Server farms with setup costs

A. Gandhi, M. Harchol-Balter, I. Adan
ISSN 1389-2355
Server farms with setup costs

Anshul Gandhi† Mor Harchol-Balter† Ivo Adan‡

†anshulg,harchol]@cs.cmu.edu ‡iadan@win.tue.nl

Computer Science Department
Carnegie Mellon University

Department of Mathematics and Computer Science
Eindhoven University of Technology

Abstract

In this paper we consider server farms with a setup cost. This model is common in manufacturing systems and data centers, where there is a cost to turn servers on. Setup costs always take the form of a time delay, and sometimes there is additionally a power penalty, as in the case of data centers. Any server can be either on, off, or in setup mode. While prior work has analyzed single servers with setup costs, no analytical results exist for multi-server systems. In this paper, we derive the first closed-form solutions and approximations for the mean response time and mean power consumption in server farms with setup costs. We also analyze variants of server farms with setup, such as server farm models with staggered boot up of servers, where at most one server can be in setup mode at a time, or server farms with an infinite number of servers. For some variants, we find that the distribution of response time can be decomposed into the sum of response time for a server farm without setup and the setup time. Finally, we apply our analysis to data centers, where both response time and power consumption are key metrics. Here we analyze policy design questions such as whether it pays to turn servers off when they are idle, whether staggered boot up helps, and other questions related to the optimal data center size.

1 Introduction

Motivation

Server farms are ubiquitous in manufacturing systems, call centers and service centers. In manufacturing systems, machines are usually turned off when they have no work to do, in order to save on operating costs. Likewise, in call centers and service centers, employees can be dismissed when there are not enough customers to serve. However, there is usually a setup cost involved in turning on a machine, or in bringing back an employee. This setup cost is typically in the form of a time delay. Thus, an important question in manufacturing systems, call centers and service centers, is whether it pays to turn machines/employees “off”, when there is not enough work to do.
Server farms are also prevalent in data centers. In data centers, servers consume peak power when they are servicing a job, but still consume about 60% [5] of that peak power, when they are idle. Idle servers can be turned off to save power. Again, however, there is a setup cost involved in turning a server back on. This setup cost is in the form of a time delay, and a power penalty, since the server consumes peak power during the entire duration of the setup time. An open question in data centers is whether it pays (from a delay perspective and a power perspective) to turn servers off when they are idle.

**Model**

Abstractly, we can model a server farm with setup costs using an $M/M/k$ queueing system, with a Poisson arrival process with rate $\lambda$, and Exponentially distributed job sizes, denoted by random variable $S \sim Exp(\mu)$. Let $\rho = \frac{\lambda}{\mu}$ denote the system load, where $0 \leq \rho < k$. In this model, a server can be in one of three states: on, off, or in setup. A server is in the on state when it is serving jobs. When the server is on, it consumes power $P_{on}$. If there are no jobs to serve, the server can either remain idle, or be turned off, where there is no time delay to turn a server off. If a server remains idle, it consumes non-zero power $P_{idle}$, which is assumed to be less than $P_{on}$. If the server is turned off, it consumes zero power.

If there are no jobs to serve, the server turns off instantaneously. To turn on an off server, it must first be put in setup mode. While in setup, a server cannot serve jobs. The time it takes for a server in setup mode to turn on is called the setup time, and during that entire time, power $P_{on}$ is consumed. We model the setup time as an exponentially distributed random variable, $I$, with rate $\alpha = \frac{1}{E[I]}$. As in an $M/M/k$, we assume a First Come First Serve central queue, from which servers pick jobs, when they become free. However, setup costs make things more complicated. 

Fig. 1 illustrates our server farm model. From the perspective of a job, if a job arrives and finds that every server is either on or in setup, the job simply waits in the queue. If the job finds at least one server off, the job also waits in the queue, but it randomly picks an off server, and puts it in setup mode. The job need not necessarily wait the full setup duration of its server, since any on server which becomes free will immediately pick the job at the head of the queue,
call it job $j$, to serve, even if job $j$ was setting up a different server, call it server $i$. At that time, if each job remaining in the queue was already setting up a server, or there are no jobs in the queue, then server $i$ is turned off. However, if there is a job, call it job $j_2$, in the queue that was not setting up a server (because job $j_2$ did not find an off server when it arrived), then server $i$ is transferred to job $j_2$ and remains in setup. We call the above model where servers can be on, off, or in setup mode, the $M/M/k/k\alpha$ model (see below). We use $T_{M/M/k/k\alpha}$ (respectively, $P_{M/M/k/k\alpha}$) to denote the response time (respectively, power consumption) in the $M/M/k/k\alpha$ model, and likewise for other models.

In this paper, we consider the following models/policies of server farms:

1. $M/M/k/k\alpha$: In this model, servers are turned off when not in use. However, there is a setup cost (in terms of delay and power) for turning on an off server. We use the notation $k\alpha$ to denote that any number of servers, up to $k$, can be in the setup mode simultaneously.

2. $M/M/\infty/\infty\alpha$: This model can be viewed as an $M/M/k/k\alpha$, in the limit as $k \to \infty$. The $M/M/\infty/\infty\alpha$ model is useful for modeling large data centers, where the number of servers is usually in the thousands [9, 14].

3. $M/M/k/1\alpha$: This model is known as the “staggered boot up” model in data centers, or “staggered spin up” in disk farms [10, 16]. Here, at most 1 server can be in setup at any point of time. The staggered boot up model is believed to avoid excessive power consumption.

4. $M/M/k$: Here we assume that servers are never turned off. Servers remain in the idle mode when there are no jobs to serve.

Prior work

Prior work on server farms with setup costs has focussed largely on single servers. There is very little work on multi-server systems with setup costs. In particular, no closed-form solutions exist for the $M/M/k/k\alpha$ and the $M/M/\infty/\infty\alpha$. For the $M/M/k/1\alpha$, Gandhi and Harchol-Balter have obtained closed-form solutions for the mean response time [11], but no results exist for the distribution of response time.

Results

For the $M/M/k/1\alpha$, we provide the first analysis of the distribution of response time. In particular, we prove that the distribution of response time can be decomposed into the sum of response time for an $M/M/k$ and the setup time (see Section 4). For the $M/M/\infty/\infty\alpha$, we provide closed-form solutions for the limiting probabilities, and also observe an interesting decomposition property on the number of jobs in the system. These can then be used to derive the mean response time and mean power consumption (see Section 5). For the $M/M/k/k\alpha$, we come up with closed-form approximations for the mean response time which work well under all ranges of load and setup times, except the regime where both the load and the setup time are high. Understanding the $M/M/k/k\alpha$ in the regime where both
the load and the setup time are high is less important, since in this regime, as we will show, it pays to leave servers on
(M/M/k policy). Both of our approximations for the M/M/k/\(k\alpha\) are based on the truncation of systems where we have an infinite number of servers (see Section 6). Finally, we analyze the limiting behavior of server farms with setup costs as the number of jobs in the system becomes very high. One would think that all \(k\) servers should be on in this case. Surprisingly, our derivations show that the limit of the expected number of on servers converges to a quantity that can be much less than \(k\). This type of limiting analysis leads to yet another approximation for the mean response time for an M/M/k/\(k\alpha\) (see Section 7).

**Impact/Application**

Using our analysis of server farms with setup costs, we answer many interesting policy design questions that arise in data centers. Each question is answered both with respect to mean response time and mean power consumption. These include, for example, “Under what conditions is it beneficial to turn servers off, to save power? (M/M/k vs. M/M/k/\(k\alpha\))”; “Does it pay to limit the number of servers that can be in setup? (M/M/k/\(k\alpha\) vs. M/M/k/1\(\alpha\))”; “How are results affected by the number of servers, load, and setup time?” (see Section 8).

## 2 Prior work

Prior work on server farms with setup costs has focussed largely on single servers. There is very little work on multi-server systems with setup costs.

**Single server with setup costs:** For a single server, Welch [18] considered the M/G/1 queue with general setup times, and showed that the mean response time can be decomposed into the sum of mean response time for the M/G/1 and the mean of the residual setup time. In [17], Takagi considers a multi-class M/G/1 queue with setup times and a variety of queueing disciplines including FCFS and LCFS, and derives the Laplace-Stieltjes transforms of the waiting times for each class. Other related work on a single server with setup costs includes [6, 7, 8, 12].

**Server farms with setup costs:** For the case of multiple servers with setup times, Artalejo et al. [4] consider an M/M/k/1\(\alpha\) queueing system with exponential service times. They solve the steady state equations for the associated Markov chain, using a combination of difference equations and Matrix analytic methods. The resulting solutions are not closed-form, but can be solved numerically. In [2], the authors consider an inventory control problem, that involves analyzing a Markov chain similar to the M/M/k/1\(\alpha\). Again, the authors provide recursive formulations for various performance measures, which are then numerically solved for various examples. Finally, Gandhi and Harchol-Balter recently analyze the M/M/k/1\(\alpha\) queueing system [11] and derive closed-form results for the mean response time and mean power consumption. However, [11] does not analyze the distribution of the response times for an M/M/k/1\(\alpha\), as we do in this paper. Importantly, to the best of our knowledge, there is no prior work on analyzing the M/M/\(\infty\)/\(\infty\)α
or the $M/M/k/k\alpha$. We provide the first analysis of these systems, deriving closed-form solutions and approximations for their mean response time and mean power consumption.

3 $M/M/k$

In the $M/M/k$ model (see Section 1), servers become idle when they have no jobs to serve. Thus, the mean response time, $E[T_{M/M/k}]$, and the mean power consumption, $E[P_{M/M/k}]$, are given by:

$$E[T_{M/M/k}] = \frac{\pi_0 \cdot \rho^k}{k! \cdot (1 - \frac{\rho}{k})^2 \cdot k\mu} + \frac{1}{\mu}, \quad \text{where} \quad \pi_0 = \left[ \sum_{i=0}^{k-1} \frac{\rho^i}{i!} + \frac{\rho^k}{k! \cdot (1 - \frac{\rho}{k})} \right]^{-1}$$

(1)

$$E[P_{M/M/k}] = \rho \cdot P_{on} + (k - \rho) \cdot P_{idle}$$

(2)

In Eq. (2), observe that $\rho$ is the expected number of on servers, and $(k - \rho)$ is the expected number of idle servers.

4 $M/M/k/1\alpha$

In data centers it is common to turn idle servers off to save power. When a server is turned on again, it incurs a setup cost, both in terms of a time delay and a power penalty. If there is a sudden burst of arrivals into the system, then many servers might be turned on simultaneously, resulting in a huge power draw, since servers in setup consume peak power. To avoid excessive power draw, data center operators sometime limit the number of servers that can be in setup at any point of time. This is referred to as “staggered boot up”. The idea behind staggered boot up is also employed in disk farms, where at most one disk is allowed to spin up at any point of time, to avoid excessive power draw. This is referred to as “staggered spin up” [10, 16]. While staggered boot up may help reduce power, its effect on the distribution of response time is not obvious.

We can represent the staggered boot up policy using an $M/M/k/1\alpha$ Markov chain, as shown in Fig. 2, with states $(i, j)$, where $i$ represents the number of servers on, and $j$ represents the number of jobs in the system. Note that when $j > i$ and $i < k$, we have exactly one of the $(k - i)$ servers in setup. However, when $i = k$, there are no servers in setup. In [11], Gandhi and Harchol-Balter obtained the limiting probabilities, $\pi_{i,j}$, of the $M/M/k/1\alpha$ Markov chain using the method of difference equations (see [3] for more information on difference equations), provided for reference in Lemma 1 below.
Lemma 1 [11] The limiting probabilities, $\pi_{i,j}$, for the $M/M/k/1\alpha$ are given by:

$$
\pi_{i,j} = \pi_{0,0} \cdot \rho^i \left( \frac{\lambda}{\lambda + \alpha} \right)^{j-i} \quad \text{if } 0 \leq i < k \text{ and } j \geq i
$$

(3)

$$
\pi_{k,j} = \frac{\pi_{0,0} \cdot \rho^k}{k!} \left( \frac{\lambda}{\lambda + \alpha} \right)^{j-k} + \frac{\lambda + \alpha}{k\mu - (\lambda + \alpha)} \left[ \left( \frac{\lambda}{\lambda + \alpha} \right)^{j-k} - \left( \frac{\rho}{k} \right)^{j-k} \right] \quad \text{if } j \geq k
$$

(4)

where $\pi_{0,0} = \left(1 - \frac{\lambda}{\lambda + \alpha}\right) \cdot \left\{ \sum_{i=0}^{k} \frac{\rho^i}{i!} + \frac{\rho^k}{k!} \frac{\lambda}{k\mu - \lambda} \right\}^{-1}$

(5)

Below, we show that the limiting probabilities from Lemma 1 can be used to derive the distribution of the response time for an $M/M/k/1\alpha$, and simultaneously we prove a beautiful decomposition result: The response time for an $M/M/k/1\alpha$ can be decomposed into the sum of the response time for an $M/M/k$ system, and the exponential setup time, $I$.

**Theorem 1** For an $M/M/k/1\alpha$, with Exponentially distributed setup time $I \sim \text{Exp}(\alpha)$, we have:

$$
T_{M/M/k/1\alpha} \overset{d}{=} I + T_{M/M/k}
$$

(6)

where $T_{M/M/k}$ is the random variable representing the response time for an $M/M/k$ system, which is independent of the setup time, $I$.

---

**Figure 2:** Markov chain for the $M/M/k/1\alpha$. 

---


Proof: In order to derive the distribution of response times for an $M/M/k/1\alpha$, we’ll first derive the $z$-transform of the number of jobs in queue, $\hat{N}_Q(z)$. Then, we’ll use this to obtain $\tilde{T}_Q(s)$, the Laplace-Stieltjes transform for the time in queue of an $M/M/k/1\alpha$.

Using Lemma 1, the limiting probabilities for the number of jobs in queue for an $M/M/k/1\alpha$ can be expressed as:

$$Pr[N_Q = i] = \pi_{0,0} + \pi_{1,1+i} + \pi_{2,2+i} + \ldots + \pi_{k,k+i}$$

$$= \pi_{0,0} \left( \sum_{j=0}^{k} \frac{\rho_j^i}{j!} \right) \beta^i + \frac{\pi_{0,0} \rho^k (\lambda + \alpha)}{k!(k\mu - \lambda - \alpha)} \left( \beta^i - \left( \frac{\rho}{k} \right)^i \right) \quad \text{where} \quad \beta = \frac{\lambda}{\lambda + \alpha}$$

$$\hat{N}_Q(z) = \sum_{i=0}^{\infty} Pr[N_Q = i] \cdot z^i = \sum_{i=0}^{\infty} \pi_{0,0} \left( \sum_{j=0}^{k} \frac{\rho_j^i}{j!} \right) \beta^i z^i + \sum_{i=0}^{\infty} \frac{\pi_{0,0} \rho^k (\lambda + \alpha)}{k!(k\mu - \lambda - \alpha)} \left( \beta^i - \left( \frac{\rho}{k} \right)^i \right) z^i$$

$$= \frac{\pi_{0,0} \left( \sum_{j=0}^{k} \frac{\rho_j^i}{j!} \right)}{1 - \beta z} + \frac{\pi_{0,0} \rho^k (\lambda + \alpha) \lambda z}{k!(k\mu - \lambda z)(\lambda + \alpha - \lambda z)}$$

At this point, we have derived the $z$-transform of the number of jobs in the queue, which we will now convert to the Laplace-Stieljes transform of the waiting time in queue. By PASTA, an arrival sees the steady state number in the queue, which is the same (in distribution) as the number of jobs seen by a departure in the queue. However, the jobs left behind by a departure are exactly the ones that arrived during the job’s time spent in the queue. Thus we have $\hat{N}_Q(z) = \tilde{T}_Q(\lambda(1 - z))$, or equivalently, $\tilde{T}_Q(s) = \hat{N}_Q(1 - \frac{s}{\lambda})$. This gives us:

$$\tilde{T}_Q(s) = \frac{\pi_{0,0} (\lambda + \alpha)}{s + \alpha} \left\{ \sum_{j=0}^{k} \frac{\rho_j^i}{j!} + \frac{\rho^k (\lambda - s)}{k!(k\mu - \lambda + s)} \right\} = \frac{\pi_{0,0} (\lambda + \alpha)}{s + \alpha} \left\{ \frac{\rho^k}{k!} \left( \frac{\lambda - s}{k\mu - \lambda + s} - \frac{\lambda}{k\mu - \lambda} \right) + \frac{\alpha}{(\lambda + \alpha) \pi_{0,0}} \right\}$$

After a few steps of algebra, the above equation simplifies to:

$$\tilde{T}_Q(s) = \left( \frac{\alpha}{s + \alpha} \right) \left\{ (1 - P_Q) + P_Q \frac{k\mu - \lambda}{k\mu - \lambda + s} \right\} = \hat{I} \cdot \hat{T}_{Q_{M/k}}(s)$$

(7)

where $P_Q$ is the probability of queueing in an $M/M/k$ system. Thus:

$$T_{Q_{M/k}} \xrightarrow{d} I + T_{Q_{M/k}} \implies T_{M/k} \xrightarrow{d} I + T_{M/k}$$
Figure 3: Markov chain for the $M/M/\infty/\infty\alpha$.

Lemma 2 [11] The mean power consumption in the $M/M/k/1\alpha$ is given by:

$$
\mathbb{E}[P_{M/M/k/1\alpha}] = P_{\text{on}} \left( \rho + \frac{\lambda}{\lambda + \alpha} - \frac{\pi_{0,0} \rho^k \lambda}{k! \cdot \alpha \cdot (1 - \rho^k)} \right)
$$

where $\pi_{0,0}$ is given by Eq. (5) (8)

5 $M/M/\infty/\infty\alpha$

Many data centers today, including those of Google, Microsoft, Yahoo and Amazon, consist of tens of thousands of servers [9, 14]. In such settings, we can model a server farm with setup costs as an $M/M/\infty/\infty\alpha$ system, as shown in Fig. 3. For this model, we make an educated guess for the limiting probabilities.

Theorem 2 For the $M/M/\infty/\infty\alpha$ Markov chain, as shown in Fig. 3, the limiting probabilities are given by:

$$
\pi_{i,j} = \frac{\pi_{0,0} \cdot \rho^j \prod_{l=1}^{j-i} \frac{\lambda}{\lambda + l\alpha}}{i!}, \quad i \geq 0, j \geq i, \quad \text{and} \quad \pi_{0,0} = e^{-\rho} \left( \sum_{j=0}^{\infty} \prod_{l=1}^{j} \frac{\lambda}{\lambda + l\alpha} \right)^{-1} = \frac{e^{-\rho}}{M(1,1 + \frac{\lambda}{\alpha})},
$$

where $M(a,b,z) = \sum_{n=0}^{\infty} \frac{a_n}{(b)_n} \frac{z^n}{n!}$ is Kummer’s function [1], and $(a)_n = a(a+1) \cdots (a+n-1), \ (a)_0 = 1$.

Proof: The correctness of Eq. (9) can be verified by direct substitution into the $M/M/\infty/\infty\alpha$ Markov chain. ■

The product-form solution in Eq. (9) implies that the number of jobs in service is independent from the number in the
queue, where the number in service is Poisson distributed with mean \( \rho \) and the number in the queue is distributed as:

\[
\Pr[N_{Q, M/M/\infty/\infty \alpha} = j] = \frac{1}{M(1, 1 + \frac{\lambda}{\alpha}, \frac{\lambda}{\alpha})} \prod_{l=1}^{j} \frac{\lambda}{\lambda + l\alpha}, \quad \text{with mean } E[N_{Q, M/M/\infty/\infty \alpha}] = \frac{1}{1 + \frac{\lambda}{\alpha}} \frac{M(2, 2 + \frac{\lambda}{\alpha}, \frac{\lambda}{\alpha})}{M(1, 1 + \frac{\lambda}{\alpha}, \frac{\lambda}{\alpha})}.
\]

Thus, by Little’s law:

\[
E[T_{M/M/\infty/\infty \alpha}] = \frac{1}{\mu} + \frac{1}{\lambda} E[N_{Q, M/M/\infty/\infty \alpha}], \quad E[P_{M/M/\infty/\infty \alpha}] = P_{on} (\rho + E[N_{Q, M/M/\infty/\infty \alpha}])
\] (10)

6 \( M/M/k/k\alpha \): Approximations inspired by the \( M/M/\infty \)

Under the \( M/M/k/k\alpha \) model, we assume a fixed finite number of servers \( k \), each of which can be either on, off, or in setup. Fig. 4 shows the \( M/M/k/k\alpha \) Markov chain, with states \((i, j)\), where \( i \) represents the number of servers on, and \( j \) represents the number of jobs in the system. Note that when \( j > i \) and \( i < k \), we have exactly \( \min\{j - i, k - i\} \) servers in setup. Since the Markov chain for the \( M/M/k/k\alpha \) (shown in Fig. 4) looks similar to the Markov chain for the \( M/M/k/1\alpha \) (shown in Fig. 2), one would expect that the difference equations method used to solve the \( M/M/k/1\alpha \) should work for the \( M/M/k/k\alpha \) too. While we can solve the difference equations for the \( M/M/k/k\alpha \), the resulting \( \pi_{i, j} \)'s do not lead to simple closed-form expressions for the mean response time or the mean power consumption. A more detailed explanation of why the \( M/M/k/k\alpha \) is not tractable in closed-form via difference equations is given in [11]. In this section, we will try to approximate \( E[T_{M/M/k/k\alpha}] \) and \( E[P_{M/M/k/k\alpha}] \) by simple closed-form expressions. While Matrix-analytic methods could, in theory, be used to solve the chain in Fig. 4, having closed-form approximations is preferable for two reasons: (i) We often care about large \( k \), in which case the Matrix-analytic methods are very cumbersome and time consuming, and (ii) Our closed-form expressions give us insights about the \( M/M/k/k\alpha \) system (which would not have been obtainable via Matrix analytic methods), that we exploit in Section 7 to derive further properties of the \( M/M/k/k\alpha \).

A major goal in analyzing the \( M/M/k/k\alpha \) is to define regimes (in terms of load and setup times) for which it pays to turn servers off when they are idle. Consider the regime where both the load is high and the setup time is high. In this regime, we clearly do not want to turn servers off when they are idle, since it takes a long time to get a server back on, and new jobs are likely to arrive very soon.\(^1\) We are most interested in understanding the behavior of the \( M/M/k/k\alpha \) in regimes where it is useful, namely, either the load is not too high, or the setup cost is not too high.

In Section 6.1, we first approximate the \( M/M/k/k\alpha \) system using a \( M/M/\infty/\infty \alpha \) system, where we truncate the

\(^1\) For a single server, we can easily prove that if both the load is high and the setup time is high, then turning the server off when idle only increases the mean power consumption (and trivially, increases the mean response time). For the mean power consumption, we have \( E[P_{M/M/1}] = \rho P_{on} + (1 - \rho) P_{idle} \) and \( E[P_{M/M/1/1\alpha}] = \rho P_{on} + (1 - \rho) \frac{\lambda}{\alpha} + \frac{\lambda}{\alpha} P_{on}. \) Thus \( E[P_{M/M/1/1\alpha}] \geq E[P_{M/M/1}] \iff \frac{\lambda}{\alpha} \leq \frac{P_{on} - P_{idle}}{P_{idle}} \), which is true when the setup time is high and the load (or \( \lambda \)) is high.
number of jobs to be less than \( k \). We find that this approximation works surprisingly well for low loads, but does not work well when the load is high. Then, in Section 6.2, we approximate the \( M/M/k/k\alpha \) system using a truncated version of the \( M/M/\infty/\infty\alpha \) system, where we have an infinite number of servers, but at most \( k \) servers can be in setup simultaneously. We find that this approximation works very well in any regime where either the load is not too high or the setup cost is not too high. Thus, this latter approximation gives us a good estimate of the \( M/M/k/k\alpha \) in all regimes where it can be useful. Our emphasis in this section is on evaluating the accuracy of our approximations. We defer discussing the intuition behind the results to Section 8, where we focus on applications of our research.

### 6.1 Truncated \( M/M/\infty/\infty\alpha \)

Consider the Markov chains for the \( M/M/k/k\alpha \) (shown in Fig. 4) and the \( M/M/\infty/\infty\alpha \) (shown in Fig. 3). The two Markov chains are exactly alike for \( j < k \). Thus, we can approximate the \( \pi_{i,j} \)'s for the \( M/M/k/k\alpha \) using the \( \pi_{i,j} \)'s for the \( M/M/\infty/\infty\alpha \) from Eq. (9), for \( j < k \). Further, when the load in the system is low, we expect the number of jobs in the system to be less than \( k \), with high probability. Thus, approximating the \( M/M/k/k\alpha \) using the \( M/M/\infty/\infty\alpha \), truncated to \( j < k \) should yield a good approximation for mean response time and mean power consumption. Under this assumption, we have the following limiting probabilities for the \( M/M/k/k\alpha \):

\[
\pi_{i,j} = \frac{\pi_{0,0}}{i!} \rho_i^{j-i} \prod_{l=1}^{j-i} \frac{\lambda}{\lambda + l\alpha}, \quad \text{for } i \geq 0, \ i \leq j < k,
\]

where

\[
\pi_{0,0} = \left( \sum_{i=0}^{k-1} \sum_{j=i}^{k-1} \frac{\rho_i^{j-i}}{i!} \prod_{l=1}^{j-i} \frac{\lambda}{\lambda + l\alpha} \right)^{-1}
\]
Using the above limiting probabilities, we can compute the approximations:

\[
E[T_{M/M/k/k\alpha}] \approx \frac{1}{N} \sum_{i=0}^{k-1} \sum_{j=i}^{k-1} j \cdot \pi_{i,j} \quad \text{and} \quad E[P_{M/M/k/k\alpha}] \approx P_{on} \sum_{i=0}^{k-1} \sum_{j=i}^{k-1} j \cdot \pi_{i,j}.
\]

Fig. 5 shows our results for \(E[T_{M/M/k/k\alpha}]\) and \(E[P_{M/M/k/k\alpha}]\) based on the approximation in Eq. (11). We obtain the exact mean response time and mean power consumption of the \(M/M/k/k\alpha\) system (dashed line in Fig. 5) using Matrix Analytic methods. We set \(P_{on} = 240\) W, which was obtained via experiments on an Intel Xeon E5320 server, running the CPU-bound LINPACK [13] workload. Our approximation seems to work very well, but only for \(\rho < \frac{k}{2}\). This is understandable because, when \(\rho < \frac{k}{2}\), the number of jobs in the system is less than \(k\) with high probability, meaning that \(\pi_{i,j}\), with \(j \geq k\) are very small.

### 6.2 Truncated \(M/M/\infty/k\alpha\)

Our previous approximation could not model the case where the number of jobs is high, and hence performed poorly for high loads. To address high loads, we now consider a different model, the \(M/M/\infty/k\alpha\), which is again inspired by the \(M/M/\infty\) system. For the \(M/M/\infty/k\alpha\) system, we have infinitely many servers, but at most \(k\) can be in setup at any time. After we derive the limiting probabilities for this model, we will then truncate the \(M/M/\infty/k\alpha\) model to having no more than \(k\) on servers. We will refer to this approximation as the truncated \(M/M/\infty/k\alpha\).

For the \(M/M/\infty/k\alpha\) model, we now derive the limiting probabilities:

**Theorem 3** For the \(M/M/\infty/k\alpha\), the limiting probabilities for \(i \geq 0, j \geq i\) are given by:

\[
\pi_{i,j} = \frac{\pi_{0,0} \cdot \rho^i}{i!} \prod_{l=1}^{j-i} \frac{\lambda}{\lambda + \alpha(l)}, \quad \text{where} \quad \alpha(l) = \min\{ka, l\alpha\}, \quad \text{and} \quad \pi_{0,0} = \sum_{i=0}^{k} \sum_{j \geq i} \frac{\rho^i}{i!} \prod_{l=1}^{j-i} \frac{\lambda}{\lambda + \alpha(l)}
\]  

(12)

Observe that \(\alpha(l)\) is a constant for \(l \geq k\).

**Proof:** The correctness of Eq. (12) can be verified by direct substitution into the Markov chain.

For the \(M/M/k/k\alpha\) system, we now truncate our \(M/M/\infty/k\alpha\) Markov chain to \((k+1)\) rows. That is, we only consider states \((i,j)\) with \(i \leq k\). Note that the truncated \(M/M/\infty/k\alpha\) is still not the same as the \(M/M/k/k\alpha\), because, for example, it allows \(k\) servers to be in setup when there are \((k-1)\) servers on, whereas the \(M/M/k/k\alpha\) system allows at most one server to be in setup when \((k-1)\) servers are on. We now derive the limiting probabilities for the truncated \(M/M/\infty/k\alpha\) model.
Theorem 4 For the truncated $M/M/\infty/k\alpha$, the limiting probabilities are given by:

$$\pi_{i,j} = \begin{cases} 
\frac{\pi_{0,0} \rho^i}{i!} \prod_{l=1}^{i-k} \frac{\lambda}{\lambda + \alpha(l)} & \text{if } 0 \leq i < k \text{ and } j \geq i \\
\left(\frac{\rho}{k}\right)^{j-k+1} \pi_{k-1,k-1} + \sum_{r=k+1}^{j-1} \sum_{l=j+1-r}^{j-k} \left(\frac{\rho}{k}\right)^l \frac{(r-k+1)\alpha}{\lambda} + \sum_{l=1}^{j-k} \left(\frac{\rho}{k}\right)^l \pi_{k-1,j-1} & \text{if } i = k \text{ and } k \leq j \leq 2k \\
\pi_{k,2k} \left(\frac{\rho}{k}\right)^{j-2k} + \pi_{k-1,2k-1} \left(\frac{\lambda}{\lambda + k\alpha}\right)^{j-2k} - \left(\frac{\rho}{k}\right)^{j-2k} & \text{if } i = k \text{ and } j > 2k 
\end{cases} \quad (13)$$

Proof: The limiting probabilities, $\pi_{i,j}$, for $0 \leq i < k$ and $j \geq i$, and also for the case of $i = k$, $j = k$, are identical to the limiting probabilities of the $M/M/\infty/k\alpha$ model (up to a normalization constant) and are therefore obtained from Eq. (12).

The limiting probabilities, $\pi_{i,j}$, for $i = k$ and $k < j \leq 2k$, follow immediately from the balance principle applied to the set $\{(k,j), (k,j+1), \ldots\}$, yielding:

$$\pi_{k,j} \mu = \pi_{k,j-1} \lambda + \sum_{l=j}^{\infty} \pi_{k-1,l} \alpha(l-k+1) \quad \text{for } j = k+1, k+2, \ldots, 2k$$

$$\Rightarrow \pi_{k,j} = \left(\frac{\rho}{k}\right)^{j-k+1} \pi_{k-1,k-1} + \sum_{r=k+1}^{j-1} \sum_{l=j+1-r}^{j-k} \left(\frac{\rho}{k}\right)^l \pi_{k-1,r} \frac{(r-k+1)\alpha}{\lambda} + \sum_{l=1}^{j-k} \left(\frac{\rho}{k}\right)^l \pi_{k-1,j-1} \quad \text{for } k < j \leq 2k$$

For the case $i = k$ and $j > 2k$, the balance equations for states $(k,j)$ form a system of second order difference equations with constant coefficients. The general solution for this system is given by:

$$\pi_{k,j} = \pi_{k,2k} \left(\frac{\rho}{k}\right)^{j-2k} + C \left[\left(\frac{\lambda}{\lambda + k\alpha}\right)^{j-2k} - \left(\frac{\rho}{k}\right)^{j-2k}\right] \quad (14)$$

where $\left(\frac{\rho}{k}\right)^{j-2k}$ is the (convergent) solution to the homogeneous equations, and $C \left(\frac{\lambda}{\lambda + k\alpha}\right)^{j-2k}$ is a particular solution to the inhomogeneous equations. We can find the value of $C$ by writing down the balance equation for the state $(k,j)$, where $j > 2k$. The terms involving $\pi_{k,2k}$ vanish from the balance equation, since $\left(\frac{\rho}{k}\right)^{j-2k}$ is the solution of the homogenous equation. This gives us:

$$C = \frac{\pi_{0,0} \lambda \rho^{k-1} \prod_{l=1}^{k} \frac{\lambda}{\lambda + l\alpha}}{(k(\mu - \alpha) - \lambda) \cdot (k - 1)!} = \frac{\pi_{k-1,2k-1} \lambda}{k\mu - (k\alpha + \lambda)} \quad (15)$$

Observe that if we derive the $\pi_{i,j}$ for the special case of a truncated $M/M/\infty/1\alpha$, then we get back the $\pi_{i,j}$ for the $M/M/k/1\alpha$, as in Lemma 1. Finally, $\pi_{0,0}$ can be derived using $\sum_{i=0}^{k} \sum_{j \geq i} \pi_{i,j} = 1$. \hfill \blacksquare
We now approximate the mean response time and the mean power consumption for the $M/M/k/k\alpha$ system, using the limiting probabilities of the truncated $M/M/\infty/k\alpha$ model:

\[
E[T_{M/M/k/k\alpha}] \approx \frac{1}{\lambda} \sum_{i=0}^{k} \sum_{j=i}^{\infty} j \cdot \pi_{i,j} \quad E[P_{M/M/k/k\alpha}] \approx P_{on} \cdot \left[ \sum_{i=0}^{k-1} \sum_{j=i}^{\infty} (\min\{j, i+k\}) \cdot \pi_{i,j} + k \sum_{j=k}^{\infty} \pi_{k,j} \right]
\]

Fig. 6 shows our results for $E[T_{M/M/k/k\alpha}]$ and $E[P_{M/M/k/k\alpha}]$ based on the truncated $M/M/\infty/k\alpha$ approximation. Again, we obtain the mean response time of the $M/M/k/k\alpha$ system using Matrix Analytic methods. We see that our approximation works very well under all cases, except in the regime where both the setup time is high and the load is high. In the regime of high load and high setup time, the truncated $M/M/\infty/k\alpha$ over estimates the number of servers in setup. For example, when there are $(k-1)$ servers on, there should be at most 1 server in setup. However, the truncated $M/M/\infty/k\alpha$ allows up to $k$ servers to be in setup. Thus, the truncated $M/M/\infty/k\alpha$ ends up with a lower mean response time than the $M/M/k/k\alpha$. Using a similar argument, we expect the mean power consumption
of the truncated $M/M/\infty/k\alpha$ to be higher than the mean power consumption of the $M/M/k/k\alpha$, for the case when the load is high and the setup time is high.

7 $M/M/k/k\alpha$: Approximations inspired by the limiting behavior as number of jobs approaches infinity

Thus far, we have approximated the $M/M/k/k\alpha$ model by using the truncated $M/M/\infty/\infty\alpha$ model and the truncated $M/M/\infty/k\alpha$ model, both of which have a 2-dimensional Markov chain. If we can approximate the $M/M/k/k\alpha$ model by using a simple 1-dimensional random walk, then we might get very simple closed-form expressions for the mean response time and the mean power consumption. To do this, we’ll need a definition:

Definition 1 For the $M/M/k/k\alpha$, $ON(n)$ denotes the number of on servers, given that there are $n$ jobs in the system.
In terms of \( ON(n) \), the number of jobs in the \( M/M/k/k\alpha \) is exactly represented by the random walk in Fig. 7. The remainder of this section is devoted to determining \( ON(n) \).

For \( n < k \), we can use the limiting probabilities from the \( M/M/\infty/\infty\alpha \) model to approximate \( ON(n) \), since the Markov chains for both the \( M/M/k/k\alpha \) and the \( M/M/\infty/\infty\alpha \) are similar for \( n < k \), where \( n \) denotes the number of jobs in the system. Thus, we have:

\[
ON(n) = \sum_{i=0}^{n} i \cdot \pi_{i,n}, \text{ for } 1 \leq n < k, \text{ where } \pi_{i,n} \text{ is given by Eq. (9).}
\]  

Now, consider \( ON(n)\rceil_{n \to \infty} \). One would expect that all \( k \) servers should be on when there are infinitely many jobs in the system. Surprisingly, we find that this need not be true.

**Theorem 5** For the \( M/M/k/k\alpha \), we have:

\[
ON(n)\rceil_{n \to \infty} = \min \left\{ k, \frac{\lambda + k\alpha}{\mu} \right\}
\]

where \( ON(n) \) denotes the expected number of servers on, given that there are \( n \) jobs in the system.

**Proof:** Consider the \( M/M/k/k\alpha \) Markov chain for states \((i, j)\), where \( j \to \infty \), as shown in Fig. 8. Using spectral expansion [15], we have that \( \pi_{i,j} = v_i w_j \), where \( w \) is the largest eigenvalue of the transition matrix, and \( v = (v_0, \ldots, v_k) \) is the corresponding eigenvector. The eigenvalue \( w_i \), for \( i = 0, 1, \ldots, k \), of the transition matrix is the unique root on the interval \((0, 1)\) of the quadratic equation \( w(\lambda + i\mu + (k - i)\alpha) = \lambda + i\mu w^2 \). Using some algebra, we can show that if \( \alpha \geq \frac{k\mu - \lambda}{k} \), then \( w_k \) is the largest eigenvalue, with corresponding eigenvector \( v = (0, \ldots, 0, 1) \). In this case, \( ON(n)\rceil_{n \to \infty} = k \). On the other hand, if \( \alpha < \frac{k\mu - \lambda}{k} \), then \( w_0 = \frac{\lambda}{\lambda + \alpha} \) is the largest eigenvalue. In this case, we can solve for \( v \) by substituting \( \pi_{i,j} = v_i w_0^j \) into the balance equation for state \((i, j)\) shown in Fig. 8:

\[
v_i = v_{i-1} \frac{(\lambda + k\alpha) \cdot (k - i + 1)}{i(k\mu - k\alpha - \lambda)} = v_0 \left( \frac{k}{i} \right) \left( \frac{\lambda + k\alpha}{k\mu - k\alpha - \lambda} \right)^i \text{ for } i > 0
\]  

15
Thus, \( ON(n)|_{n \to \infty} = \lim_{n \to \infty} (\sum_{i=0}^{k} \pi_{i,n})^{-1} \cdot \sum_{i=0}^{k} i \cdot \pi_{i,n} \)

\[
= \left[ v_0 \left( \frac{k \mu}{k \mu - k \alpha - \lambda} \right)^k \right]^{-1} \cdot v_0 k (\lambda + k \mu) \cdot \left( \frac{k \mu}{k \mu - k \alpha - \lambda} \right)^{k-1} = \frac{\lambda + k \alpha \mu}{\mu}
\]

We now combine the above two cases to conclude that \( ON(n)|_{n \to \infty} = \min\left\{ k, \frac{\lambda + k \alpha \mu}{\mu} \right\} \).

Now that we have \( ON(n) \) for \( 1 \leq n < k \) and \( n \to \infty \), we can try and approximate \( ON(n) \) for \( n \geq k \), using a straight line fit, with points \( ON(k-2) \) and \( ON(k-1) \), and enforcing that \( ON(n) \leq \min\left\{ k, \frac{\lambda + k \alpha \mu}{\mu} \right\} \). Given the above approximation for \( ON(n) \), we can immediately solve the random walk in Fig. 7 for \( E[T_{M/M/k/k\alpha}] \) and \( E[P_{M/M/k/k\alpha}] \).

Fig. 9 shows our results for \( E[T_{M/M/k/k\alpha}] \) and \( E[P_{M/M/k/k\alpha}] \) based on the random walk in Fig. 7. We see that our approximation for \( ON(n) \) works very well under all cases, except in the regime where both the setup time is high and the load is high. We also considered an exponential curve fit for \( ON(n) \) in the region \( n \geq k \). We found that such a curve fit performed slightly better than our straight line fit.

### 8 Application

In data centers today, both response time and power consumption are important performance metrics. However there is a tradeoff between leaving servers idle and turning them off. Leaving servers idle when they have no work to do results in excessive power consumption, since idle servers consume as much as 60% of peak power [5]. On the other hand, turning servers off when they have no work to do incurs a setup cost (in terms of both a time delay and peak power consumption during that time). Analytically, it is not known how to deal with the tradeoff between leaving servers idle and turning them off. We provide the first modeling and analysis of server farm models with setup costs, allowing us to answer many important policy design questions in data centers.
We start by considering three specific data center policies:

1. **ON/IDLE**: This policy is essentially an $M/M/k$ queueing system, where servers can either be on (consuming power $P_{on}$), or idle (consuming power $P_{idle}$). This policy was analyzed in Section 3.

2. **ON/OFF**: This is the $M/M/k/k\alpha$ system, analyzed in Sections 6 and 7, where servers are turned off when not in use and later incur a setup cost (time delay and power penalty).

3. **ON/OFF/STAG**: This is the staggered boot up policy, represented by the $M/M/k/1\alpha$ system, analyzed in Section 4, where at most one server can be in setup at a time.

In evaluating the performance of the above policies, we use the approximations and closed-form results derived throughout this paper, except for the case of high setup time and high load, where we resort to Matrix analytic methods. Unless otherwise stated, assume that we are using the approximation in Section 7 for the $M/M/k/k\alpha$. In our com-
parisons, we set $P_{on} = 240W$, $P_{idle} = 150W$ and $P_{off} = 0W$. These values were obtained via our experiments on an Intel Xeon E5320 server, running the CPU-bound LINPACK [13] workload.

8.1 Comparing different policies

An obvious question in data centers is “Which policy should be used to reduce response times and power consumption?” Clearly, no single policy is always superior. In this subsection, we’ll compare the $ON/IDLE$, $ON/OFF$ and $ON/OFF/STAG$ policies for various regimes of setup time and load.

![Graphs showing mean response time and mean power consumption for policies ON/IDLE, ON/OFF, ON/OFF/STAG.](image)

Figure 10: Comparison of $E[T]$ and $E[P]$ for the policies $ON/IDLE$, $ON/OFF$ and $ON/OFF/STAG$. Throughout, we set $\mu = 1job/sec$, $k = 30$ and $P_{on} = 240W$.

Fig. 10 shows the mean response time and mean power consumption for all three policies. With respect to mean response time, the $ON/IDLE$ policy starts out at $E[T] = \frac{1}{\mu}$ for low values of $\rho$, and increases as $\rho$ increases. We observe a similar trend for the $ON/OFF/STAG$ policy, since, by Eq. (6), $E[T_{M/M/k/1\alpha}] = E[T_{M/M/k}] + \frac{1}{\alpha}$. For the $ON/OFF$ policy, we see a different behavior: At high loads, the mean response time increases due to queueing in the system. But for low loads, the mean response time initially drops with increasing load. This can be reasoned as follows: When the system load is extremely low, almost every arrival encounters an empty system, and thus every job
must incur the setup time. As the load increases, however, the setup time is amortized over many jobs: A job need not wait for a full setup time because some other server is likely to become available.

For mean power consumption, it is clear that $E[P_{ON/OFF/STAG}] < E[P_{ON/OFF}]$, since at most one server can be in setup for the $ON/OFF/STAG$ policy. Also, we expect $E[P_{ON/OFF}] < E[P_{ON/IDLE}]$, since we are turning servers off in the $ON/OFF$ policy. However, in the regime where both the setup time and the load are high, $E[P_{ON/OFF}] > E[P_{ON/IDLE}]$, since in this regime, the $ON/OFF$ policy wastes a lot of power in turning servers on, confirming our intuition from Section 6.

Based on Fig. 10, we advocate using the $ON/OFF$ policy in the case of low setup times, since the percentage reduction afforded by $ON/OFF$ with respect to power, when compared with $ON/IDLE$, is higher than the percentage increase in response time when using $ON/OFF$ compared to $ON/IDLE$. For low setup times, the $ON/OFF/STAG$ policy results in slightly lower power consumption than the $ON/OFF$ policy, but results in much higher response times. For the case of high setup times, the $ON/IDLE$ policy is superior to the $ON/OFF$ policy when the load is high. However, when the load is low, the choice between $ON/IDLE$ and $ON/OFF$ depends on the importance of $E[T]$ over $E[P]$.

### 8.2 Mixed strategies: Achieving the best of both worlds

By the above discussions, $ON/IDLE$ is superior for reducing $E[T]$, whereas $ON/OFF$ is often superior for reducing $E[P]$. However, by combining these simple policies, we now show that it is actually possible to do better than either of them. We propose a strategy where we turn an idle server off only if the total number of on and idle servers exceeds a certain threshold, say $t$. We refer to this policy as the $ON/IDLE(t)$ policy. Note that $ON/IDLE(0)$ is the $ON/OFF$ policy, and $ON/IDLE(k)$ is the $ON/IDLE$ policy. To evaluate the $ON/IDLE(t)$ policy, we use Matrix analytic methods.

![Figure 11: Comparison of $E[T]$ and $E[P]$ as a function of the threshold, $t$, for the $ON/IDLE(t)$ policy. Throughout, we set $\mu = 1 \text{ job/sec, } k = 30, \rho = 15$, and $P_{on} = 240W$.](image)

Fig. 11 shows the effect of $t$ on $E[T]$ and $E[P]$. As $t$ increases from 0 to $k$, $E[T]$ decreases monotonically from $E[T_{ON/OFF}]$ to $E[T_{ON/IDLE}]$, as expected. By contrast, under the right setting of $t$, $E[P]$ for the $ON/IDLE(t)$ policy can be significantly lower than both $E[P_{ON/OFF}]$ and $E[P_{ON/IDLE}]$, as shown in Fig. 11 (d), where the power
savings is over 20%. This observation can be reasoned as follows: When \( t = 0 \), we have the ON/OFF policy, which wastes a lot of power in turning servers on. As \( t \) increases, the probability that an arrival finds all servers busy decreases, and thus, less power is wasted in turning servers on. However, as \( t \) approaches \( k \), the ON/IDLE(\( t \)) policy wastes power by keeping a lot of servers idle. At \( t = k \), we have the ON/IDLE policy, which keeps all \( k \) servers on or idle, and hence wastes a lot of power. We define \( t^* \) to be the value of \( t \) at which we get the lowest power consumption for the ON/IDLE(\( t \)) policy. For example, for the scenario in Fig. 11 (d), \( t^* = 17 \). Thus, we expect the ON/IDLE(\( t^* \)) policy to have a lower power consumption than both ON/OFF and ON/IDLE.

### 8.3 Large server farms

It is interesting to ask whether the tradeoff between ON/IDLE and ON/OFF that we witnessed in Section 8.1 becomes more or less exaggerated as the size of the server farm \( (k) \) increases. Using Matrix analytic methods to analyze cases where \( k \) is very large, say \( k = 100 \), is cumbersome and time consuming, since we have to solve a Markov chain with 101 rows. In comparison, our approximations yield immediate results, and hence we use these to analyze the case of large \( k \). Additionally, we compare the performance of the ON/IDLE(\( t^* \)) policy, defined in Section 8.2, with the performance of the ON/IDLE, ON/OFF, and the ON/OFF/STAG policies.

Fig. 12 shows the effect of \( k \) on \( E[T] \) and \( E[P] \) for the policies ON/IDLE, ON/OFF, ON/OFF/STAG, and ON/IDLE(\( t^* \)) where load is always fixed at \( \rho = 0.5k \). Comparing the ON/IDLE and ON/OFF policies, we see that the difference between \( E[P_{ON/OFF}] \) and \( E[P_{ON/IDLE}] \) increases with \( k \), and the difference between \( E[T_{ON/OFF}] \) and \( E[T_{ON/IDLE}] \) decreases with \( k \). Thus, for high \( k \), the ON/OFF policy is superior to the ON/IDLE policy. We also tried other values of \( \frac{\rho}{k} \), such as 0.1 and 0.9. As the value of \( \frac{\rho}{k} \) increases (or, as load increases), the superiority of ON/OFF over ON/IDLE decreases, and in the limit as \( \rho \to k \), both policies result in similar \( E[T] \) and \( E[P] \).

Now, consider the ON/IDLE(\( t^* \)) policy. The mean response time for ON/IDLE(\( t^* \)) is almost as low as the mean response time for ON/IDLE policy, and its mean power consumption is at least as low as the best of the ON/OFF and the ON/IDLE policies, and can be far lower as witnessed in Fig. 11 (d). In all cases, we find that the \( E[T] \) for ON/OFF/STAG is much worse than the other policies, and the \( E[P] \) for ON/OFF/STAG is only slightly better than the ON/IDLE(\( t^* \)) policy.

### 9 Conclusion

In this paper we consider server farms with a setup cost, which are common in manufacturing systems, call centers and data centers. In such settings, a server (or machine) can be turned off to save power (or operating cost), but turning on an off server incurs a setup cost. The setup cost usually takes the form of a time delay, and sometimes there is an additional power penalty as well. While the effect of setup costs is well understood for a single server, multi-server
systems with setup costs have only been studied under very restrictive models and only via numerical methods.

We provide the first analysis of server farms with setup costs, resulting in simple closed-form solutions and approximations for the mean response time and the mean power consumption. We also consider variants of server farms with setup costs, such as server farms with staggered boot up, where at most one server can be in setup at any time. For this variant, we prove that the distribution of response time can be decomposed into the sum of the distribution of response time for an $M/M/k$, and the setup time. Another variant we consider is server farms with infinitely many servers, for which we prove that the limiting probabilities have a product-form: the number of jobs in service is Poisson distributed and is independent of the number of jobs in queue (waiting for a server to setup).

Additionally, our analysis provides us with interesting insights on server farms with setup costs. For example, while turning servers off is believed to save power, we find that under high loads, turning servers off can result in higher power consumption (than leaving them on) and far higher response times. Furthermore, the tradeoff between keeping servers on and turning them off changes with the size of the server farm; as the size of the server farm is increased, the advantages of turning servers off increase as well. We also find that hybrid policies, whereby a fixed number of servers
are maintained in the *on* or *idle* states (and the others are allowed to turn *off*), greatly reduce power consumption and achieve near-optimal response time when setup time is high. Finally, we prove an asymptotic bound on the throughput of server farms with setup costs: as the number of jobs in the system increases to infinity, the throughput converges to a quantity less than the server farm capacity, when the setup time is high.

References


