Original citation:
Wang, Kai, Lin, Minghong, Ciucu, Florin, Wierman, Adam and Lin, Chuang. (2015) Characterizing the impact of the workload on the value of dynamic resizing in data centers. Performance Evaluation, Volume 85-86 . pp. 1-18.

.

Characterizing the Impact of the Workload on the Value of Dynamic Resizing in Data Centers☆

Kai Wang[a,*], Minghong Lin[b], Florin Ciucu[c], Adam Wierman[d], Chuang Lin[d]

[a] *State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China*
[b] *Facebook, USA*
[c] *Computer Science Department, University of Warwick, UK*
[d] *Dept. of Computing and Mathematical Sciences, California Institute of Technology, USA*
[e] *Dept. of Computer Science and Technology, Tsinghua University, China*

Abstract

Energy consumption imposes a significant cost for data centers; yet much of that energy is used to maintain excess service capacity during periods of predictably low load. Resultantly, there has recently been interest in developing designs that allow the service capacity to be dynamically resized to match the current workload. However, there is still much debate about the value of such approaches in real settings. In this paper, we show that the value of dynamic resizing is highly dependent on statistics of the workload process. In particular, both slow time-scale non-stationarities of the workload (e.g., the peak-to-mean ratio) and the fast time-scale stochasticity (e.g., the burstiness of arrivals) play key roles. To illustrate the impact of these factors, we combine optimization-based modeling of the slow time-scale with stochastic modeling of the fast time-scale. Within this framework, we provide both analytic and numerical results characterizing when dynamic resizing does (and does not) provide benefits.

Keywords: Data Centers, Dynamic Resizing, Energy Efficient IT, Stochastic Network Calculus

1. Introduction

Energy costs represent a significant, and growing, fraction of a data center's budget. Hence there is a push to improve the energy efficiency of data centers, both in terms of the components (servers, disks, network, power infrastructure [3, 4, 5, 6, 7]) and the algorithms [8, 9, 10, 11, 12]. One specific aspect of data center design that is the focus of this paper is dynamically resizing the service capacity of the data center so that during periods of low load some servers are allowed to enter a power-saving mode (e.g., go to sleep or shut down).

The potential benefits of dynamic resizing have been a point of debate in the community [13, 9, 14, 15, 16]. On one hand, it is clear that, because data centers are far from perfectly energy proportional, significant energy is used to maintain excess capacity during periods of predictably low load when there is a diurnal workload with a high peak-to-mean ratio. On the other hand, there are also significant costs to dynamically adjusting the number of active servers. These costs come in terms of the engineering challenges in making this possible [17, 18, 19], as well as the latency, energy, and wear-and-tear costs of the actual "switching" operations involved [20, 9, 21].

The challenges for dynamic resizing highlighted above have been the subject of significant research. At this point, many of the engineering challenges associated with facilitating dynamic resizing have been resolved, e.g., [17, 18, 19].

☆This paper is the full version of an accepted poster at ACM Sigmetrics/Performance 2012 [1] and an accepted short paper at the IEEE Infocom 2013 conference [2]

*Corresponding author. Tel.: +86 138 1049 1901.

Email addresses: wangkai@iscas.ac.cn (Kai Wang), mhlin@caltech.edu (Minghong Lin), F.Ciucu@warwick.ac.uk (Florin Ciucu), adamw@caltech.edu (Adam Wierman), chlin@tsinghua.edu.cn (Chuang Lin)

Additionally, the algorithmic challenge of deciding, without knowledge of the future workload, whether to incur the significant "switching costs" associated with changing the available service capacity has been studied in depth and a number of promising algorithms have emerged [11, 22, 9, 23, 12].

However, despite this body of work, the question of characterizing the potential benefits of dynamic resizing has still not been properly addressed. Providing new insight into this topic is the goal of the current paper.

The perspective of this paper is that, apart from engineering challenges, the key determinant of whether dynamic resizing is valuable is the workload, and that proponents on different sides tend to have different assumptions in this regard. In particular, a key observation, which is the starting point for our work, is that there are two factors of the workload which provide dynamic resizing potential savings:

(i) Non-stationarities at a slow time-scale, e.g., diurnal workload variations.

(ii) Stochastic variability at a fast time-scale, e.g., the burstiness of request arrivals.

The goal of this work is to investigate the impact of and interaction between these two features with respect to dynamic resizing.

To this point, *we are not aware of any work characterizing the benefits of dynamic resizing that captures both of these features*. There is one body of literature which provides algorithms that take advantage of (i), e.g., [9, 10, 11, 22, 24, 25, 16]. This work tends to use an optimization-based approach to develop dynamic resizing algorithms. There is another body of literature which provides algorithms that take advantage of (ii), e.g., [23, 21]. This work tends to assume a stationary queueing model with Poisson arrivals to develop dynamic resizing algorithms.

The first contribution of the current paper is to provide an analytic framework that captures both effects (i) and (ii). We accomplish this by using an optimization framework at the slow time-scale (see Section 2), which is similar to that of [11], and combining this with stochastic network calculus and large deviations modeling for the fast time-scale (see Section 3), which allows us to study a wide variety of underlying arrival processes. We consider both light-tailed models with various degrees of burstiness and heavy-tailed models that exhibit self-similarity. The interface between the fast and slow time-scale models happens through a constraint in the optimization problem that captures the Service Level Agreement (SLA) for the data center, which is used by the slow time-scale model but calculated using the fast time-scale model (see Section 3).

Using this modeling framework, we are able to provide both analytic and numerical results that yield new insight into the potential benefits of dynamic resizing (see Section 4). Specifically, we use trace-driven numerical simulations to study (i) the role of burstiness for dynamic resizing, (ii) the role of the peak-to-mean ratio for dynamic resizing, (iii) the role of the SLA for dynamic resizing, and (iv) the interaction between (i), (ii), and (iii). The key realization is that each of these parameters are extremely important for determining the value of dynamic resizing. In particular, for any fixed choices of two of these parameters, the third can be chosen so that dynamic resizing does or does not provide significant cost savings for the data center. Thus, performing a detailed study of the interaction of these factors is important. To that end, Figures 12-14 provide concrete illustrations of which settings of peak-to-mean ratio, burstiness, and SLAs dynamic resizing is and is not valuable. Hence, debate about the *potential* value of dynamic resizing can be transformed into debate about characteristics of the workload and the SLA.

There are some interesting facts about these parameters individually that our case studies uncover. Two important examples are the following. First, while one might expect that increased burstiness provides increased opportunities for dynamic resizing, it turns out the burstiness at the fast time-scale actually reduces the potential cost savings achievable via dynamic resizing. The reason is that dynamic resizing necessarily happens at the slow time-scale, and so the increased burstiness at the fast time-scale actually results in the SLA constraint requiring *more* servers be used at the slow time-scale due to the possibility of a large burst occurring. Second, it turns out the impact of the SLA can be quite different depending on whether the arrival process is heavy- or light-tailed. In particular, as the SLA becomes more strict, the cost savings possible via dynamic resizing under heavy-tailed arrivals decreases quickly; however, the cost savings possible via dynamic resizing under light-tailed workloads is unchanged.

In addition to detailed case studies, we provide analytic results that support many of the insights provided by the numerics. In particular, Theorems 1 and 2 provide monotonicity and scaling results for dynamic resizing in the case of Poisson arrivals and heavy-tailed, self-similar arrivals.

The remainder of the paper is organized as follows. The model is introduced in Sections 2 and 3, where Section 2 introduces the optimization model of the slow time-scale and Section 3 introduces the model of the fast time-scale and analyzes the impact different arrival models have on the SLA constraint of the dynamic resizing algorithm used in the slow time-scale. Then, Section 4 provides case studies and analytic results characterizing the impact of the workload on the benefits of dynamic resizing. The related proofs are presented in Section 5. Finally, Section 6 provides concluding remarks.

2. Slow Time-scale Model

In this section and the one that follows, we introduce our model. We start with the "slow time-scale model". This model is meant to capture what is happening at the time-scale of the data center control decisions, i.e., at the time-scale which the data center is willing to adjust its service capacity. For many reasons, this is a much slower time-scale than the time-scale at which requests arrive to the data center. We provide a model for this "fast time-scale" in the next section.

The slow time-scale model parallels closely the model studied in [11]. The only significant change is to add a constraint capturing the SLA to the cost optimization solved by the data center. This is a key change, which allows an interface to the fast time-scale model.

2.1. The Workload

At this time-scale, our goal is to provide a model which can capture the impact of diurnal non-stationarities in the workload. To this end, we consider a discrete-time model such that there is a time interval of interest which is evenly divided into "frames" $k \in \{1, ..., K\}$. In practice, the length of a frame could be on the order of 5-10 minutes, whereas the time interval of interest could be as long as a month/year. The mean request arrival rate to the data center in frame k is denoted by λ_k, and non-stationarities are captured by allowing different rates during different frames. Although we could allow λ_k to have a vector value to represent more than one type of workload as long as the resulting cost function is convex in our model, we assume λ_k to have a scalar value in this paper to simplify the presentation. Because the request inter-arrival times are much shorter than the frame length, typically in the order of 1-10 seconds, capacity provisioning can be based on the average arrival rate during a frame.

2.2. The Data Center Cost Model

The model for data center costs focuses on the server costs of the data center, as minimizing server energy consumption also reduces cooling and power distribution costs. We model the cost of a server by the operating costs incurred by an active server, as well as the switching cost incurred to toggle a server into and out of a power-saving model (e.g., off/on or sleeping/waking). Both components can be assumed to include energy cost, delay cost, and wear-and-tear cost. The model framework we adopt is fairly standard and has been used in a number of previous papers, e.g., see [11, 26] for a further discussion of the work and [27] for a discussion of how the model relates to implementation challenges.

Note that this model ignores many issues surrounding reliability and availability, which are key components of data center service level agreements (SLAs). In practice, a solution that toggles servers must still maintain the reliability and availability guarantees; however this is beyond the scope of the current paper. See [18] for a discussion.

The Operating Cost

The operating costs are modeled by a convex function $f(\lambda_{i,k})$, which is the same for all the servers, where $\lambda_{i,k}$ denotes the average arrival rate to server i during frame k. The convexity assumption is quite general and captures many common server models. One example, which we consider in our numeric examples later, is to say that the operating costs are simply equal to the energy cost of the server, i.e., the energy cost of an active server handling arrival rate $\lambda_{i,k}$. This cost is often modeled using an affine function as follows

$$f(\lambda_{i,k}) = e_0 + e_1\lambda_{i,k} \,, \tag{1}$$

where e_0 and e_1 are constants [28, 4, 29]. Note that when servers use dynamic speed scaling, if the energy cost is modeled as polynomial in the chosen speed, the cost $f(\cdot)$ remains convex. In practice, we expect that $f(\cdot)$ will be empirically measured by observing the system over time.

The Switching Cost

The switching cost, denoted by β, models the cost of toggling a server back-and-forth between active and power-saving models. The switching cost includes the costs of the energy used toggling a server, the delay in migrating connections/data when toggling a server, and the increased wear-and-tear on the servers toggling.

2.3. The Data Center Optimization

Given the cost model above, the data center has two control decisions at each time: determining n_k, the number of active servers in every time frame, and assigning arriving jobs to servers, i.e., determining $\lambda_{i,k}$ such that $\Sigma_{i=1}^{n_k} \lambda_{i,k} = \lambda_k$. All servers are assumed to be homogeneous with constant rate capacity $\mu > 0$. Modeling heterogeneous servers is possible but the online problem will become more complicated [26] , which is out of the scope of this paper.

The goal of the data center is to determine n_k and $\lambda_{i,k}$ to minimize the cost incurred during $[0, K]$, which is modeled as follows:

$$\min \sum_{k=1}^{K} \sum_{i=1}^{n_k} f(\lambda_{i,k}) + \beta \sum_{k=1}^{K} (n_k - n_{k-1})^+ \tag{2}$$

$$s.t. \begin{cases} 0 \le \lambda_{i,k} \le \lambda_k \\ \Sigma_{i=1}^{n_k} \lambda_{i,k} = \lambda_k \\ \mathbb{P}(D_k > \bar{D}) \le \bar{\varepsilon} \,, \end{cases} \tag{3}$$

where the final constraint is introduced to capture the SLA of the data center. We use D_k to represent the steady-state delay during frame k, and $(\bar{D}, \bar{\epsilon})$ to represent an SLA of the form "the probability of a delay larger than $\bar{D}$ must be bounded by probability $\bar{\varepsilon}$".

This model generalizes the data center optimization problem from [11] by accounting for the additional SLA constraint. The specific values in this constraint are determined by the stochastic variability at the fast time-scale. In particular, we derive (for a variety of workload models) a sufficient constraint $n_k \ge \frac{C_k(\bar{D},\bar{\varepsilon})}{\mu}$ such that

$$n_k \ge \frac{C_k(\bar{D}, \bar{\varepsilon})}{\mu} \Longrightarrow \mathbb{P}(D_k > \bar{D}) \le \bar{\varepsilon} \,. \tag{4}$$

Here, μ is the constant rate capacity of each server and $C_k(\bar{D}, \bar{\varepsilon})$ is to be determined for each considered arrival model. One should interpret $C_k(\bar{D}, \bar{\varepsilon})$ as the overall effective capacity/bandwidth needed in the data center such that the SLA delay constraint is satisfied within frame k.

Note that the new constraint is only sufficient for the original SLA constraint. The reason is that $C_k(\bar{D}, \bar{\varepsilon})$ will be computed, in the next section, from upper bounds on the distribution of the transient delay within a frame.

With the new constraint, however, the optimization problem in (2)-(3) can be considerably simplified. Indeed, note that n_k is fixed during each time frame k and the remaining optimization for $\lambda_{i,k}$ is convex. Thus, we can simplify the form of the optimization problem by using the fact that the optimal dispatching strategy $\lambda^*_{i,k}$ is load balancing, i.e., $\lambda^*_{1,k} = \lambda^*_{2,k} = \ldots = \lambda_k/n_k$. This decouples dispatching $\lambda^*_{i,k}$ from capacity planning n_k, and so Eqs. (2)-(3) become:

$$\underline{\text{Data Center Optimization Problem}}$$

$$\min \sum_{k=1}^{K} n_k f(\lambda_k/n_k) + \beta \sum_{k=1}^{K} (n_k - n_{k-1})^+ \tag{5}$$

$$\text{s.t. } n_k \ge \frac{C_k(\bar{D}, \bar{\varepsilon})}{\mu} \,.$$

Note that (5) is a convex optimization, since $n_k f(\lambda_k/n_k)$ is the perspective function of the convex function $f(\cdot)$.

As we have already pointed out, the key difference between the optimization above, and that of [11], is the SLA constraint. However, this constraint plays a key role in the current paper. It is this constraint that provides a bridge between the slow time-scale and fast time-scale models. Specifically, the fast time-scale model uses large deviations and stochastic network calculus techniques to calculate $C_k(\bar{D}, \bar{\varepsilon})$.

2.4. *Algorithms for Dynamic Resizing*

Though the Data Center Optimization Problem described above is convex, in practice it must be solved *online*, i.e., without knowledge of the future workload. Thus, in determining n_k, the algorithm may not have access to the future arrival rates λ_l for $l > k$. This fact makes developing algorithms for dynamic resizing challenging. However, progress has been made recently [11, 30].

Deriving algorithms for this problem is not the goal of the current paper. Thus, we make use of a recent algorithm called Lazy Capacity Provisioning (LCP) [11]. We choose LCP because of the strong analytic performance guarantees it provides – LCP provides cost within a factor of 3 of optimal for any (even adversarial) workload process.

LCP works as follows. Let $(n^L_{k,1}, \ldots, n^L_{k,k})$ be the solution vector to the following optimization problem

$$\begin{aligned} \min \sum_{l=1}^{k} n_l f(\lambda_l/n_l) + \beta \sum_{l=1}^{k} (n_l - n_{l-1})^+ \\ \text{s.t. } n_l \geq \frac{C_l(\bar{D}, \bar{\varepsilon})}{\mu}\ ,\ n_0 = 0\ . \end{aligned}$$

Similarly, let $(n^U_{k,1}, \ldots, n^U_{k,k})$ be the solution vector to the following optimization problem

$$\begin{aligned} \min \sum_{l=1}^{k} n_l f(\lambda_l/n_l) + \beta \sum_{l=1}^{k} (n_{l-1} - n_l)^+ \\ \text{s.t. } n_l \geq \frac{C_l(\bar{D}, \bar{\varepsilon})}{\mu}\ ,\ n_0 = 0\ . \end{aligned}$$

Denote $(n)^b_a = \max(\min(n, b), a)$ as the projection of n into the closed interval $[a, b]$. Then LCP can be defined using $n^L_{k,k}$ and $n^U_{k,k}$ as follows. Informally, LCP stays "lazily" between the upper bound $n^U_{k,k}$ and the lower bound $n^L_{k,k}$ in all frames.

Lazy Capacity Provisioning, LCP

Let $n^{LCP} = (n^{LCP}_0, \ldots, n^{LCP}_K)$ denote the vector of active servers under LCP. This vector can be calculated online using the following forward recurrence relation:

$$n^{LCP}_k = \begin{cases} 0, & k \leq 0 \\ (n^{LCP}_{k-1})^{n^U_{k,k}}_{n^L_{k,k}}, & 1 \leq k \leq K\ . \end{cases}$$

Note that, in [11], LCP is introduced and analyzed for the optimization from Eq. (5) without the SLA constraint. However, it is easy to see that the algorithm and performance guarantee extend to our setting. Specifically, the guarantees on LCP hold in our setting because the SLA constraint can be removed by defining the operating cost to be ∞ instead of $n_k f(\lambda_k/n_k)$ when $n_k < C_k(\bar{D}, \bar{\varepsilon})/\mu$.

A last point to highlight about LCP is that, as described, it does not use any predictions about the workload in future frames. Such information could clearly be beneficial, and can be incorporated into LCP if desired, see [11].

3. Fast Time-scale Model

Given the model of the slow time-scale in the previous section, we now zoom in to give a description for the fast time-scale model. By "fast" time-scale, we mean the time-scale at which requests arrive, e.g., on the order of 1-10 seconds, as opposed to the "slow" time-scale (frame length) at which dynamic resizing decisions are made by the data center e.g., on the order of 5-15 minutes. To model the fast time-scale, we evenly break each frame from the slow time-scale into "slots" $t \in \{1, \ldots, U\}$, such that frame_length $= U \cdot$ slot_length.

We consider a variety of models for the workload process at this fast time-scale, including both light-tailed models with various degrees of burstiness, as well as heavy-tailed models that exhibit self-similarity. In all cases, our assumption is that the workload is stationary over the slots that make up each time frame.

The goal of this section is to derive the value of $C_k(\bar{D}, \bar{\varepsilon})$ in the constraint $n_k \geq \frac{C_k(\bar{D},\bar{\varepsilon})}{\mu}$ from Eq. (4), and thus enable an interface between the fast and slow time-scales by parameterizing the Data Center Optimization Problem from Eq. (5) for a broad range of workloads.

Note that throughout this section we suppress frame's subscript k for n_k, λ_k, C_k, and D_k, and focus on a generic frame.

Our approach for deriving the SLA constraint for the Data Center Optimization Problem will be to first derive an "aggregation property" which allows the data center to be modeled as a single server, and to then derive bounds on the distribution of the transient delay under a variety of arrival processes.

Note that, in this paper, for simplicity of exposition, the arrival and server models in the fast time-scale have been defined such that stationary increments are implicitly assumed. This is a mild assumption given the length of frames defined by the slow time-scale, and is satisfied by the traces used in Section 4. However, if one would like to study non-stationary processes, the results in this paper can be extended. The resulting bounding functions will generally be dynamic and time-dependent.

3.1. An Aggregation Property

To get intuition for the aggregation property we prove, observe that if the arrival process were modeled as Poisson and job sizes were exponential, then an "aggregation property" would be immediate, since the response time distribution only depends on the load. Hence the SLA could be derived by considering a single server. Outside of this simple case, however, we need to derive a suitable single server approximation.

The aggregation result that we derive and apply is formulated in the framework of stochastic network calculus [31], and so we begin by briefly introducing this framework.

Denote the cumulative arrival (workload) process at the data center's dispatcher by $A(t)$. That is, for each slot $t = 1, \ldots, U$, $A(t)$ counts the total number of jobs arrived in the time interval $[0, t]$. Depending on the total number n of active servers, the arrival process is dispatched into the sub-arrival processes $A_i(t)$ with $i = 1, \ldots, n$ such that $A(t) = \sum_i A_i(t)$. The cumulative response processes from the servers are denoted by $R_i(t)$, whereas the total cumulative response process from the data center is denoted by $R(t) = \sum_i R_i(t)$. All arrival and response processes are assumed to be non-negative, non-decreasing, and left-continuous, and satisfy the initial condition $A(0) = R(0) = 0$. For convenience we use the bivariate extensions $A(s, t) := A(t) - A(s)$ and $R(s, t) := R(t) - R(s)$.

The service provided by a server is modeled in terms of probabilistic lower bounds using the concept of a stochastic service process. This is a bivariate random process $S(s, t)$ which is non-negative, non-decreasing, and left-continuous. Formally, a server is said to guarantee a (stochastic) service process $S(s, t)$ if for *any* arrival process $A(t)$ the corresponding response process $R(t)$ from the server satisfies for all $t \geq 0$

$$R(t) \geq A * S(t) \,, \tag{6}$$

where '$*$' denotes the min-plus convolution operator, i.e., for two (random) processes $A(t)$ and $S(s, t)$,

$$A * S(t) := \inf_{0 \leq s \leq t} \{A(s) + S(s, t)\} \,. \tag{7}$$

The inequality in (6) is assumed to hold almost surely. Note that the lower bound set by the service process is invariant to the arrival processes.

We are now ready to state the aggregation property. The proof is deferred to Section 5.

Lemma 1. *Consider an arrival process $A(t)$ which is dispatched to n servers. Each server i is work-conserving with constant rate capacity $\mu > 0$. Arrivals are dispatched deterministically across the servers such that each server i receives a fraction $\frac{1}{n}$ of the arrivals. Then, the system has service process $S(s, t) = n\mu(t - s)$, i.e., $R(t) \geq A * S(t)$.*

The significance of the Lemma is that if the SLA is verified for the virtual server with arrival process $A(t)$ and service process $S(s, t)$, then the SLA is verified for each of the n servers. We point out that the lemma is based on the availability of a dispatching policy with equal weights for homogenous servers.

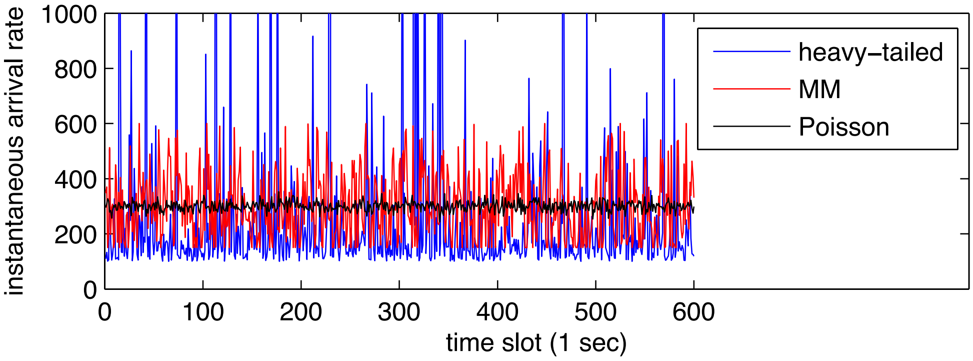


Figure 1: Three synthetically generated traces within 1 frame, with $\lambda = 300$ of Poisson, Markov-Modulated (MM) ($T = 1$, $\lambda_l = 0.5\lambda$ and $\lambda_h = 2\lambda$), and heavy-tailed arrivals ($b = \lambda/3$ and $\alpha = 1.5$).

3.2. Arrival Processes

Now that we can reduce the study of the multi-server system to the study of a single server system using Lemma 1, we can move to characterizing the impact of the arrival process on the SLA constraint in the Data Center Optimization Problem.

In particular, the next step in deriving the SLA constraint $n \geq \frac{C(\bar{D},\bar{\varepsilon})}{\mu}$ is to derive a bound on the distribution of the delay at the virtual server with arrival process $A(t)$ and service process $S(s,t) = C(\bar{D},\bar{\varepsilon})(t-s)$, i.e.,

$$\mathbb{P}\Big(D(t) > \bar{D}\Big) \leq \varepsilon(\bar{D}) \,. \tag{8}$$

It is important to observe that the violation probability ε holds for the *transient virtual* delay process $D(t)$, which is defined as $D(t) := \inf\{d : A(t-d) \leq R(t)\}$, and which models the delay spent in the system by the job leaving the system, if any, at time t. By this definition, and using the servers' homogeneity and also the deterministic splitting of arrivals from Lemma 1, the virtual delay for the aggregate virtual server is the same as the virtual delay for the individual servers. This fact guarantees that an SLA constraint on the virtual server implicitly holds for the individual servers as well. Moreover, the violation probability ε in Eq. (8) is derived so that it is time invariant, which implies that it bounds the distribution of the stead-state delay $D = \lim_{t\to\infty} D(t)$ as well. Therefore, the value of $C(\bar{D},\bar{\varepsilon})$ can be finally computed by solving the implicit equation $\varepsilon(\bar{D}) = \bar{\varepsilon}$.

In the following, we follow the outline above to compute $C(\bar{D},\bar{\varepsilon})$ for light- and heavy-tailed arrival processes. Interested readers may refer to the technical report [32] for details. Figure 1 depicts examples of the three types of arrival processes we consider in 1 frame: Poisson, Markov-Modulated (MM), and heavy-tailed arrivals. In all three cases, the mean arrival rate is $\lambda = 300$. The figure clearly illustrates the different levels of burstiness of the three traces.

3.2.1. Light-tailed Arrivals

We consider two examples of light-tailed arrival processes: Poisson and Markov-Modulated (MM) processes.

Poisson Arrivals

We start with the case of Poisson processes, which are characterized by a low level of burstiness, due to the independent increments property. The following proposition, providing the tail of the virtual delay, is a minor variation of a result from [33]; for the proof see [32].

Proposition 1. *Let $A(t)$ be a Poisson process with some rate $\lambda > 0$, and define*

$$\theta^* := \sup\left\{\theta > 0 : \frac{\lambda}{\theta}\left(e^{\theta} - 1\right) \leq C(\bar{D},\bar{\varepsilon})\right\} \,. \tag{9}$$

Then a bound on the transient delay process is given for all $t \geq 0$ by

$$\mathbb{P}\left(D(t) > \bar{D}\right) \leq e^{-\theta^* C(\bar{D},\bar{\varepsilon})\bar{D}} := \varepsilon(\bar{D}) \,. \tag{10}$$

Solving for $C(\bar{D}, \bar{\varepsilon})$ by setting the violation probability $\varepsilon(\bar{D})$ equal to $\bar{\varepsilon}$ yields the implicit solution

$$C(\bar{D}, \bar{\varepsilon}) = -\frac{1}{\theta^* \bar{D}} \log \bar{\varepsilon} \,.$$

Further, using the monotonicity of the function $\frac{\lambda}{\theta}\left(e^{\theta} - 1\right)$ in $\theta > 0$ we immediately get the explicit solution

$$C(\bar{D}, \bar{\varepsilon}) = \frac{K}{\log(1 + K)} \lambda \,, \tag{11}$$

where

$$K = -\frac{\log \bar{\varepsilon}}{\lambda \bar{D}} \,.$$

Markov-Modulated Arrivals

Consider now the case of Markov-Modulated (MM) processes which, unlike the Poisson processes, do not necessarily have independent increments. The key feature for the purposes of this paper is that the burstiness of MM processes can be arbitrarily adjusted.

We consider a simple MM processes with two states. Let a discrete and homogeneous Markov chain $x(s)$ with two states denoted by 'low' and 'high', and transition probabilities p_h and p_l between the 'low' and 'high' states, and vice-versa, respectively. Assuming that a source produces at some constant rates $\lambda_l > 0$ and $\lambda_h > \lambda_l$ while the chain $x(s)$ is in the 'low' and 'high' states, respectively, then the corresponding MM cumulative arrival process is

$$A(t) = \sum_{s=1}^{t} \left(\lambda_l I_{\{x(s)=\text{'}low\text{'}\}} + \lambda_h I_{\{x(s)=\text{'}high\text{'}\}}\right) \,, \tag{12}$$

where $I_{\{\cdot\}}$ is the indicator function. The average rate of $A(t)$ is $\lambda = \frac{p_l}{p_h+p_l}\lambda_l + \frac{p_h}{p_h+p_l}\lambda_h$.

To adjust the burstiness level of $A(t)$ we introduce the parameter $T := \frac{1}{p_h} + \frac{1}{p_l}$, which is the average time for the Markov chain $x(s)$ to change states twice. We note that the higher the value of T is, the higher the burstiness level becomes (the time periods whilst x(s) spends in the 'high' or 'low' states get longer and longer).

To compute the delay bound let us construct the matrix

$$\Psi(\theta) = \begin{pmatrix} (1 - p_h)e^{\theta\lambda_l} & p_h e^{\theta\lambda_h} \\ p_l e^{\theta\lambda_l} & (1 - p_l)e^{\theta\lambda_h} \end{pmatrix} ,$$

for some $\theta > 0$ and consider its spectral radius

$$\lambda(\theta) := \frac{(1 - p_h)e^{\theta\lambda_l} + (1 - p_l)e^{\theta\lambda_h} + \sqrt{\Delta}}{2} \,, \tag{13}$$

where $\Delta = \left((1 - p_h)e^{\theta\lambda_l} - (1 - p_l)e^{\theta\lambda_h}\right)^2 + 4p_h p_l e^{\theta(\lambda_l+\lambda_h)}$. Let also

$$K(\theta) := \max\left\{\frac{p_h e^{\theta\lambda_h}}{\lambda(\theta) - (1 - p_h)e^{\theta\lambda_l}}, \frac{\lambda(\theta) - (1 - p_h)e^{\theta\lambda_l}}{p_h e^{\theta\lambda_h}}\right\} \,. \tag{14}$$

The two terms are the ratios of the elements of the right-eigenvector of the matrix $\Psi(\theta)$. Also, let

$$\theta^* := \sup\left\{\theta > 0 : \frac{1}{\theta} \log \lambda(\theta) \leq C(\bar{D}, \bar{\varepsilon})\right\} \,.$$

Using these constructions, and also the constant rate service assumption, a result on the backlog bound from [31], pp. 340, immediately lends itself to the corresponding result on the virtual delay:

Proposition 2. *Consider a MM cumulative arrival process as defined in (12), with the $\lambda(\theta)$ given in (13), and $K(\theta)$ given in (14), then a bound on the transient delay process is*

$$\mathbb{P}\Big(D(t) > \bar{D}\Big) \leq K(\theta^*)e^{-\theta^* C(\bar{D},\bar{\varepsilon})\bar{D}} := \varepsilon(\bar{D}) \, .$$

Setting the violation probability $\varepsilon(\bar{D})$ equal to $\bar{\varepsilon}$ in Theorem 2 yields the implicit solution

$$C(\bar{D},\bar{\varepsilon}) = -\frac{1}{\theta^*\bar{D}} \log \frac{\bar{\varepsilon}}{K(\theta^*)} \, . \tag{15}$$

3.3. Heavy-tailed and Self-similar Arrivals

We now consider the class of heavy-tailed and self-similar arrival processes. These processes are fundamentally different from light-tailed processes in that deviations from the mean increase in time and decay in probability as a power law, i.e., more slower than the exponential.

We consider in particular the case of a source generating jobs in every slot according to i.i.d. Pareto random variables X_i with tail distribution for all $x \geq b$:

$$\mathbb{P}\left(X_i > x\right) = (x/b)^{-\alpha} \, , \tag{16}$$

where $1 < \alpha < 2$. X has finite mean $E[X] = \alpha b/(\alpha - 1)$ and infinite variance. For the corresponding bound on the transient delay we reproduce a result from [34].

Proposition 3. *Consider a source generating jobs in every slot according to i.i.d. Pareto random variables X_i, with the tail distribution from (16). The bound on the transient delay is*

$$\mathbb{P}\Big(D(t) > \bar{D}\Big) \leq K\Big(C(\bar{D},\bar{\varepsilon})\bar{D}\Big)^{1-\alpha} := \varepsilon(\bar{D}) \, , \tag{17}$$

where

$$K = \inf_{1<\gamma<\frac{C(\bar{D},\bar{\varepsilon})}{\lambda}} \left\{ \left(\frac{C(\bar{D},\bar{\varepsilon})}{\gamma} - \lambda\right)^{-1} \frac{\alpha\gamma^{\frac{\alpha-1}{\alpha}}}{(\alpha-1)\log\gamma} \right\} .$$

Setting the violation probability $\varepsilon(\bar{D})$ equal to $\bar{\varepsilon}$, we get the implicit solution

$$\inf_{1<\gamma<\frac{C(\bar{D},\bar{\varepsilon})}{\lambda}} \left\{ \frac{\gamma}{C(\bar{D},\bar{\varepsilon})^{\alpha-1}\Big(C(\bar{D},\bar{\varepsilon}) - \gamma\lambda\Big)} \frac{\gamma^{\frac{\alpha-1}{\alpha}}}{\log\gamma^{\frac{\alpha-1}{\alpha}}} \right\} = \bar{\varepsilon}\bar{D}^{\alpha-1} \, . \tag{18}$$

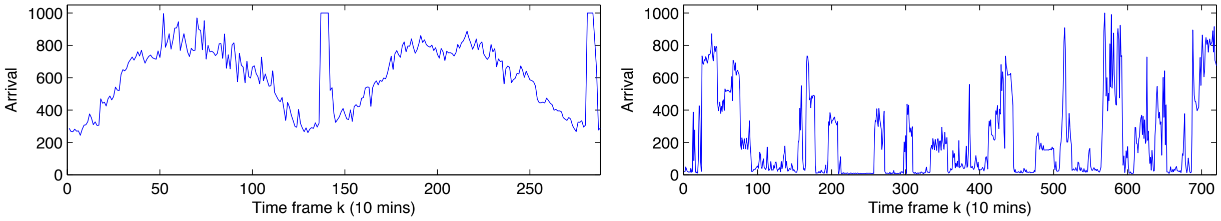
4. Case Studies

Given the model described in the previous two sections, we are now ready to explore the potential of dynamic resizing in data centers, and how this potential depends on the interaction between non-stationarities at the slow time-scale and burstiness/self-similarity at the fast time-scale. Our goal in this section is to provide insight into which workloads dynamic resizing is valuable for. To accomplish this, we provide a mixture of analytic results and trace-driven numerical simulations in this section.

It is important to note that the case studies that follow depend fundamentally on the modeling performed so far in the paper, which allows us to capture and adjust independently, both fast time-scale and slow time-scale properties of the workload. The generality of our model framework enables thus a rigorous study of the impact of the workload on value of dynamic resizing.

4.1. Setup

Throughout the experimental setup, our aim is to choose parameters that provide conservative estimates of the case savings from dynamic resizing. Thus, one should interpret the savings shown as a lower-bound on the potential savings.



(a) Hotmail

(b) MSR

Figure 2: Illustration of the traces used for numerical experiments.

Model Parameters

The time frame for adapting the number of servers n_k is assumed to be 10 min, and each time slot is assumed to be 1 s, i.e., $U = 600$. When not otherwise specified, we assume the following parameters for the data center SLA agreement: the (virtual) delay upper bound $\bar{D} = 200$ms, and the delay violation probability $\bar{\varepsilon} = 10^{-3}$.

The cost is characterized by the two parameters of e_0 and e_1, and the switching cost β. We choose units such that the fixed energy cost is $e_0 = 1$. The load-dependent energy consumption is set to $e_1 = 0$, because the energy consumption of current servers with typical utilization level is dominated by the fixed costs [28, 4, 29]. Note that adjusting e_0 and e_1 changes the magnitude of potential savings under dynamic resizing, but does not affect the qualitative conclusions about the impact of the workload. So, due to space constraints, we fix these parameters during the case studies.

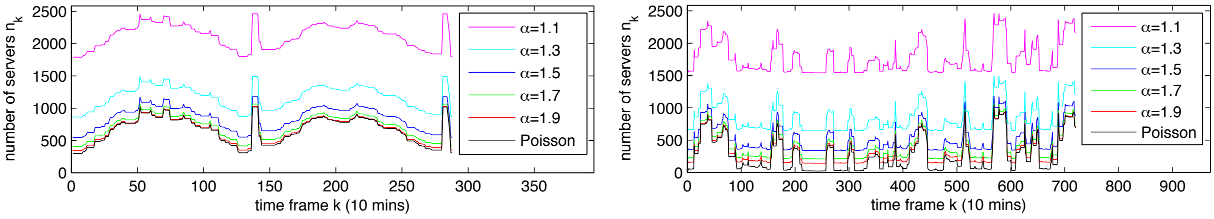
The normalized switching cost β/e_0 measures the duration a server must be powered down to outweigh the switching cost. Unless otherwise specified, we use $\beta = 6$, which corresponds to the energy consumption for one hour (six frames). This was chosen as an estimate of the time a server should sleep so that the wear-and-tear of power cycling matches that of operating [20, 11].

Workload Information

For the slow time-scale, i.e., λ_k, the workloads are drawn from two real-world data center traces. The first set of traces is from Hotmail, a large email service running on tens of thousands of servers. We used traces from 8 such servers over a 48-hour period, starting at midnight (PDT) on Monday August 4 2008 [18]. The second set of traces is taken from 6 RAID volumes at MSR Cambridge. The traced period was 1 week starting from 5PM GMT on the 22nd February 2007 [18]. Thus, these activity traces represent a service used by millions of users and a small service used by hundreds of users.

The original load is averaged over each frame, on the order of 10 minutes, and the traces are normalized as peak load λ_{peak}=1000, which are visualized in Figure 2. Both sets of traces show strong diurnal properties and have peak-to-mean ratios (PMRs) of 1.64 and 4.64 for Hotmail and MSR respectively. The traces are then used as the values of λ_k in the corresponding frame k. We also adjust the peak-to-mean ratio for some experiments by scaling λ_k as $\hat{\lambda}_k = c(\lambda_k)^{\gamma}$, varying γ and adjusting c to keep the mean constant. In this way, we can simulate a variety of realistic arrival processes with discrete-event simulation for the optimization problem (Eq. (5)), fix a cost function, and then use a scheduling algorithm for the slow time-scale, at each time step using the fast time-scale as input.

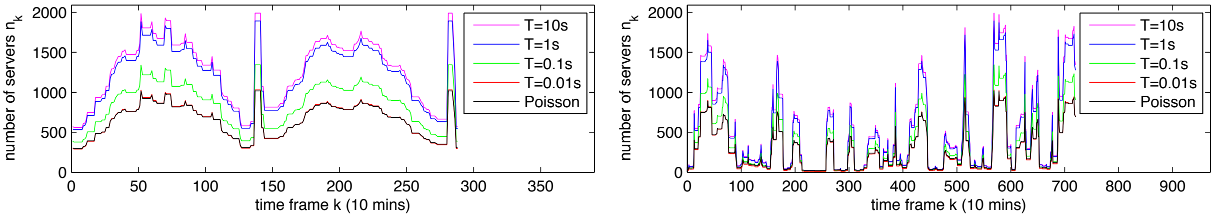
To simulate the stochastic burstiness of the workload at the fast time-scale model, at each time step for the optimization problem (Eq. (5)), we adapt the workload based on the mean arrival rate in each frame (λ_k) to parameterize the arrival processes, $A(t)$. For the stochastic processes being simulated, we consider two examples of light-tailed arrival processes: Poisson and Markov-Modulated (MM) processes, as well as the class of heavy-tailed and self-similar arrival processes. To parameterize the MM processes, we take $\lambda_l = .5\lambda$, $\lambda_h = 2\lambda$, and we adjust the burst parameter T while keeping λ fixed for each process. To parameterize the heavy-tailed processes we adjust the tail index α for each process, and b in Eq. (16) is adapted accordingly in order to also keep λ fixed. Unless otherwise stated, we fix $\alpha = 1.5$ and $T = 1$.



(a) Hotmail

(b) MSR

Figure 3: Impact of burstiness on provisioning n_k for heavy-tailed arrivals.



(a) Hotmail

(b) MSR

Figure 4: Impact of burstiness on provisioning n_k for MM arrivals.

Comparative Benchmark

We contrast three designs: (i) the optimal dynamic resizing, (ii) dynamic resizing via LCP, and (iii) the optimal 'static' provisioning.

The results for the optimal dynamic resizing should be interpreted as characterizing the potential of dynamic resizing. But, realizing this potential is a challenge that requires both sophisticated online algorithms and excellent predictions of future workloads.[1]

The results for LCP should be interpreted as one example of how much of the potential for dynamic resizing can be attained with an online algorithm. One reason for choosing LCP is that it does not rely on predicting the workload in future frames, and thus provides a conservative bound on the achievable cost savings.

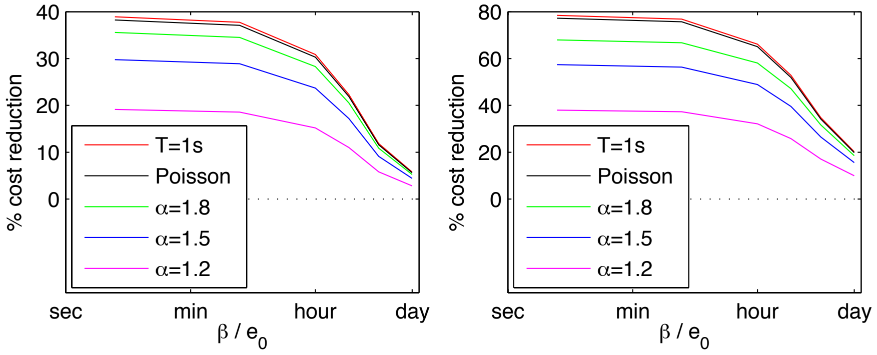
The results for the optimal static provisioning should be taken as an optimistic benchmark for today's data centers, which typically do not use dynamic resizing. We consider the cost incurred by an optimal static provisioning scheme that chooses a constant number of servers that minimizes the costs incurred based on full knowledge of the entire workload. This policy is clearly not possible in practice, but it provides a *very conservative* estimate of the savings from right-sizing since it uses perfect knowledge of all peaks and eliminates the need for overprovisioning in order to handle the possibility of flash crowds or other traffic bursts.

4.2. Results

Our experiments are organized to illustrate the impact of a wide variety of parameters on the cost savings attainable via dynamic resizing, e.g., the burstiness parameter on the fast time-scale model, i.e., α and T, and the peak-to-mean ratio reflected by the slow time-scale model λ_k. The goal is to understand for which workloads dynamic resizing can provide large enough cost savings to warrant the extra implementation complexity. Remember, our setup is designed so that the cost savings illustrated is a conservative estimate of the true cost savings provided by dynamic resizing.

The SLA constraint $n_k \geq \frac{C_k(\bar{D},\bar{\varepsilon})}{\mu}$ from Eq. (4) provides a bridge between the slow time-scale and fast time-scale models, i.e., $C_k(\bar{D},\bar{\varepsilon})$ will be computed from upper bounds on the distribution of the transient delay within a frame. For given input parameters in Eq. (5), i.e., $\lambda_k, \bar{D}, \bar{\varepsilon}$, depending on different arrival processes being simulated, we can

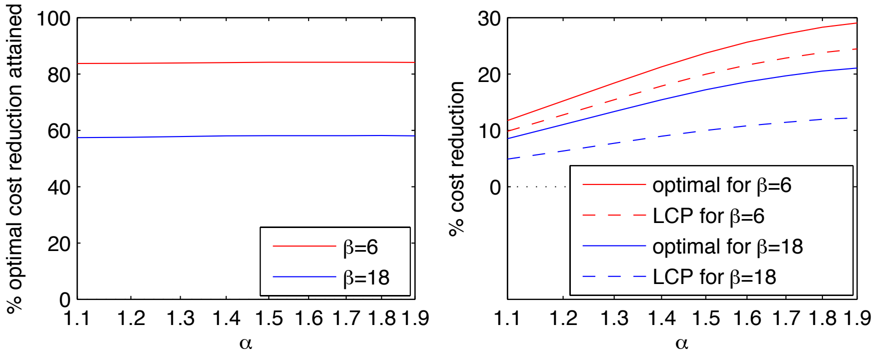
[1] Note that short-term predictions of workload demand within 24 hours can be quite accurate [35, 29].



(a) Hotmail

(b) MSR

Figure 5: Impact of burstiness on the cost savings of dynamic resizing for different switching costs, β.



(a) cost savings

(b) relative savings of LCP

Figure 6: Impact of burstiness on the performance of LCP in the Hotmail trace.

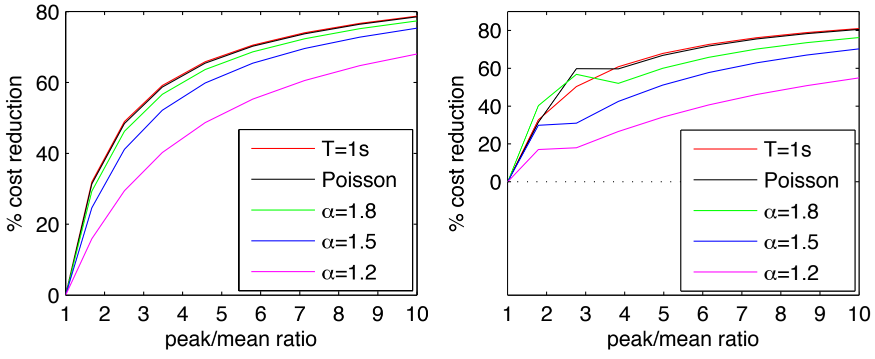
use Eq. (11), (15) (with given T), or (18) (with given α), in the fast time-scale model to derive the numerical result of capacity constraint, $C_k(\bar{D}, \bar{\varepsilon})$, in the slow time-scale model. In this way, the theoretical portion of the paper is used in the numerical results, and by solving the optimization problem (Eq. (5)), we can finally derive the numerical analysis as follows.f

The Role of Burstiness

A key goal of our model is to expose the impact of burstiness on dynamic resizing, and so we start by focusing on that parameter. Recall that we can vary burstiness in both the light-tailed and heavy-tailed settings using T for MM arrivals and α for heavy-tailed arrivals.

The impact of burstiness on provisioning: A priori, one may expect that burstiness can be beneficial for dynamic resizing, since it indicates that there are periods of low load during which energy may be saved. However, this is not actually true since resizing decisions must be made at the slow time-scale while burstiness is a characteristic of the fast time-scale. Thus, burstiness is actually detrimental for dynamic resizing, since it means that the provisioning decisions made on the slow time-scale must be made with the bursts in mind, which results in a larger number of servers needed to be provisioned for the same average workload. This effect can be seen in Figures 3 and 4, which show the optimal dynamic provisioning as α and T vary. Recall that burstiness increases as α decreases and T increases.

The impact of burstiness on cost savings: The larger provisioning created by increased burstiness manifests itself in the cost savings attainable through dynamic capacity provisioning as well. This is illustrated in Figure 5,



(a) Hotmail (b) MSR

Figure 7: Impact of peak-to-mean ratio on the cost savings of the optimal dynamic resizing.

which shows the cost savings of the optimal dynamic provisioning as compared to the optimal static provisioning for varying α and T as a function of the switching cost β.

The impact of burstiness on LCP: Interestingly, though Figure 5 shows that the potential of dynamic resizing is limited by increased burstiness, it turns out that the relative performance of LCP is not hurt by burstiness. This is illustrated in Figure 6, which shows the percent of the optimal cost savings that LCP achieves. Importantly, it is nearly perfectly flat as the burstiness is varied.

The Role of the Peak-to-Mean Ratio

The impact of the peak-to-mean ratio on the potential benefits of dynamic resizing is quite intuitive: if the peak-to-mean ratio is high, then there is more opportunity to benefit from dynamically changing capacity. Figure 7 illustrates this well-known effect. The workload for the figure is generated from the traces by scaling λ_k as $\hat{\lambda}_k = c(\lambda_k)^\gamma$, varying γ and adjusting c to keep the mean constant.

In addition to illustrating that a higher peak-to-mean ratio makes dynamic resizing more valuable, Figure 7 also highlights that there is a strong interaction between burstiness and the peak-to-mean ratio, where if there is significant burstiness the benefits that come from a high peak-to-mean ratio may be diminished considerably.

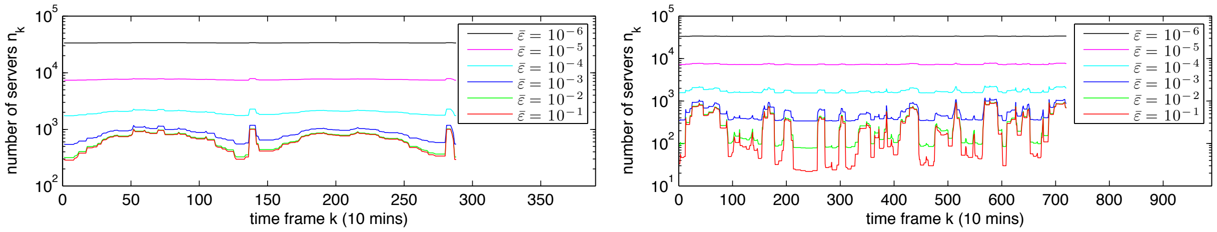
The Role of the SLA

The SLA plays a key role in the provisioning of a data center. Here, we show that the SLA can also have a strong impact on whether dynamic resizing is valuable, and that this impact depends on the workload. Recall that in our model the SLA consists of a violation probability $\bar{\varepsilon}$ and a delay bound $\bar{D}$. We deal with each of these in turn.

Figures 8 and 9 highlight the role the violation probability $\bar{\varepsilon}$ has on the provisioning of n_k under the optimal dynamic resizing in the cases of heavy-tailed and MM arrivals. Interestingly, we see that there is a significant difference in the impact of $\bar{\varepsilon}$ depending on the arrival process. As $\bar{\varepsilon}$ gets smaller in the heavy-tailed case the provisioning gets significantly flatter, until there is almost no change in n_k over time. In contrast, no such behavior occurs in the MM case and, in fact, the impact of $\bar{\varepsilon}$ is quite small. This difference is a fundamental effect of the "heaviness" of the tail of the arrivals, i.e., a heavy tail requires significantly more capacity in order to counter a drop in $\bar{\varepsilon}$.

This contrast between heavy- and light-tailed arrivals is also evident in Figure 11, which highlights the cost savings from dynamic resizing in each case as a function of $\bar{\varepsilon}$. Interestingly, the cost savings under light-tailed arrivals is largely independent of $\bar{\varepsilon}$, while under heavy-tailed arrivals the cost savings is monotonically increasing with $\bar{\varepsilon}$.

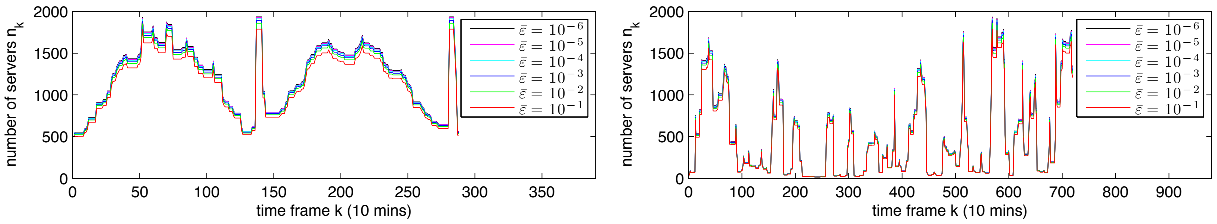
The second component of the SLA is the delay bound $\bar{D}$. The impact of $\bar{D}$ on provisioning is much less dramatic. We show an example in the case of heavy-tailed arrivals in Figure 10. Not surprisingly, the provisioning increases as $\bar{D}$ drops. However, the flattening observed as a result of $\bar{\varepsilon}$ is not observed here. The case of MM arrivals is qualitatively the same, and so we do not include it.



(a) Hotmail

(b) MSR

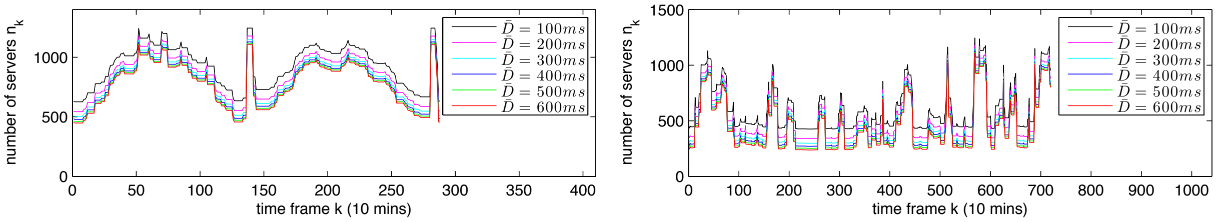
Figure 8: Impact of $\bar{\varepsilon}$ on provisioning n_k for heavy tailed arrivals.



(a) Hotmail

(b) MSR

Figure 9: Impact of $\bar{\varepsilon}$ on provisioning n_k for MM arrivals.



(a) Hotmail

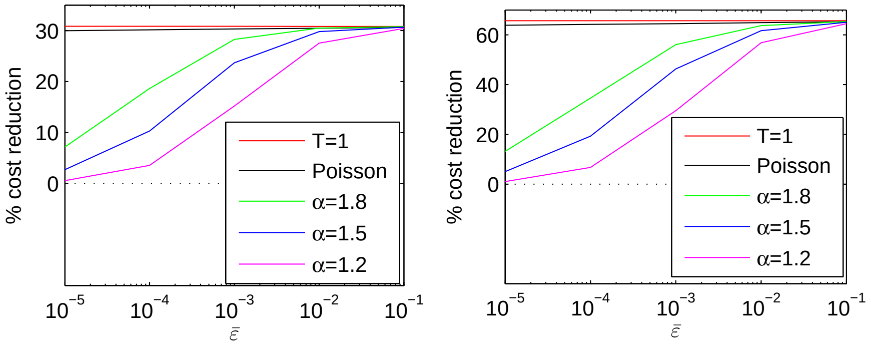
(b) MSR

Figure 10: Impact of $\bar{D}$ on provisioning n_k for heavy tailed arrivals.

When is Dynamic Resizing Valuable?

Now, we are finally ready to address the question of when (i.e., for what workloads) is dynamic resizing valuable. To address this question, we must look at the interaction between the peak-to-mean ratio and the burstiness. Our goal is to provide a concrete understanding of for which (peak-to-mean, burstiness, SLA) settings the potential savings from dynamic resizing is large enough to warrant implementation. Figures 12–14 focus on this question. Our hope is that these figures highlight that a precursor to any debate about the value of dynamic resizing must be a joint understanding of the expected workload characteristics and the desired SLA, since for any fixed choices of two of these parameters (peak-to-mean, burstiness, SLA), the third can be chosen so that dynamic resizing does or does not provide significant cost savings for the data center.

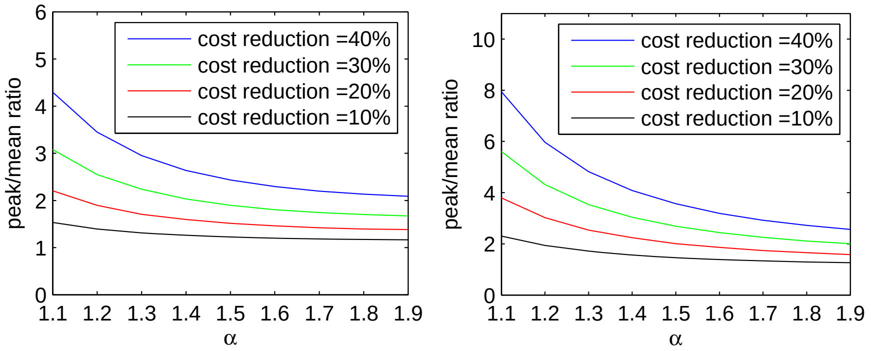
Starting with Figure 12, we see a set of curves for different levels of cost savings. The interpretation of the figures is that below (above) each curve the savings from optimal dynamic resizing is smaller (larger) than the specified value for the curve. Thus, for example, if the peak-to-mean ratio is 2 in the Hotmail trace, a 10% cost savings is possible for all levels of burstiness, but a 30% cost savings is only possible for $\alpha > 1.5$. However, if the peak-to-mean ratio is 3, then a 30% cost savings is possible for all levels of burstiness. It is difficult to say what peak-to-mean and burstiness settings are "common" for data centers, but as a point of reference, one might expect large-scale services to have a peak-to-mean ratio similar to that of the Hotmail trace, i.e., around 1.5-2.5; and smaller scale services to have peak-to-mean ratios similar to that of the MSR trace, i.e., around 4-6. The burstiness also can vary widely, but as a rough



(a) Hotmail

(b) MSR

Figure 11: Impact of $\bar{\varepsilon}$ on the cost savings of dynamic resizing.



(a) Hotmail

(b) MSR

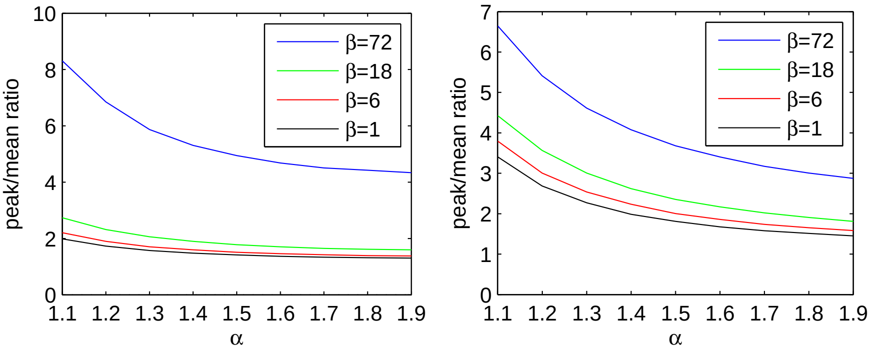
Figure 12: Characterization of burstiness and peak-to-mean ratio necessary for dynamic resizing to achieve different levels of cost reduction.

estimate, one might expect α to be around 1.4-1.6.

Of course, many of the settings of the data center will effect the conclusions illustrated in Figure 12. Two of the most important factors to understand the effects of are the switching cost, β, and the SLA, particularly $\bar{\varepsilon}$.

Figure 13 highlights the impact of the magnitude of the switching costs on the value of dynamic resizing. The curves represent the threshold on peak-to-mean ratio and burstiness necessary to obtain 20% cost savings from dynamic resizing. As the switching costs increase, the workload must have a larger peak-to-mean ratio and/or less burstiness in order for dynamic resizing to be valuable. This is not unexpected. However, what is perhaps surprising is the small impact played by the switching cost. The class of workloads where dynamic resizing is valuable only shrinks slightly as the switching cost is varied from on the order of the cost of running a server for 10 minutes ($\beta = 1$) to running a server for 3 hours ($\beta = 18$).

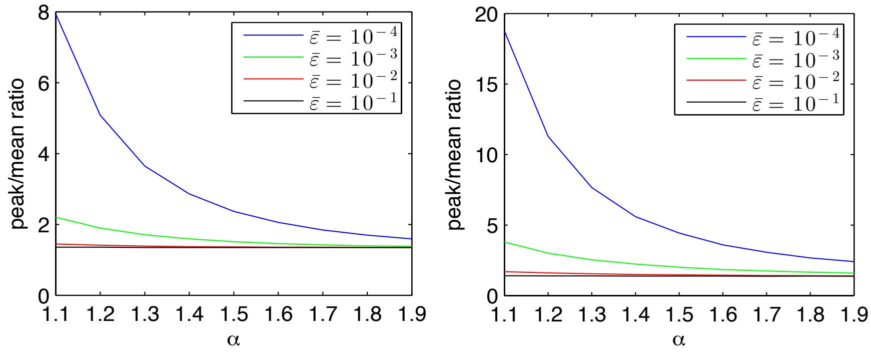
Interestingly, while the impact of the switching costs on the value of dynamic resizing is small, the impact of the SLA is quite large. In particular, the violation probability $\bar{\varepsilon}$ can dramatically affect whether dynamic resizing is valuable or not. This is shown in Figure 14, on which the curves represent the threshold on peak-to-mean ratio and burstiness necessary to obtain 20% cost savings from dynamic resizing. We see that, as the violation probability is allowed to be larger, the impact of the peak-to-mean ratio on the potential of savings from dynamic resizing disappears; and the value of dynamic resizing starts to depend almost entirely on the burstiness of the arrival process. The reason for this can be observed in Figure 8, which highlights that the optimal provisioning n_k becomes nearly flat as $\bar{\varepsilon}$ increases.



(a) Hotmail

(b) MSR

Figure 13: Characterization of burstiness and peak-to-mean ratio necessary for dynamic resizing to achieve 20% cost reduction as a function of the switching cost, β.



(a) Hotmail

(b) MSR

Figure 14: Characterization of burstiness and peak-to-mean ratio necessary for dynamic resizing to achieve 20% cost reduction as a function of the SLA, $\bar{\varepsilon}$.

Supporting Analytic Results

To this point we have focused on numerical simulations, and further we provide analytic support for the behavior we observed in the experiments above. In particular, the following two theorems characterize the impact of burstiness and the SLA $(\bar{D}, \bar{\varepsilon})$ on the value of dynamic resizing under Poisson and heavy-tailed arrivals. This is accomplished by deriving the effect of these parameters on $C(\bar{D}, \bar{\varepsilon})$, which constrains the optimal provisioning n_k. A smaller (larger) $C(\bar{D}, \bar{\varepsilon})$ implies a smaller (larger) provisioning n_k, which in turn implies smaller (larger) costs.

We start providing a result for the case of Poisson arrivals. The proof is given in Section 5.

Theorem 1. *The service capacity constraint from Eq. (11) increases as the delay constraint $\bar{D}$ or the violation probability $\bar{\varepsilon}$ decrease. It also satisfies the scaling law*

$$C(\bar{D}, \bar{\varepsilon}) = \Theta\left(\frac{\bar{D}^{-1} \log \bar{\varepsilon}^{-1}}{\log\left(\bar{D}^{-1} \log \bar{\varepsilon}^{-1}\right)}\right),$$

as $\bar{D}^{-1} \log \bar{\varepsilon}^{-1} \to \infty$.

This theorem highlights that as $\bar{\varepsilon}$ decreases and/or $\bar{D}$ decreases $C(\bar{D}, \bar{\varepsilon})$, and thus the cost of the optimal provisioning, increases. This shows that the observations made in our numeric experiments hold more generally. Perhaps the most interesting point about this theorem, however, is the contrast of the growth rate with that in the case of heavy-tailed arrivals, which is summarized in the following theorem. The proof is given in Section 5.

Theorem 2. *The implicit solution for the capacity constraint from Eq. (18) increases as the delay constraint $\bar{D}$ or the violation probability $\bar{\varepsilon}$ decrease, or the value of α decreases. It also satisfies the scaling law*

$$C(\bar{D}, \bar{\varepsilon}) = \Theta\left(\left(\frac{1}{\bar{\varepsilon}\bar{D}^{\alpha-1}}\right)^{\frac{1}{\alpha}}\right) \tag{19}$$

as $\bar{\varepsilon}\bar{D}^{\alpha-1} \to 0$ for any given $\alpha \in (1, 2)$.

A key observation about this theorem is that the growth rate of $C(\bar{D}, \bar{\varepsilon})$ with $\bar{\varepsilon}$ is much faster than in the case of the Poisson (polynomial instead of logarithmic). This supports what is observed in Figure 11. Additionally, Theorem 2 highlights the impact of burstiness, α, and shows that the behavior we have seen in our experiments holds more generally.

5. Proofs

In this section, we collect the proofs for the results in previous sections. We start with the proof of Lemma 1, the aggregation property used to model the multiserver system with a single service process.

Proof 1 (Proof of Lemma 1). *Fix $t \geq 1$. Because each server i has a constant rate capacity μ, it follows that the bivariate processes $S_i(s,t) = \mu(t-s)$ are service processes for the individual servers (see [31], pp. 167), i.e.,*

$$\begin{aligned} R_i(t) &\geq \inf_{0 \leq s \leq t} \{A_i(s) + \mu(t-s)\} \\ &= \frac{1}{n} \inf_{0 \leq s \leq t} \{A(s) + n\mu(t-s)\} \ , \end{aligned}$$

where $R_i(t)$ is the departure process from server i. In the last line we used the load-balancing dispatching assumption, i.e., $A_i(s) = \frac{1}{n}A(s)$. Adding the terms for $i = 1, \ldots, n$ it immediately follows that

$$\textstyle\sum_i R_i(t) \geq \inf_{0 \leq s \leq t} \{A(s) + n\mu(t-s)\},$$

which shows that the bivariate process $S(s,t) = n\mu(t-s)$ is a service process for the virtual system with arrival process $A(t) = \sum_i A_i(t)$ and departure process $R(t) = \sum_i R_i(t)$.

We next prove the monotonicity and scaling results in Theorems 1 and 2.

Proof 2 (Proof of Theorem 1). *First, note that the monotonicity properties follow immediately from the fact that the function $f(x) = (1+x)^{\frac{1}{x}}$ is non-decreasing. Next, to prove the more detailed scaling laws, simply notice that* $\log(1 + cf(n)) = \Theta(\log f(n))$ *for some non-decreasing function $f(n)$ and a constant $c > 0$. The result follows.*

Proof 3 (Proof of Theorem 2). *We first consider the monotonicity properties and then the scaling law.*

Monotonicity properties: *To prove the monotonicity results on $\bar{D}$ and $\bar{\varepsilon}$, observe that the left hand side (LHS) in the implicit equation from Eq. (18) is a non-increasing function in $C(\bar{D}, \bar{\varepsilon})$ because the range of the infimum expands whereas the function in the infimum decreases, by increasing $C(\bar{D}, \bar{\varepsilon})$. Moreover, the LHS is unbounded at the boundary $C(\bar{D}, \bar{\varepsilon}) = \lambda$. The solution $C(\bar{D}, \bar{\varepsilon})$ is thus non-increasing in both $\bar{D}$ and $\bar{\varepsilon}$.*

Next, to prove monotonicity in α, fix $\alpha_1 \leq \alpha_2$ and denote by C_1 the implicit solution of Eq. (18) for $\alpha = \alpha_1$. In the first step we prove that $C_1\bar{D} \geq 1$. Let C be the solution of

$$\frac{1}{C^{\alpha_1}} = \bar{\varepsilon}\bar{D}^{\alpha_1 - 1},$$

where the LHS was obtained by relaxing the LHS of Eq. (18) (we used that $\gamma > 1$, $C - \gamma\lambda < C$, and $x \geq \log x$ for all $x \geq 1$). Consequently, $C_1 \geq C$, and by assuming that the units are properly scaled such that $\bar{D} \geq 1$, it follows that $C\bar{D} \geq 1$ and hence $C_1\bar{D} \geq 1$.

Secondly, we prove that γ, i.e., the optimal value in the solution of C_1, satisfies $\gamma < e$. Consider the function $f(\gamma) = \frac{\gamma^{a+1}}{a \log \gamma}$ with $a = \frac{\alpha-1}{\alpha}$. If, by contradiction, $\gamma \geq e$, then $f'(\gamma) > 0$ and consequently $f(\gamma)$ is increasing on $[e, \infty)$. Since the function $\frac{1}{C_1 - \gamma\lambda}$ is also increasing in γ, we get a contradiction that γ is the optimal solution as assumed, and hence $\gamma < e$.

Finally, consider the function $g(a) = \frac{\gamma^a}{a \log \gamma}$ with $a = \frac{\alpha-1}{\alpha}$. The previous property $\gamma < e$ implies that $g'(a) \leq 0$ and further that g is non-increasing in a and hence in α as well. Since $\frac{1}{(C_1 \bar{D})^{\alpha-1}}$ is also non-increasing in α, we obtain that

$$\inf_{1<\gamma<\frac{C_1}{\lambda}} \left\{ \frac{\gamma}{(C_1 D)^{\alpha_1 - 1} (C_1 - \gamma\lambda)} \frac{\gamma^{\frac{\alpha_1-1}{\alpha_1}}}{\log \gamma^{\frac{\alpha_1-1}{\alpha_1}}} \right\}$$
$$\geq \inf_{1<\gamma<\frac{C_1}{\lambda}} \left\{ \frac{\gamma}{(C_1 D)^{\alpha_2 - 1} (C_1 - \gamma\lambda)} \frac{\gamma^{\frac{\alpha_2-1}{\alpha_2}}}{\log \gamma^{\frac{\alpha_2-1}{\alpha_2}}} \right\} .$$

Using the monotonicity in C_1 in the term inside the infimum, it follows that $C_1 \geq C_2$, where C_2 is the implicit solution of Eq. (18) for $\alpha = \alpha_1$. Therefore, $C(\bar{D}, \bar{\varepsilon})$ is non-increasing in α.

Scaling law: *To prove the scaling law, denote by C the implicit solution of the equation*

$$\inf_{1<\gamma<\max\left\{\frac{C}{\lambda}, e^{\frac{2\alpha}{\alpha-1}}\right\}} \left\{ \frac{1}{C^\alpha} \frac{\gamma^{\frac{\alpha-1}{\alpha}}}{\log \gamma^{\frac{\alpha-1}{\alpha}}} \right\} = \bar{\varepsilon} \bar{D}^{\alpha-1} . \tag{20}$$

The LHS here was constructed by relaxing the function inside the infimum in the LHS of Eq. (18) and extending the range of the infimum. This means that the implicit solution C is smaller than the implicit solution $C(\bar{D}, \bar{\varepsilon})$. The function inside the infimum of the LHS of Eq. (20) is convex on the domain of γ and attains its infimum at $\gamma = e^{\frac{\alpha}{\alpha-1}}$. Solving for C and using that $C(\bar{D}, \bar{\varepsilon}) \geq C$ proves the lower bound.

To prove the upper bound, let us fix α, $\bar{D}_0$ and $\bar{\varepsilon}_0$, and denote by $C_0(\bar{D}_0, \varepsilon_0)$ the corresponding implicit solution. Using the monotonicity of the implicit solution in $\bar{\varepsilon}\bar{D}^{\alpha-1}$, as shown above, it follows that

$$C(\bar{D}, \bar{\varepsilon}) \geq C_0(\bar{D}_0, \bar{\varepsilon}_0) ,$$

where $\bar{\varepsilon}\bar{D}^{\alpha-1} \leq \bar{\varepsilon}_0 \bar{D}_0^{\alpha-1}$. Fixing $\gamma_0 = \frac{C_0(\bar{D}_0, \bar{\varepsilon}_0) + \lambda}{2\lambda}$, let C be the solution of the equation

$$\frac{\gamma_0}{C^{\alpha-1} (C - \gamma_0 \lambda)} \frac{\gamma_0^{\frac{\alpha-1}{\alpha}}}{\log \gamma_0^{\frac{\alpha-1}{\alpha}}} = \bar{\varepsilon} \bar{D}^{\alpha-1} . \tag{21}$$

Because the range of γ in the solution of $C(\bar{D}, \bar{\varepsilon})$ includes γ_0, it follows that $C(\bar{D}, \bar{\varepsilon}) \leq C$. On the other hand, the LHS of Eq. (21) satisfies

$$\frac{\gamma_0}{C^{\alpha-1} (C - \gamma_0 \lambda)} \frac{\gamma_0^{\frac{\alpha-1}{\alpha}}}{\log \gamma_0^{\frac{\alpha-1}{\alpha}}} \leq \frac{K_0}{C^\alpha} , \tag{22}$$

where $K_0 = \frac{\gamma_0 C_0(\bar{D}_0, \bar{\varepsilon}_0)}{C_0(\bar{D}_0, \bar{\varepsilon}_0) - \gamma_0 \lambda} \frac{\gamma_0^{\frac{\alpha-1}{\alpha}}}{\log \gamma_0^{\frac{\alpha-1}{\alpha}}}$. Here we used that $C \geq C_0(\bar{D}_0, \bar{\varepsilon}_0)$ (note that we showed before that $C \geq C(\bar{D}, \bar{\varepsilon})$ and $C(\bar{D}, \bar{\varepsilon}) \geq C_0(\bar{D}_0, \bar{\varepsilon}_0)$). Finally, combining Eqs. (21) and (22) we immediately get the scaling law $C = O\left(\left(\frac{1}{\bar{\varepsilon}\bar{D}^{\alpha-1}}\right)^{\frac{1}{\alpha}}\right)$, and since $C(\bar{D}, \bar{\varepsilon}) \leq C$, the proof is complete.

6. Conclusion

Our goal in this paper is to provide new insight into the debate about the potential of dynamic resizing in data centers. Clearly, there are many facets of this issue relating to the engineering, algorithmic, and reliability challenges involved in dynamic resizing which we have ignored in this paper. These are all important issues when trying to *realize the potential* of dynamic resizing. But, the point we have made in this paper is that when *quantifying the potential* of dynamic resizing it is of primary importance to understand the joint impact of workload and SLA characteristics.

To make this point, we have presented a new model that, for the first time, captures the impact of SLA characteristics in addition to both slow time-scale non-stationarities in the workload and fast time-scale burstiness in the workload. This model allows us to provide the first study of dynamic resizing that captures both the stochastic burstiness and diurnal non-stationarities of real workloads. Within this model, we have provided both trace-based numerical case studies and analytical results. Perhaps most tellingly, our results highlight that even when two of SLA, peak-to-mean ratio, and burstiness are fixed, the other one can be chosen to ensure that there either are or are not significant savings possible via dynamic resizing. Figures 12-14 illustrate how dependent the potential of dynamic resizing is on these three parameters. These figures highlight that a precursor to any debate about the value of dynamic resizing must be an understanding of the workload characteristics expected and the SLA desired. Then, one can begin to discuss whether this potential is obtainable.

Future work on this topic includes providing a more detailed study of how other important factors affect the potential of dynamic resizing, e.g., storage issues, reliability issues, and the availability of renewable energy. Note that provisioning capacity to take advantage of renewable energy when it is available is an important benefit of dynamic resizing that we have not considered at all in the current paper.

Acknowledgment

This research is supported by the NSF grant of China (No. 61303058), the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDA06010600), the 973 Program of China (No. 2010CB328105), and NSF grant CNS 0846025 and DoE grant DE-EE0002890.

References

[1] K. Wang, M. Lin, F. Ciucu, A. Wierman, C. Lin, Characterizing the impact of the workload on the value of dynamic resizing in data centers, in: Proc. of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems, 2012.

[2] K. Wang, M. Lin, F. Ciucu, A. Wierman, C. Lin, Characterizing the impact of the workload on the value of dynamic resizing in data centers, in: Proc. of the 32th IEEE International Conference on Computer Communications (Infocom mini-conference), 2012.

[3] D. Meisner, B. T. Gol, T. F. Wenisch, PowerNap: Eliminating server idle power, in: ACM SIGPLAN Notices, Vol. 44, 2009, pp. 205–216.

[4] L. A. Barroso, U. Hölzle, The case for energy- proportional computing, Computer 40 (12) (2007) 33–37.

[5] E. V. Carrera, E. Pinheiro, R. Bianchini, Conserving disk energy in network servers, in: Proc. of the 17th annual International Conference on Supercomputing, ICS '03, ACM, New York, NY, USA, 2003, pp. 86–97.

[6] B. Diniz, D. Guedes, W. Meira, Jr., R. Bianchini, Limiting the power consumption of main memory, in: Proc. of the 34th annual International Symposium on Computer Architecture (ISCA), ACM, New York, NY, USA, 2007, pp. 290–301.

[7] S. Govindan, J. Choi, B. Urgaonkar, A. Sivasubramaniam, A. Baldini, Statistical profiling-based techniques for effective power provisioning in data centers, in: Proc. of the 4th ACM European conference on Computer systems (EuroSys), 2009.

[8] S. Albers, Energy-efficient algorithms, Communications of the ACM 53 (5) (2010) 86–96.

[9] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, F. Zhao, Energy-aware server provisioning and load dispatching for connection-intensive internet services, in: Proc. of the 5th USENIX Symposium on Networked Systems Design & Implementation (NSDI), 2008.

[10] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, R. P. Doyle, Managing energy and server resources in hosting centers, in: Proc. of the eighteenth ACM Symposium on Operating Systems Principles, SOSP'01, ACM, New York, NY, USA, 2001, pp. 103–116.

[11] M. Lin, A. Wierman, L. L. H. Andrew, E. Thereska, Dynamic right-sizing for power-proportional data centers, in: Proc. of the 30th IEEE International Conference on Computer Communications (Infocom), 2011, pp. 1098–1106.

[12] T. Lu, M. Chen, L. L. Andrew, Simple and effective dynamic provisioning for power-proportional data centers, IEEE Transactions on Parallel and Distributed Systems 24 (6) (2013) 1161–1171.

[13] A. Greenberg, J. Hamilton, D. A. Maltz, P. Patel, The cost of a cloud: research problems in data center networks, SIGCOMM Comput. Commun. Rev. 39 (2008) 68–73.

[14] D. Meisner, C. M. Sadler, L. A. Barroso, W.-D. Weber, T. F. Wenisch, Power management of online data-intensive services, in: Proc. of the 38th annual International Symposium on Computer Architecture (ISCA), 2011.

[15] J. Heo, P. Jayachandran, I. Shin, D. Wang, T. Abdelzaher, X. Liu, Optituner: On performance composition and server farm energy minimization application, IEEE Transactions on Parallel and Distributed Systems 22 (11) (2011) 1871–1878.

[16] Q. Zhang, M. F. Zhani, S. Zhang, Q. Zhu, R. Boutaba, J. L. Hellerstein, Dynamic energy-aware capacity provisioning for cloud computing environments, in: Proc. of the 9th International Conference on Autonomic Computing (ICAC), 2012.

[17] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, Live migration of virtual machines, in: Proc. of the 2nd USENIX Symposium on Networked Systems Design & Implementation (NSDI), 2005, pp. 273–286.

[18] E. Thereska, A. Donnelly, D. Narayanan, Sierra: a power-proportional, distributed storage system, Tech. Rep. MSR-TR-2009-153, Microsoft Research (2009).

[19] H. Amur, J. Cipar, V. Gupta, G. R. Ganger, M. A. Kozuch, K. Schwan, Robust and flexible power-proportional storage, in: the 1st ACM Symposium on Cloud Computing (SoCC), 2010.

[20] P. Bodik, M. P. Armbrust, K. Canini, A. Fox, M. Jordan, D. A. Patterson, A case for adaptive datacenters to conserve energy and improve reliability, Tech. Rep. UCB/EECS-2008-127, University of California at Berkeley (2008).

[21] A. Gandhi, M. Harchol-Balter, M. A. Kozuch, The case for sleep states in servers, in: Proc. of the 4th Workshop on Power-Aware Computing and Systems (HotPower), ACM, New York, NY, USA, 2011, pp. 2:1–2:5.

[22] F. Ahmad, T. N. Vijaykumar, Joint optimization of idle and cooling power in data centers while maintaining response time, in: Proc. of the 15th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'10, ACM, New York, NY, USA, 2010, pp. 243–256.

[23] A. Gandhi, V. Gupta, M. Harchol-Balter, M. A. Kozuch, Optimality analysis of energy-performance trade-off for server farm management, Performance Evaluation 67 (11) (2010) 1155–1171.

[24] Y. Chen, D. Gmach, C. Hyser, Z. Wang, C. Bash, C. Hoover, S. Singhal, Integrated management of application performance, power and cooling in data centers, in: Proc. of the 12th IEEE/IFIP Network Operations and Management Symposium (NOMS), 2010.

[25] A. Verma, G. Dasgupta, T. K. Nayak, P. De, R. Kothari, Server workload analysis for power minimization using consolidation, in: Proc. of the 2009 conference on USENIX Annual Technical Conference (USENIX ATC), 2009.

[26] M. Lin, Z. Liu, A. Wierman, L. L. Andrew, Online algorithms for geographical load balancing, in: Proc. of the 3rd International Green Computing Conference (IGCC), 2012.

[27] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, C. Hyser, Renewable and cooling aware workload management for sustainable data centers, in: Proc. of the Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems, 2012.

[28] SPEC power data on SPEC website at http://www.spec.org.

[29] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, G. Jiang, Power and performance management of virtualized computing environments via lookahead control, in: Proc. of the 5th IEEE International Conference on Autonomic Computing (ICAC), 2008.

[30] M. Lin, A. Wierman, L. L. H. Andrew, E. Thereska, Online dynamic capacity provisioning in data centers, in: Proc. of the 49th Allerton Conference on Communication, Control, and Computing, 2011.

[31] C.-S. Chang, Performance Guarantees in Communication Networks, Springer Verlag, 2000.

[32] K. Wang, M. Lin, F. Ciucu, A. Wierman, C. Lin, Characterizing the impact of the workload on the value of dynamic resizing in data centers, Tech. Rep. ISCAS-2014-GD1, available at http://arxiv.org/abs/1207.6295v2 (2014).

[33] F. Ciucu, Network calculus delay bounds in queueing networks with exact solutions, in: Proc. of the 20th International Teletraffic Congress (ITC), 2007, pp. 495–506.

[34] J. Liebeherr, A. Burchard, F. Ciucu, Delay bounds in communication networks with heavy-tailed and self-similar traffic, IEEE Transactions on Information Theory 58 (2) (2012) 1010 –1024.

[35] D. Gmach, J. Rolia, L. Cherkasova, A. Kemper, Workload analysis and demand prediction of enterprise data center applications, in: Proc. of the 10th IEEE International Symposium on Workload Characterization (IISWC), IEEE Computer Society, 2007.