $\mathscr{A}$ represents a sequence of decisions, i.e., $\mathbf{m}(t)$, for $t = 0, 1, \cdots, K-1$, which we need to optimize. It is natural that optimally solving **P1** requires complete offline information (i.e., workload arrivals, renewable energy supplies, and electricity prices) over the entire budgeting period that is very difficult, if not impossible, to accurately predict in advance, especially in view of the frequent traffic surges due to breaking events that can significantly boost the workloads [5, 22] and unpredictability of weather conditions that heavily affect the renewable energy availability [14].

T-step lookahead algorithm. We introduce an offline algorithm with T-step lookahead information as a benchmark. Specifically, we divide the entire budgeting period into R frames, each having $T \geq 1$ time slots, such that $K = RT$. There exists an oracle that has the complete information over the entire frame (i.e., T time slots) at the beginning of each frame. Then, at the beginning of the r-th frame, for $r = 0, 1, \cdots, R-1$, the oracle chooses a sequence of decisions to solve the following problem:

$$\mathbf{P2}: \quad \min_{\mathbf{m}(t)} \frac{1}{T}\sum_{t=rT}^{(r+1)T-1} g(\lambda(t), \mathbf{m}(t)) \tag{11}$$

$$s.t., \quad \text{constraints (6), (7),} \tag{12}$$

$$\sum_{t=rT}^{(r+1)T-1} [p(\lambda(t), \mathbf{m}(t)) - r(t)]^+ \leq \frac{Z}{R}. \tag{13}$$

Note that (13) is a *stronger* version of the original energy capping constraint in (8), as it requires the satisfaction of energy capping for every T time slots. Nonetheless, if T is sufficiently large, (13) will be almost equivalent to (8) and the oracle can still *approximately* solve the original problem **P1** [17]. In essence, **P2** encapsulates a *family* of offline algorithms parameterized by the lookahead information window size T. Next, to ensure there exists at least one feasible solution to **P2**, we make the two assumptions that are very mild in practice.

Boundedness assumption: The workload arrival rate $\lambda(t)$, electricity price $u(t)$, as well as renewable energy supplies $r(t)$ are finite, for $t = 0, 1, \cdots, K-1$.

Feasibility assumption: For the r-th frame, where $r = 0, 1, \cdots, R-1$, there exists at least one sequence of resource management decisions that satisfy the constraints of **P2**.

The boundedness assumption, combined with (6), ensures that the cost function is finite, while the feasibility assumption guarantees that the oracle can make a sequence of feasible decisions to solve **P2**. We denote the minimum average cost for the r-th frame by G_r^*, for $r = 0, 1, \cdots, R$, considering all the decisions that satisfy the constraints (12)(13) and that have perfect information over the frame. Thus, the long-term minimum average cost achieved by the oracle's optimal T-step lookahead algorithm is given by $\frac{1}{R}\sum_{r=0}^{R-1} G_r^*$, which is the benchmark that we compare ORM with.

5 Online Resource Management for Energy Capping

In this section, we present an online algorithm, ORM, which is provably efficient in terms of cost minimization compared to the optimal offline algorithm with T-step look ahead information.

5.1 ORM

The long-term energy capping constraint couples the online decisions across different time slots: the current decisions will implicitly affect the future decisions (e.g., using more electricity at this time slot will result in less energy budget available for future uses), while they have to be made without foreseeing the future. As the foundation of ORM, we construct a (virtual) brown energy budget deficit queue that *replaces* the long-term constraint (8), thereby decoupling the decisions for different time slots and enabling an online algorithm. Specifically, assuming $q(0) = 0$, we construct an energy budget deficit queue whose dynamics evolves as follows

$$q(t+1) = \left\{q(t) + [p(\lambda(t), \mathbf{m}(t)) - r(t)]^+ - z\right\}^+, \tag{14}$$

where $q(t)$ is the queue length indicating how far the current electricity usage deviates from the energy capping constraint, and $z = \frac{Z}{K}$ is the average electricity (or brown) energy budget per time slot. Intuitively, a larger queue length implies that the data center has drawn more electricity than the total energy budget thus far, and it needs to reduce the electricity usage for long-term energy capping. Leveraging this intuition, we develop our online algorithm, ORM, as presented in Algorithm 1.

ORM is purely online and requires only the currently available information (i.e., $\lambda(t)$, $r(t)$, $u(t)$) as the inputs. We use $V_0, V_1, \cdots, V_{R-1}$ to denote a sequence of positive control

Algorithm 1 ORM

1: Input: $\lambda(t)$, $r(t)$ and $u(t)$ at the beginning of each time $t = 0, 1, \cdots, K-1$
2: **if** $t = rT$, $\forall r = 0, 1, \cdots, R-1$ **then**
3: $q(t) \leftarrow 0$ and $V \leftarrow V_r$
4: **end if**
5: Choose $\mathbf{m}(t)$ subject to (6)(7) to minimize

$$V \cdot g(\lambda(t), \mathbf{m}(t)) + q(t) \cdot [p(\lambda(t), \mathbf{m}(t)) - r(t)]^+ \quad (15)$$

6: Update $q(t)$ according to (14).

parameters (also referred to as cost-budget parameters) to dynamically adjust the tradeoff between cost minimization and electricity usage over the R frames, each having T time slots. Line 5 defines an optimization problem to decide $\mathbf{m}(t)$ based on online information. When $q(t)$ increases (i.e., the current electricity usage further exceeds the allowed energy budget), minimizing the electricity usage is more critical for the data center operator due to the energy capping constraint. Thus, $q(t)$ can be essentially viewed as a feedback parameter to push the system towards a zero-queue state (i.e., satisfying energy capping).

5.2 Performance analysis

This subsection presents the performance analysis of ORM in Theorem 1, whose proof is available in [16].

Theorem 1. *Suppose that boundedness and feasibility assumptions are satisfied. Then, for any $T \in \mathbb{Z}^+$ and $R \in \mathbb{Z}^+$ such that $K = RT$, the following statements hold.*

a. *The energy capping constraint is* approximately *satisfied with a bounded deviation:*

$$\begin{aligned} &\frac{1}{K}\sum_{t=0}^{K-1}[p(\lambda(t), m(t)) - r(t)]^+ \\ \leq &\frac{Z}{K} + \frac{\sum_{r=0}^{R-1}\sqrt{C(T) + V_r(G_r^* - g_{\min})}}{R\sqrt{T}}, \end{aligned} \quad (16)$$

where $C(T) = B + D(T-1)$ with B and D being finite constants defined in [16], G_r^* is the minimum average cost achieved over the r-th frame by the optimal offline algorithm with T-slot lookahead information, for $r = 0, 1, \cdots, R-1$, and $g_{\min}$ is the minimum hourly cost that can be achieved by any feasible decisions throughout the budgeting period.

b. *The average cost $\bar{g}^*$ achieved by ORM satisfies:*

$$\bar{g}^* \leq \frac{1}{R}\sum_{r=0}^{R-1} G_r^* + \frac{C(T)}{R} \cdot \sum_{r=0}^{R-1}\frac{1}{V_r}. \quad (17)$$

Theorem 1 shows that, given a fixed value of T and R, ORM is $O(1/V)$-optimal with respect to the average cost against the optimal T-step lookahead policy, i.e., ORM incurs no more than $O(1/V)$ additive cost than the minimum value, while the energy capping constraint is guaranteed to be *approximately* satisfied with a bounded "fudge factor" of $\frac{\sum_{r=0}^{R-1}\sqrt{C(T)+V_r(G_r^*-g_{\min})}}{R\sqrt{T}}$. With a larger V, the cost is closer to the minimum but the potential deviation of electricity usage from the capping constraint can be larger, and vice versa. While admittedly (16) and (17) may not be tight for all scenarios, the established bounds hold under very mild assumptions (boundness and feasibility assumptions) that almost all the practical scenarios can easily satisfy. Thus, our analysis is still useful and it provides a robust performance guarantee. We will also consider case studies using real-world traces to provide more accurate estimates of costs and electricity usage for ORM.

6 Simulation

This section presents trace-based simulation studies to validate our analysis.

6.1 Data sets

We consider a data center with a peak power of 50MW and an average PUE of 2.0.[3] The data center consists of 100,000 servers in total, each with a maximum power of 250W and idle power of 150W. The budgeting period in our study is 6 months and the default energy capping constraint is 1.39×10^5MWh. The weight converting the delay cost to monetary cost in (4) is set to $\beta = 0.003$.

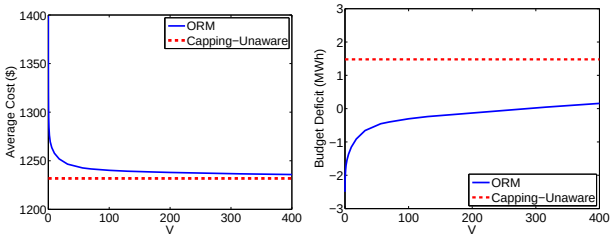
• Workloads: We use three different types of workloads with equal weights in the delay cost (i.e., $w_1 = w_2 = w_3 = 1$). We obtain the workload trace by profiling the server usage log of Florida International University (a large public university in the U.S.) from January 1 to June 30, 2012. We also adopt workload traces for Microsoft Research and Hotmail shown in [13] and repeat the traces for 6 months by adding random noises of up to $\pm 40\%$. The normalized service rates of each server for these three workloads are 0.95, 1.00 and 1.05, respectively. We scale these workloads in proportion to the total maximum service rate provided by the data center.

• On-site renewable energy: We obtain from [1] the hourly renewable energies (generated through solar panels and wind turbines) during the first six months of 2012, and scale them proportionally such that on-site renewable energy supply takes up approximately 10% of the total energy demand in the data center.

• Electricity: We obtain hourly electricity prices from one trading node in California available in [1] from January 1 to June 30, 2012.

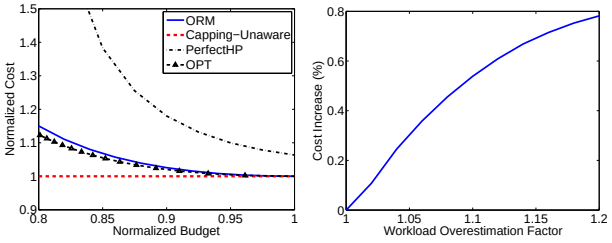
Since the access to commercial data centers is unavailable, we obtain the trace data from various sources, but it captures the variation of workloads, renewable energy supplies and electricity prices over the budgeting period. Thus,

[3] State-of-the-art techniques have reduced this value of around 1.12 [7].



(a) Cost versus V. (b) Budget deficit versus V.

Figure 1: Impact of V.



(a) Impact of budget. (b) Workload overestimation.

Figure 2: Performance comparison and sensitivity study.

it serves the purpose of evaluating the performance and benefits of ORM.

6.2 Results

We present three sets of simulation results using the above trace data to complement the analysis.

Impact of V. We first consider a constant V throughout the budgeting period. Fig. 1(a) and Fig. 1(b) show the impact of V on the average hourly cost (i.e., $\bar{g}$) and average hourly energy budget deficit (i.e., average hourly electricity usage minus the desired budget), respectively. Note that The result conforms with our analysis that with a greater V, ORM is less concerned with the budget deficit while caring more about the cost. In the extreme case in which V goes to infinity, ORM reduces to a *capping-unaware* algorithm that minimizes the cost without considering the energy capping constraint. Clearly, the capping-unaware algorithm achieves a cost that is a lower bound on the cost that can be possibly achieved by any algorithm It can be seen from Fig. 1(a) and Fig. 1(b) that the cost achieved by ORM is fairly close to the lower bound on the cost achieved by the capping-unaware algorithm when V is approximately 100, whereas still satisfying the energy capping constraint.

Performance comparison. We compare ORM with three other algorithms in Fig. 2(a) under various energy budgets. Under our simulation settings, the capping-unaware algorithm consumes 1.47×10^5MWh electricity energy over 6 months, which we normalize to 1. We appropriately choose V such that ORM achieves a zero budget deficit.

Comparison with capping-unaware: We see that even given a 85% energy budget, ORM only exceeds the capping-unaware algorithm by approximately 10% in terms of the average cost, while still being able to satisfy the budget constraint (whereas the capping-unaware algorithm clearly violates the budget constraint), as shown in Figs. 2(a). Furthermore, when the energy budget exceeds approximately 95%, ORM achieves almost the same cost as the capping-unaware algorithm while consuming less electricity energy over the budgeting period.

Comparison with OPT: The optimal offline algorithm (called OPT) has complete information over the entire budgeting period, and it achieves the minimum cost among all the possible algorithms satisfying the budget constraint. It can be seen from Fig. 2(a) that ORM is remarkably close to OPT in terms of the cost, demonstrating the flexibility of ORM in satisfying various budget constraints while resulting in a satisfactory cost.

Comparison with PerfectHP: We now consider a heuristic variant of the best known existing prediction-based solution studied in [11, 20, 24]: PerfectHP, which perfectly predicts the workloads over the next 48 hours and allocates the energy budget in proportion to the hourly workloads. Prediction of on-site renewable supply is not considered due to the practical difficulty of weather forecasting. Fig. 2(a) demonstrates that, while satisfying the energy capping constraint, ORM significantly outperforms PerfectHP by reducing the cost (by even more than 50% when the budget is low).

Sensitivity study. In practice, it may not be possible to perfectly predict hour-ahead workload arrivals. To cope with possible traffic spikes, we can either turn on more servers as a backup or directly overestimate the workload arrival rate by a certain overestimation factor ϕ: the higher ϕ, the more overestimates. We choose the later approach, and Fig. 2(b) shows that the total cost only increases by less than 0.8% even when we overestimate the workloads by 20%. This is because although workload overestimation may turn on more servers and incur a higher electricity cost at some time slots, the delay cost will be decreased.

Other sensitivity studies are also performed, demonstrating that ORM provides a satisfactory performance given various workloads and even with server toggling costs. These results are omitted due to space limitations.

7 Conclusions

In this paper, we studied "energy budgeting" and proposed a provably-efficient online resource management algorithm, ORM, to dynamically control the number of active servers for minimizing the data center operational cost while satisfying energy capping constraint. It was rigorously proved that ORM achieves a close-to-minimum operational cost compared to the optimal offline algorithm with future information, while bounding the potential violation of energy capping constraint, in an almost arbitrarily random environment. We also performed a trace-based simulation study to complement the analysis.

References

[1] California ISO, `http://www.caiso.com/`.

[2] N. Deng, C. Stewart, D. Gmach, M. Arlitt, and J. Kelley. Adaptive green hosting. In *ICAC*, 2012.

[3] N. Deng, C. Stewart, D. Gmach, and M. F. Arlitt. Policy and mechanism for carbon-aware cloud applications. In *NOMS*, 2012.

[4] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy. Optimal power allocation in server farms. In *SIGMETRICS*, 2009.

[5] A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M. A. Kozuch. Autoscale: Dynamic, robust capacity management for multi-tier data centers. *ACM Trans. Comput. Syst.*, 30(4):14:1–14:26, Nov. 2012.

[6] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav. It's not easy being green. *SIGCOMM Comput. Commun. Rev.*, 42(4):211–222, Aug. 2012.

[7] Google. Google's green ppas: What, how, and why.

[8] Greenpeace. How dirty is your data? a look at the energy choices that power cloud computing, 2011.

[9] B. Guenter, N. Jain, and C. Williams. Managing cost, performance and reliability tradeoffs for energy-aware server provisioning. In *IEEE Infocom*, 2011.

[10] M. Harchol-Balter. Task assignment with unknown duration. *J. of ACM*, 49(2):260–288, 2002.

[11] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi. Capping the brown energy consumption of internet services at low cost. In *IGCC*, 2010.

[12] H. Lim, A. Kansal, and J. Liu. Power budgeting for virtualized data centers. In *USENIX ATC*, 2011.

[13] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska. Dynamic right-sizing for power-proportional data centers. In *IEEE Infocom*, 2011.

[14] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser. Renewable and cooling aware workload management for sustainable data centers. In *SIGMETRICS*, 2012.

[15] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew. Greening geographical load balancing. In *SIGMETRICS*, 2011.

[16] A. S. M. H. Mahmud and S. Ren. Online resource management for data center with energy capping. *Tech. Report*, Available at: `http://www.cs.fiu.edu/~sren/doc/tech/fcw_13_full.pdf`.

[17] M. J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.

[18] V. Paxson and S. Floyd. Wide-area traffic: The failure of poisson modeling. *IEEE/ACM Trans. Networking*, 3(2):226–244, Jun. 1995.

[19] N. U. Prabhu. *Foundations of Queueing Theory*. Kluwer Academic Publishers, 1997.

[20] C. Ren, D. Wang, B. Urgaonkar, and A. Sivasubramaniam. Carbon-aware energy capacity planning for datacenters. In *MASCOTS*, 2012.

[21] S. Ren, Y. He, and F. Xu. Provably-efficient job scheduling for energy and fairness in geographically distributed data centers. In *ICDCS*, 2012.

[22] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari. Server workload analysis for power minimization using consolidation. In *USENIX ATC*, 2009.

[23] Y. Yao, L. Huang, A. Sharma, L. Golubchik, and M. J. Neely. Data centers power reduction: A two time scale approach for delay tolerant workloads. In *Infocom*, 2012.

[24] Y. Zhang, Y. Wang, and X. Wang. Electricity bill capping for cloud-scale data centers that impact the power markets. In *ICPP*, 2012.