Performance Evaluation 67 (2010) 1123–1138

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

Server farms with setup costs

Anshul Gandhi[a,*], Mor Harchol-Balter[a], Ivo Adan[b]

[a] Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA

[b] Department of Mathematics and Computer Science, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands

ARTICLE INFO

Article history:
Available online 6 August 2010

Keywords:
Setup times
Power management
Data centers
M/M/k
Staggered boot up

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 are known 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, how to optimally mix policies, 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% [1] 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 the $M/M/k$ queueing system, with a Poisson arrival process with rate λ, and exponentially distributed job sizes, denoted by random variable $S \sim \text{Exp}(\mu)$. Let $\rho = \frac{\lambda}{\mu}$ denote the system

* Corresponding author.
E-mail addresses: anshulg@cs.cmu.edu (A. Gandhi), harchol@cs.cmu.edu (M. Harchol-Balter), iadan@win.tue.nl (I. Adan).

0166-5316/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2010.07.004

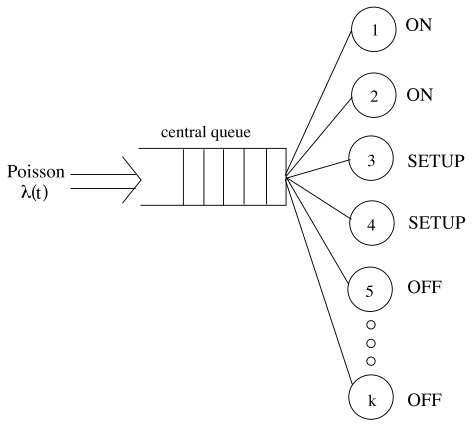


Fig. 1. Illustration of our server farm model.

load, where $0 \le \rho < k$. Thus, for stability, we require $\lambda < k\mu$. In this model, a server can be in one of four states: *on*, *idle*, *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. So $0 = P_{off} < P_{idle} < P_{on}$.

To turn *on* an *off* server, the server 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 $\alpha = \frac{1}{\mathbf{E}[I]}$.

We model our server farm using an $M/M/k$ with a single central First Come First Served (FCFS) queue, from which servers pick jobs when they become free. Fig. 1 illustrates our server farm model. Every server is either *on*, *idle*, *off*, or in *setup* mode.

We consider the following three operating policies:

1. *ON/IDLE*: Under this policy, servers are never turned *off*. Servers all start in the *idle* mode, and remain in the *idle* mode when there are no jobs to serve. All servers are either *on* or *idle*. We model this policy by using the $M/M/k$ queueing system. The response time analysis is well known, and the analysis of power consumption is straightforward, since it only requires knowing the expected number of servers which are *on* as opposed to *idle*.
2. *ON/OFF*: Under this policy, servers are immediately turned *off* when not in use. However, there is a setup cost (in terms of delay and power) for turning *on* an *off* server. At any point in time, there are $i \le k$ *on* servers, and $j \ge i$ jobs in the system, where k is the total number of servers in the system. The number of servers in *setup* is then $\min\{j - i, k - i\}$. The above facts follow from the property that any server not in use is immediately switched *off*. In more detail, there are three types of jobs: those who are currently running at an *on* server (we call these "running" jobs), those that are currently waiting for a server to *setup* (we call these "setting up" jobs), and those jobs in the queue who couldn't find a server to *setup* (we call these "waiting" jobs). An arriving job will always try to turn *on* an *off* server, if there is one available, by putting it into *setup* mode. Later arrivals may not be able to turn *on* a server, since all servers might already be *on* or in *setup* mode, and hence will become "waiting" jobs. Let B be denote the first (to arrive) of the "setting up" jobs, if there is one, and let C be the first of the "waiting" jobs, if there is one. When a "running" job, A, completes service, its server, s_A, is transferred to B, if B exists, or else to C, if C exists, or else is turned *off* if neither B nor C exists. If s_A was transferred to B, then B's server, s_B, is now handed over to job C, if it exists, otherwise s_B is turned *off*. This will become clearer when we consider the Markov chain model for the *ON/OFF* policy.
3. *ON/OFF/STAG*: This model is known as the "staggered boot up" model in data centers, or "staggered spin up" in disk farms [2,3]. The *ON/OFF/STAG* policy is the same as the *ON/OFF* policy, except that in the *ON/OFF/STAG* policy, at most 1 server can be in *setup* at any point of time. Thus, if there are i *on* servers, and j jobs in the system, then under the *ON/OFF/STAG* policy, there will be $\min\{1, k - i\}$ servers in *setup*, where k is the total number of servers in the system. The *ON/OFF/STAG* is believed to avoid excessive power consumption.

Of the above policies, the ON/OFF policy is the most difficult to analyze. In order to analyze this policy, it will be useful to first analyze the limiting behavior of the system as the number of servers goes to infinity. We will analyze two models with infinite servers:

4. *ON/OFF*(∞): This model can be viewed as the *ON/OFF* policy model with an infinite number of servers. Thus, in this model, we can have an infinite number of servers in *setup*.
5. *ON/OFF*(∞)*/kSTAG*: The *ON/OFF*(∞)*/kSTAG* is the same as the *ON/OFF*(∞), except that in the *ON/OFF*(∞)*/kSTAG*, at most k servers can be in *setup* at any point of time.

The infinite server models are also useful for modeling large data centers, where the number of servers is usually in the thousands [4,5]. Throughout this paper, we will use the notation T_{policy} (respectively, P_{policy}) to denote the response time (respectively, power consumption), where the placeholder "policy" will be replaced by one of the above policies, e.g., *ON/OFF*.

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 are known for the *ON/OFF* and the *ON/OFF*(∞). For the *ON/OFF/STAG*, Gandhi and Harchol-Balter have obtained closed-form solutions for the mean response time [6], but no results are known for the distribution of response time.

Results

For the *ON/OFF/STAG*, 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 the *ON/IDLE* and the setup time (see Section 4). For the *ON/OFF*(∞), 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 in the *ON/OFF*(∞) (see Section 5). For the *ON/OFF*, 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 *ON/OFF* 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* (*ON/IDLE* policy). Both of our approximations for the *ON/OFF* 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 the *ON/OFF* (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? (*ON/IDLE* vs. *ON/OFF*)"; "Does it pay to limit the number of servers that can be in *setup*? (*ON/OFF* vs. *ON/OFF/STAG*)"; "Can one create a superior strategy by mixing two strategies with a threshold for switching between them?"; "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 [7] 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 [8], 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 [9–12].

Server farms with setup costs: For the case of multiple servers with setup times, Artalejo et al. [13] consider the *ON/OFF/STAG* 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 [14], the authors consider an inventory control problem, that involves analyzing a Markov chain similar to the *ON/OFF/STAG*. 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 *ON/OFF/STAG* queueing system [6] and derive closed-form results for the mean response time and mean power consumption. However, [6] does not analyze the distribution of the response times for the *ON/OFF/STAG*, as we do in this paper. Importantly, to the best of our knowledge, there is no prior work on analyzing the *ON/OFF*(∞) or the *ON/OFF*. We provide the first analysis of these systems, deriving closed-form solutions and approximations for their mean response time and mean power consumption.

3. *ON/IDLE*

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

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

$$\mathbf{E}\left[P_{ON/IDLE}\right] = \rho \cdot P_{on} + (k-\rho) \cdot P_{idle}. \tag{2}$$

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

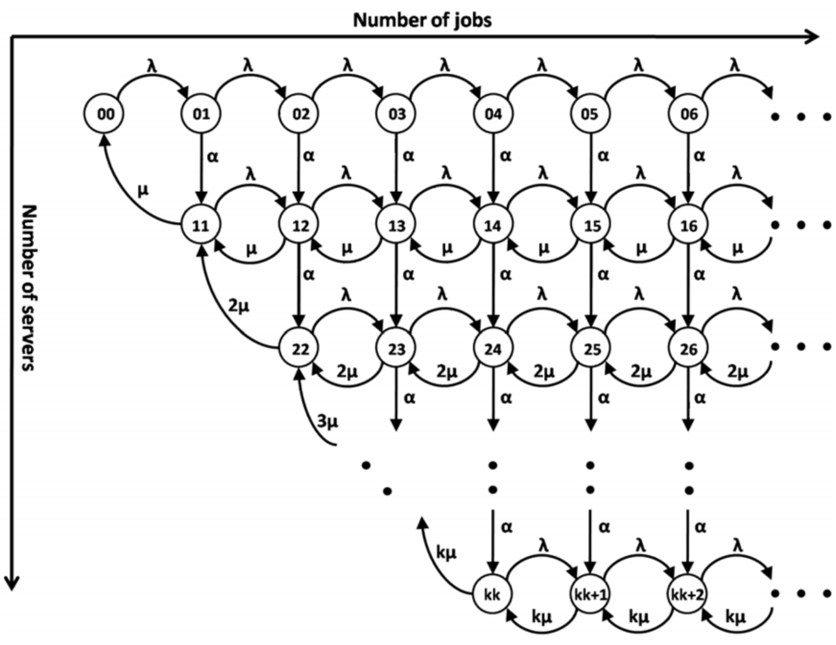


Fig. 2. Markov chain for the *ON/OFF/STAG*.

4. *ON/OFF/STAG*

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 sometimes 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" [2,3]. 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 the *ON/OFF/STAG* 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 [6], Gandhi and Harchol-Balter obtained the limiting probabilities, $\pi_{i,j}$, of the *ON/OFF/STAG* Markov chain using the method of difference equations (see [15] for more information on difference equations), provided for reference in Lemma 1 below.

Lemma 1 ([6]). *The limiting probabilities, $\pi_{i,j}$, for the ON/OFF/STAG are given by:*

$$\pi_{i,j} = \frac{\pi_{0,0} \cdot \rho^i}{i!} \left(\frac{\lambda}{\lambda + \alpha}\right)^{j-i} \quad \text{if } 0 \le i < k \text{ and } j \ge i \tag{3}$$

$$\pi_{k,j} = \frac{\pi_{0,0} \cdot \rho^k}{k!} \left(\left(\frac{\lambda}{\lambda+\alpha}\right)^{j-k} + \frac{\lambda+\alpha}{k\mu - (\lambda+\alpha)} \left[\left(\frac{\lambda}{\lambda+\alpha}\right)^{j-k} - \left(\frac{\rho}{k}\right)^{j-k} \right] \right) \quad \text{if } j \ge k \tag{4}$$

where

$$\pi_{0,0} = \left(1 - \frac{\lambda}{\lambda+\alpha}\right) \cdot \left\{ \sum_{i=0}^{k} \frac{\rho^i}{i!} + \frac{\rho^k}{k!} \frac{\lambda}{k\mu - \lambda} \right\}^{-1}. \tag{5}$$

Below, we show that the limiting probabilities from Lemma 1 can be used to derive the distribution of the response time for the *ON/OFF/STAG*, and simultaneously we prove a beautiful decomposition result: The response time for the *ON/OFF/STAG* can be decomposed into the sum of two independent random variables, one being the response time for the *ON/IDLE* system, and the other the exponential setup time, I. This decomposition result is counter-intuitive since not all jobs will experience the setup time.

Theorem 1. *For the ON/OFF/STAG, with an exponentially distributed setup time $I \sim \text{Exp}(\alpha)$, we have:*

$$T_{ON/OFF/STAG} \overset{d}{=} I + T_{ON/IDLE} \tag{6}$$

where $T_{ON/IDLE}$ is the random variable representing the response time for the ON/IDLE system, and is independent of the setup time, I.

Proof. In order to derive the distribution of response times for the *ON/OFF/STAG*, we'll first derive the z-transform of the number of jobs in the queue,[1] $\hat{N}_Q(z)$. Then, we'll use this to obtain $\tilde{T}_Q(s)$, the Laplace–Stieltjes transform for the time in the queue of the *ON/OFF/STAG*.

Using Lemma 1, the limiting probabilities for the number of jobs in the queue for the *ON/OFF/STAG* can be expressed as:

$$\begin{aligned}\Pr[N_Q = i] &= \pi_{0,i} + \pi_{1,1+i} + \pi_{2,2+i} + \cdots + \pi_{k,k+i}\\ &= \pi_{0,0}\left(\sum_{j=0}^{k}\frac{\rho^j}{j!}\right)\beta^i + \frac{\pi_{0,0}\rho^k(\lambda+\alpha)}{k!(k\mu-\lambda-\alpha)}\left(\beta^i - \left(\frac{\rho}{k}\right)^i\right) \quad \text{where } \beta = \frac{\lambda}{\lambda+\alpha}\end{aligned}$$

$$\begin{aligned}\hat{N}_Q(z) &= \sum_{i=0}^{\infty}\Pr[N_Q=i]\cdot z^i = \sum_{i=0}^{\infty}\pi_{0,0}\left(\sum_{j=0}^{k}\frac{\rho^j}{j!}\right)\beta^i z^i + \sum_{i=0}^{\infty}\frac{\pi_{0,0}\rho^k(\lambda+\alpha)}{k!(k\mu-\lambda-\alpha)}\left(\beta^i-\left(\frac{\rho}{k}\right)^i\right)z^i\\ &= \frac{\pi_{0,0}\left(\sum_{j=0}^{k}\frac{\rho^j}{j!}\right)}{1-\beta z} + \frac{\pi_{0,0}\rho^k(\lambda+\alpha)\lambda z}{k!(k\mu-\lambda z)(\lambda+\alpha-\lambda z)}.\end{aligned}$$

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 the 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 (a departure from the queue refers to a job going into service). 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, $\tilde{T}_Q(s) = \hat{N}_Q\left(1-\frac{s}{\lambda}\right)$. This gives us:

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

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

$$\tilde{T}_Q(s) = \left(\frac{\alpha}{s+\alpha}\right)\left\{(1-P_Q)+P_Q\frac{k\mu-\lambda}{k\mu-\lambda+s}\right\} = \tilde{I}\cdot\tilde{T}_{Q_{ON/IDLE}}(s) \tag{7}$$

where P_Q is the probability of queueing in the *ON/IDLE* system. Thus:

$$\begin{aligned}&T_{Q_{ON/OFF/STAG}} \overset{d}{=} I + T_{Q_{ON/IDLE}}\\ &\Rightarrow T_{ON/OFF/STAG} \overset{d}{=} I + T_{ON/IDLE}. \quad \square\end{aligned}$$

Lemma 2 ([6]). *The mean power consumption in the ON/OFF/STAG is given by:*

$$\mathbf{E}\left[P_{ON/OFF/STAG}\right] = P_{on}\left(\rho + \frac{\lambda}{\lambda+\alpha} - \frac{\pi_{0,0}\rho^k\lambda}{k!\cdot\alpha\cdot\left(1-\frac{\rho}{k}\right)}\right) \quad \textit{where } \pi_{0,0} \text{ is given by Eq. (5).}$$

5. *ON/OFF*(∞)

Many data centers today, including those of Google, Microsoft, Yahoo and Amazon, consist of tens of thousands of servers [4,5]. In such settings, we can model a server farm with setup costs as the *ON/OFF*(∞) system, as shown in Fig. 3. For this model, we make an educated guess for the limiting probabilities.

Theorem 2. *For the ON/OFF*(∞) *Markov chain, as shown in Fig. 3, the limiting probabilities are given by:*

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

$$\textit{where } M(a,b,z) = \sum_{n=0}^{\infty}\frac{(a)_n}{(b)_n}\frac{z^n}{n!} \text{ is Kummer's function [16], and } (a)_n = a(a+1)\cdots(a+n-1),\ (a)_0 = 1. \tag{8}$$

Proof. The correctness of Eq. (8) can be verified by direct substitution into the *ON/OFF*(∞) Markov chain. $\square$

[1] Note that the queue is the waiting room for the jobs. Thus, jobs receiving service are not part of the queue.

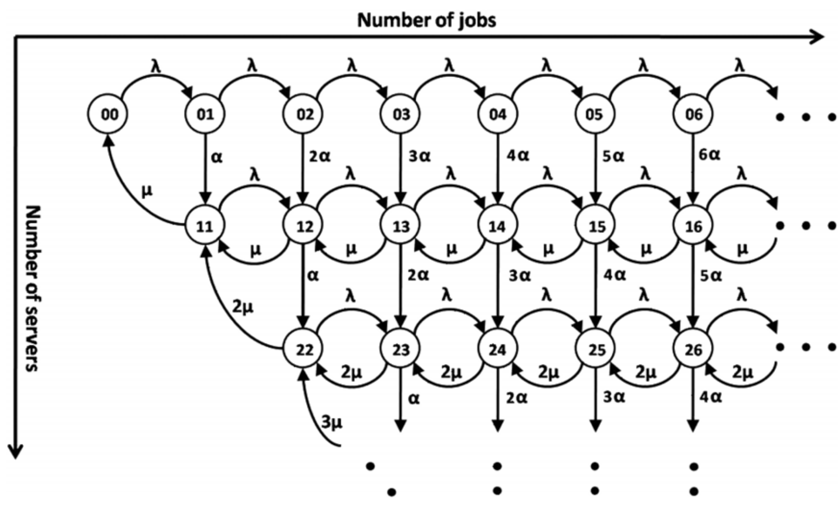


Fig. 3. Markov chain for the *ON*/*OFF*(∞).

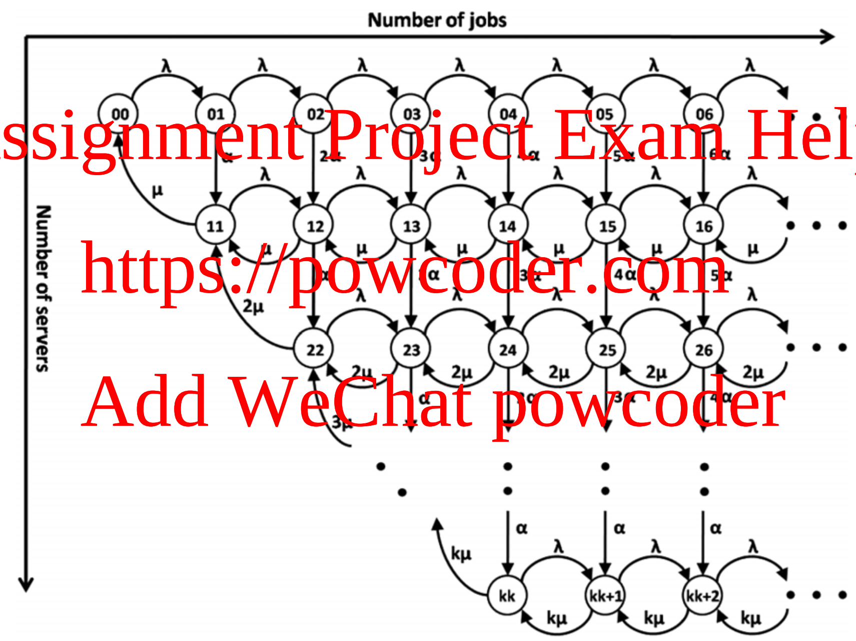


Fig. 4. Markov chain for the *ON*/*OFF*.

The *product-form* solution in Eq. (8) 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 ρ and the number in the queue is distributed as:

$$\Pr[N_{Q_{ON/OFF(\infty)}}=j]=\frac{1}{M\left(1,1+\frac{\lambda}{\alpha},\frac{\lambda}{\alpha}\right)}\prod_{l=1}^{j}\frac{\lambda}{\lambda+l\alpha},\quad \text{with mean } \mathbf{E}\left[N_{Q_{ON/OFF(\infty)}}\right]=\frac{1}{1+\frac{\lambda}{\alpha}}\frac{M\left(2,2+\frac{\lambda}{\alpha},\frac{\lambda}{\alpha}\right)}{M\left(1,1+\frac{\lambda}{\alpha},\frac{\lambda}{\alpha}\right)}.$$

Thus, by Little's law:

$$\mathbf{E}\left[T_{ON/OFF(\infty)}\right]=\frac{1}{\mu}+\frac{1}{\lambda}\mathbf{E}[N_{Q_{ON/OFF(\infty)}}],\qquad \mathbf{E}\left[P_{ON/OFF(\infty)}\right]=P_{on}\left(\rho+\mathbf{E}[N_{Q_{ON/OFF(\infty)}}]\right). \tag{9}$$

6. *ON*/*OFF*: approximations based on the *ON*/*OFF*/(∞)

Under the *ON*/*OFF* 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 *ON*/*OFF* Markov chain, with states (i,j), where i represents the number of servers *on*, and j represents the number of jobs in the system. Given that $j \geq i$ and $i \leq k$, we have exactly $\min\{j-i, k-i\}$ servers in *setup*. Since the Markov chain for the *ON/OFF* (shown in Fig. 4) looks similar to the Markov chain for the *ON/OFF/STAG* (shown in Fig. 2), one would expect that the difference equations method used to solve the *ON/OFF/STAG* should work for the *ON/OFF* too. While we can solve the difference equations for the *ON/OFF*, 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 *ON/OFF* is not tractable in closed-form via difference equations is given in [6]. In this section, we will try to approximate $\mathbf{E}\left[T_{ON/OFF}\right]$ and $\mathbf{E}\left[P_{ON/OFF}\right]$ 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 *ON/OFF* system (which would not have been obtainable via Matrix analytic methods), that we exploit in Section 7 to derive further properties of the *ON/OFF*.

A major goal in analyzing the *ON/OFF* 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.[2] We are most interested in understanding the behavior of the *ON/OFF* 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 *ON/OFF* system using a *ON/OFF*(∞) system, where we truncate the 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 *ON/OFF* system using a truncated version of the *ON/OFF*(∞)*/kSTAG* 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 *ON/OFF* 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 *ON/OFF*(∞)

Consider the Markov chains for the *ON/OFF* (shown in Fig. 4) and the *ON/OFF*(∞) (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 *ON/OFF* using the $\pi_{i,j}$'s for the *ON/OFF*(∞) from Eq. (8), 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 *ON/OFF* using the *ON/OFF*(∞), 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 *ON/OFF*:

$$\pi_{i,j} = \frac{\pi_{0,0}\cdot\rho^i}{i!}\prod_{l=1}^{j-i}\frac{\lambda}{\lambda+l\alpha}, \quad \text{for } i \geq 0,\ i \leq j < k, \text{ where } \pi_{0,0} = \left(\sum_{i=0}^{k-1}\sum_{j=i}^{k-1}\frac{\rho^i}{i!}\prod_{l=1}^{j-i}\frac{\lambda}{\lambda+l\alpha}\right)^{-1}. \tag{10}$$

Using the above limiting probabilities, we can compute the approximations:

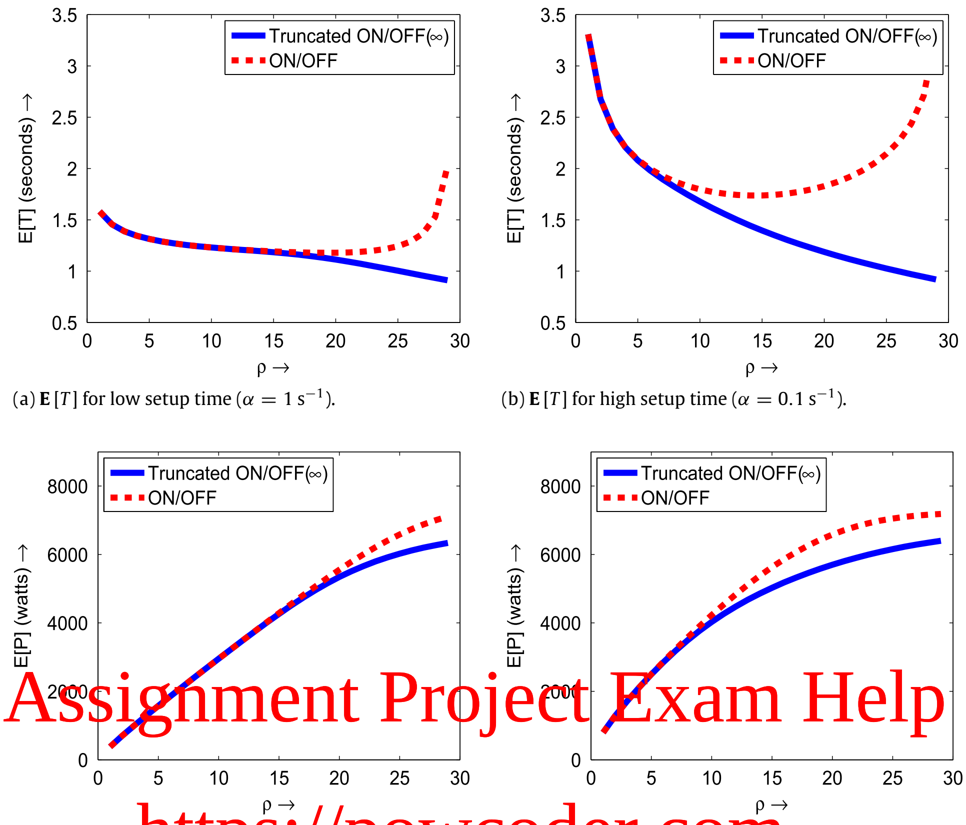
$$\mathbf{E}\left[T_{ON/OFF}\right] \approx \frac{1}{\lambda}\sum_{i=0}^{k-1}\sum_{j=i}^{k-1} j\cdot\pi_{i,j} \quad \text{and} \quad \mathbf{E}\left[P_{ON/OFF}\right] \approx P_{on}\sum_{i=0}^{k-1}\sum_{j=i}^{k-1} j\cdot\pi_{i,j}.$$

Fig. 5 shows our results for $\mathbf{E}\left[T_{ON/OFF}\right]$ and $\mathbf{E}\left[P_{ON/OFF}\right]$ based on the approximation in Eq. (10). We obtain the exact mean response time and mean power consumption of the *ON/OFF* 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 [17] 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. Further, the accuracy of our approximation seems to decrease as the setup time increases.

6.2. Truncated *ON/OFF*(∞)*/kSTAG*

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 *ON/OFF*(∞)*/kSTAG*. For the *ON/OFF*(∞)*/kSTAG* 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 *ON/OFF*(∞)*/kSTAG* model to having no more than k *on* servers. We will refer to this approximation as the truncated *ON/OFF*(∞)*/kSTAG*.

[2] 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 $\mathbf{E}\left[P_{ON/IDLE}\right] = \rho P_{on} + (1-\rho)P_{idle}$ and $\mathbf{E}\left[P_{ON/OFF}\right] = \rho P_{on} + (1-\rho)\frac{\frac{1}{\alpha}}{\frac{1}{\alpha}+\frac{1}{\lambda}}P_{on}$. Thus $\mathbf{E}\left[P_{ON/OFF}\right] \geq \mathbf{E}\left[P_{ON/IDLE}\right] \iff \frac{\alpha}{\lambda} \leq \frac{P_{on}-P_{idle}}{P_{idle}}$, which is true when the setup time is high and the load (or λ) is high.



(a) $\mathbf{E}[T]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).

(b) $\mathbf{E}[T]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).

(c) $\mathbf{E}[P]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).

(d) $\mathbf{E}[P]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).

Fig. 5. Approximations for the *ON/OFF* based on the truncated *ON/OFF*(∞) Markov chain. Figs. (a) and (b) show our results for mean response time in the cases of low setup time and high setup time respectively. Figs. (c) and (d) show corresponding results for mean power consumption. We see that our approximations work well only for low loads. Further, the accuracy of our approximation deteriorates as the setup time increases. Throughout, we set $\mu = 1$ job/s, $k = 30$ and $P_{on} = 240$ W.

For the *ON/OFF*(∞)/*kSTAG* model, we now derive the limiting probabilities. Observe that $\alpha(l)$ in Theorem 3 is a constant for $l \geq k$.

Theorem 3. *For the ON/OFF(∞)/kSTAG, the limiting probabilities for $i \geq 0, j \geq i$ are given by:*

$$\pi_{i,j} = \frac{\pi_{0,0} \cdot \rho^i}{i!} \prod_{l=1}^{j-i} \frac{\lambda}{\lambda + \alpha(l)}, \quad \text{where } \alpha(l) = \min\{k\alpha, l\alpha\}, \quad \text{and} \quad \pi_{0,0} = \sum_{i=0}^{k} \sum_{j \geq i} \frac{\rho^i}{i!} \prod_{l=1}^{j-i} \frac{\lambda}{\lambda + \alpha(l)}. \tag{11}$$

Proof. The correctness of Eq. (11) can be verified by direct substitution into the Markov chain. □

For the *ON/OFF* system, we have a total of k servers. Therefore we now truncate our *ON/OFF*(∞)/*kSTAG* Markov chain to $(k+1)$ rows. That is, we only consider states (i, j) with $i \leq k$. Note that the truncated *ON/OFF*(∞)/*kSTAG* is still not the same as the *ON/OFF*, because, for example, it allows k servers to be in *setup* when there are $(k-1)$ servers *on*, whereas the *ON/OFF* 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 *ON/OFF*(∞)/*kSTAG* model.

Theorem 4. *For the truncated ON/OFF(∞)/kSTAG, the limiting probabilities are given by:*

$$\pi_{i,j} = \begin{cases} \dfrac{\pi_{0,0} \cdot \rho^i}{i!} \displaystyle\prod_{l=1}^{j-i} \frac{\lambda}{\lambda + \alpha(l)} & \text{if } 0 \leq i < k \text{ and } j \geq i \\ \left(\dfrac{\rho}{k}\right)^{j-k+1} \pi_{k-1,k-1} + \displaystyle\sum_{r=k+1}^{j-1} \sum_{l=j+1-r}^{j-k} \left(\frac{\rho}{k}\right)^{l} \pi_{k-1,r} \frac{(r-k+1)\alpha}{\lambda} + \sum_{l=1}^{j-k} \left(\frac{\rho}{k}\right)^{l} \pi_{k-1,j-1} & \\ \quad \text{if } i = k \text{ and } k \leq j \leq 2k & \\ \pi_{k,2k} \left(\dfrac{\rho}{k}\right)^{j-2k} + \dfrac{\pi_{k-1,2k-1}\lambda}{k\mu - (k\alpha + \lambda)} \left[\left(\dfrac{\lambda}{\lambda + k\alpha}\right)^{j-2k} - \left(\dfrac{\rho}{k}\right)^{j-2k} \right] & \text{if } i = k \text{ and } j > 2k. \end{cases} \tag{12}$$

Proof. The limiting probabilities, $\pi_{i,j}$, for $0 \le i < k$ and $j \ge i$, and also for the case of $i = k, j = k$, are identical to the limiting probabilities of the $ON/OFF(\infty)/kSTAG$ model (up to a normalization constant) and are therefore obtained from Eq. (11).

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

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

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

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:

$$\pi_{k,j} = \pi_{k,2k} \left(\frac{\rho}{k}\right)^{j-2k} + C\left[\left(\frac{\lambda}{\lambda + k\alpha}\right)^{j-2k} - \left(\frac{\rho}{k}\right)^{j-2k}\right] \tag{13}$$

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:

$$C = \frac{\pi_{0,0}\lambda\rho^{k-1} \cdot \prod_{l=1}^{k} \frac{\lambda}{\lambda+l\alpha}}{(k(\mu-\alpha)-\lambda)\cdot(k-1)!} = \frac{\pi_{k-1,2k-1}\lambda}{k\mu-(k\alpha+\lambda)} \tag{14}$$

Observe that if we derive the $\pi_{i,j}$ for the special case of a truncated $ON/OFF/STAG$ with infinite servers, then we get back the $\pi_{i,j}$ for the $ON/OFF/STAG$, as in Lemma 1. Finally, $\pi_{0,0}$ can be derived using $\sum_{i=0}^{k}\sum_{j\ge i}\pi_{i,j} = 1$. □

We now approximate the mean response time and the mean power consumption for the ON/OFF system, using the limiting probabilities of the truncated $ON/OFF(\infty)/kSTAG$ model:

$$\mathbf{E}\left[T_{ON/OFF}\right] \approx \frac{1}{\lambda}\sum_{i=0}^{k}\sum_{j=i}^{\infty} j\cdot\pi_{i,j} \qquad \mathbf{E}\left[P_{ON/OFF}\right] \approx P_{on}\cdot\left[\sum_{i=0}^{k-1}\sum_{j=i}^{\infty}(\min\{j, i+k\})\cdot\pi_{i,j} + k\sum_{j=k}^{\infty}\pi_{k,j}\right]. \tag{15}$$

Fig. 6 shows our results for $\mathbf{E}\left[T_{ON/OFF}\right]$ and $\mathbf{E}\left[P_{ON/OFF}\right]$ based on the truncated $ON/OFF(\infty)/kSTAG$ approximation. Again, we obtain the mean response time of the ON/OFF 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 $ON/OFF(\infty)/kSTAG$ overestimates 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 $ON/OFF(\infty)/kSTAG$ allows up to k servers to be in *setup*. Thus, the truncated $ON/OFF(\infty)/kSTAG$ ends up with a lower mean response time than the ON/OFF. Using a similar argument, we expect the mean power consumption of the truncated $ON/OFF(\infty)/kSTAG$ to be higher than the mean power consumption of the ON/OFF, for the case when the load is high and the setup time is high.

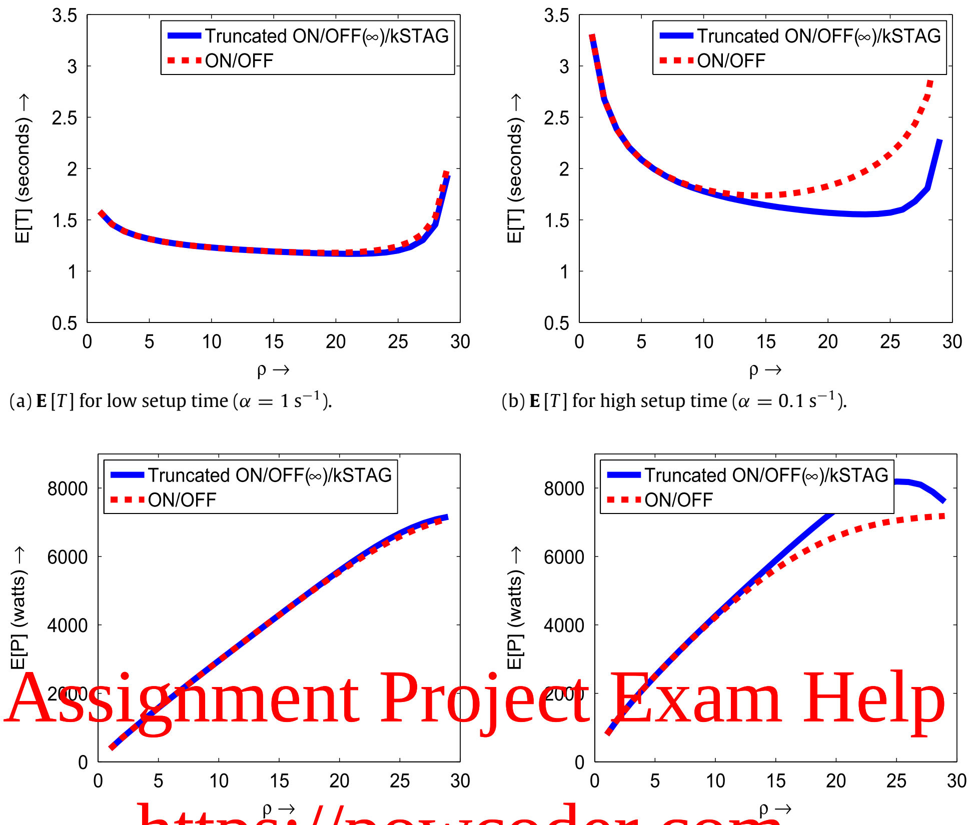
7. *ON/OFF*: asymptotic approximation as the number of jobs approaches infinity

Thus far, we have approximated the ON/OFF model by using the truncated $ON/OFF(\infty)$ model and the truncated $ON/OFF(\infty)/kSTAG$ model, both of which have a 2-dimensional Markov chain. If we can approximate the ON/OFF 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 ON/OFF, $ON(n)$ denotes the expected number of *on* servers, given that there are n jobs in the system,

$$ON(n) = \frac{\sum_{i=0}^{k} i\cdot\pi_{i,n}}{\sum_{i=0}^{k}\pi_{i,n}}. \tag{16}$$

In terms of $ON(n)$, the number of jobs in the ON/OFF can be represented by the random walk in Fig. 7. The remainder of this section is devoted to determining $ON(n)$.



(a) $\mathbf{E}[T]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$). (b) $\mathbf{E}[T]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).

(c) $\mathbf{E}[P]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$). (d) $\mathbf{E}[P]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).

Fig. 6. Approximations for the ON/OFF based on the truncated $ON/OFF(\infty)/kSTAG$ Markov chain. Figs. (a) and (b) show our results for mean response time in the cases of low setup time and high setup time respectively. Figs. (c) and (d) show corresponding results for mean power consumption. We see that our approximations work well under all cases, except in the regime where both the setup time is high and the load is high. Throughout, we set $\mu = 1$ job/s, $k = 30$ and $P_{on} = 240$ W.

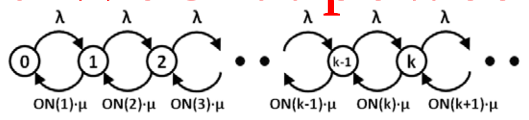


Fig. 7. Markov chain depicting the number of jobs in the ON/OFF, in terms of $ON(n)$.

For $n < k$, we can use the limiting probabilities from the $ON/OFF(\infty)$ model to approximate $ON(n)$, since the Markov chains for both the ON/OFF and the $ON/OFF(\infty)$ are similar for $n < k$, where n denotes the number of jobs in the system. Thus, we can approximate $ON(n)$ for $1 \le n < k$ by Eq. (16), where $\pi_{i,n}$ is given by Eq. (8).

Now, consider $ON(n)|_{n\to\infty}$. 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 ON/OFF, we have:*

$$ON(n)|_{n\to\infty} = \min\left\{k, \frac{\lambda + k\alpha}{\mu}\right\} \tag{17}$$

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

Proof. Consider the ON/OFF Markov chain for states (i, n), where $n > k$, as shown in Fig. 8. Using spectral expansion [18], we have that

$$\pi_{i,n} \sim C v_i w^n \quad \text{as } n \to \infty, \tag{18}$$

for some constant C, where w is the largest zero on $(0, 1)$ of the determinant of the $(k+1)$-dimensional fundamental matrix $A(x)$ and $v = (v_0, \ldots, v_k)$ is a non-null vector satisfying $vA(w) = 0$. The nonzero entries of $A(x)$ are located on the diagonal

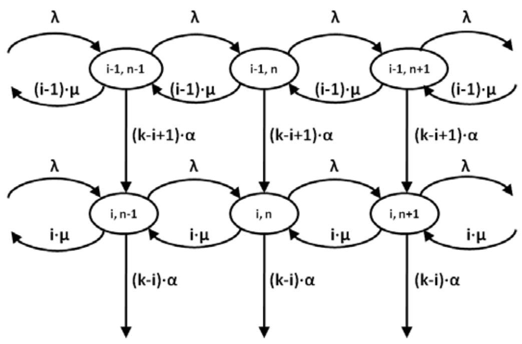


Fig. 8. Markov chain for the *ON*/*OFF*, showing the state (i, n), for $n > k$.

and sub-diagonal, and are defined as

$$A(x)_{i,i} = \lambda - (\lambda + i\mu + (k-i)\alpha)x + i\mu x^2, \qquad A(x)_{i,i+1} = (k-i)\alpha x, \quad i = 0, 1, \ldots, k.$$

The determinant of the fundamental matrix $A(x)$ has exactly $(k+1)$ zeros in $(0, 1)$, denoted by w_i, $i = 0, \ldots, k$, where each w_i is the unique root on $(0, 1)$ 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 zero, with corresponding non-null vector $v = (0, \ldots, 0, 1)$. Thus $ON(n)|_{n\to\infty} = k$. 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 $\pi_{i,n} = v_i w_0^n$ into the balance equation for state (i, n) shown in Fig. 8:

$$v_i = v_{i-1}\frac{(\lambda + k\alpha)\cdot(k-i+1)}{i(k\mu - k\alpha - \lambda)} = v_0\binom{k}{i}\left(\frac{\lambda + k\alpha}{k\mu - k\alpha - \lambda}\right)^i \quad \text{for } i > 0. \tag{19}$$

Thus, by substituting Eqs. (18) and (19) in Eq. (16), we have:

$$ON(n)|_{n\to\infty} = \lim_{n\to\infty}\left(\sum_{i=0}^{k}\pi_{i,n}\right)^{-1}\cdot\sum_{i=0}^{k} i\cdot\pi_{i,n}$$
$$= \left[v_0\left(\frac{k\mu}{k\mu - k\alpha - \lambda}\right)^k\right]^{-1}\cdot\frac{v_0 k(\lambda + k\alpha)}{k\mu - k\alpha - \lambda}\cdot\left(\frac{k\mu}{k\mu - k\alpha - \lambda}\right)^{k-1} = \frac{\lambda + k\alpha}{\mu}.$$

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

Observe that the condition in Eq. (17) suggests that when the setup time is high relative to the service time, then the system throughput is less than $k\mu$. This follows from the fact that a server in *setup* mode will most likely not get a chance to turn *on* because another server will complete servicing a job first, and the setting up job will be transferred there.

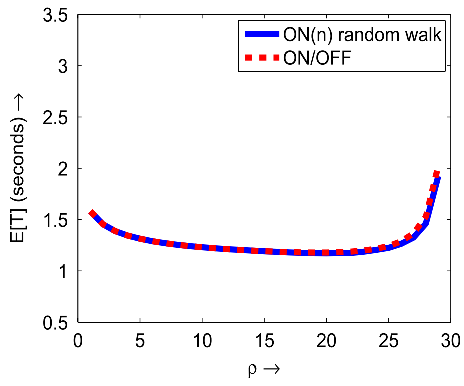
Now that we have $ON(n)$ for $1 \leq n < k$ and $n \to \infty$, we can try and approximate $ON(n)$ for $n \geq k$, using a straight line fit, with points $ON(k-2)$ and $ON(k-1)$, and enforcing that $ON(n) \leq \min\left\{k, \frac{\lambda+k\alpha}{\mu}\right\}$. Given the above approximation for $ON(n)$, we can immediately solve the random walk in Fig. 7 for $\mathbf{E}\left[T_{ON/OFF}\right]$ and $\mathbf{E}\left[P_{ON/OFF}\right]$. Fig. 9 shows our results for $\mathbf{E}\left[T_{ON/OFF}\right]$ and $\mathbf{E}\left[P_{ON}/OFF\right]$ 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

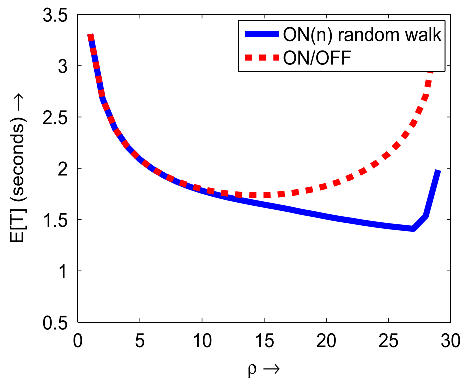
In data centers today, both response time and power consumption are important performance metrics. However, there is a tradeoff between leaving servers *idle* and turning them *off*. Leaving servers *idle* when they have no work to do results in excessive power consumption, since *idle* servers consume as much as 60% of peak power [1]. 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).

We start by revisiting three specific data center policies we defined earlier:

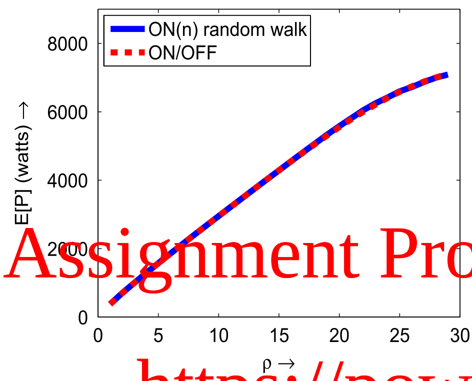
1. *ON*/*IDLE*: Under this policy, servers can either be *on* (consuming power P_{on}), or *idle* (consuming power P_{idle}). This policy was analyzed in Section 3.



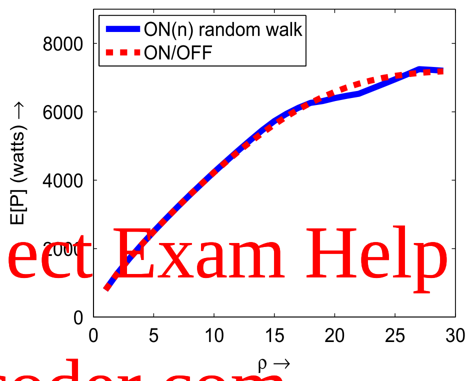
(a) $\mathbf{E}[T]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).



(b) $\mathbf{E}[T]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).



(c) $\mathbf{E}[P]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).



(d) $\mathbf{E}[P]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).

Fig. 9. Approximations for the *ON/OFF* based on the *ON*(n) approximation. Figs. (a) and (b) show our results for mean response time in the cases of low setup time and high setup time respectively. Figs. (c) and (d) show corresponding results for mean power consumption. We see that our approximations work well under all cases, except in the regime where both the setup time is high and the load is high. Throughout, we set $\mu = 1$ job/s, $k = 30$ and $P_{on} = 240$ W.

2. *ON/OFF*: Under this policy, analyzed in Sections 6 and 7, 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 analyzed in Section 4, where at most one server can be in *setup* at a time.

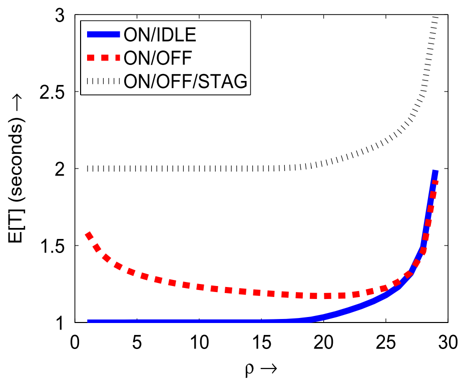
In evaluating the performance of the above policies, we use the approximations and closed-form results derived throughout this paper, except for the case of high setup time and high load, where we resort to Matrix analytic methods. Unless otherwise stated, assume that we are using the approximation in Section 7 for the *ON/OFF*. In our comparisons, we set $P_{on} = 240$ W, $P_{idle} = 150$ W and $P_{off} = 0$ W. These values were obtained via our experiments on an Intel Xeon E5320 server, running the CPU-bound LINPACK [17] workload.

8.1. *Comparing different policies*

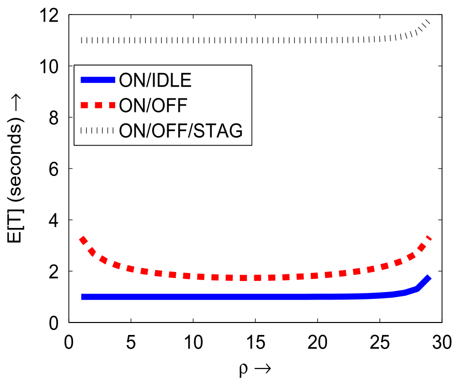
An obvious question in data centers is "Which policy should be used to reduce response times and power consumption?" Clearly, no single policy is always superior. In this subsection, we'll compare the *ON/IDLE*, *ON/OFF* and *ON/OFF/STAG* policies for various regimes of setup time and load.

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 $\mathbf{E}[T] = \frac{1}{\mu}$ for low values of ρ, and increases as ρ increases. We observe a similar trend for the *ON/OFF/STAG* policy, since, by Eq. (6), $\mathbf{E}\left[T_{ON/OFF/STAG}\right] = \mathbf{E}\left[T_{ON/IDLE}\right] + \frac{1}{\alpha}$. 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 an 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.

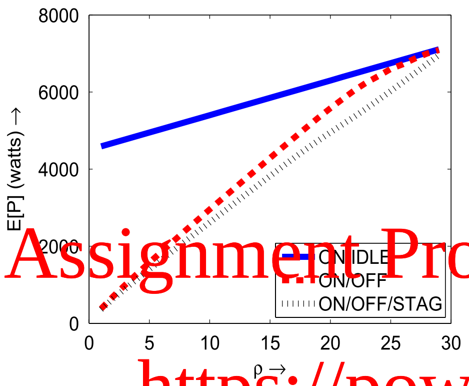
For mean power consumption, it is clear that $\mathbf{E}\left[P_{ON/OFF/STAG}\right] < \mathbf{E}\left[P_{ON/OFF}\right]$, since at most one server can be in *setup* for the *ON/OFF/STAG* policy. Also, we expect $\mathbf{E}\left[P_{ON/OFF}\right] < \mathbf{E}\left[P_{ON/IDLE}\right]$, since we are turning servers *off* in the *ON/OFF* policy.



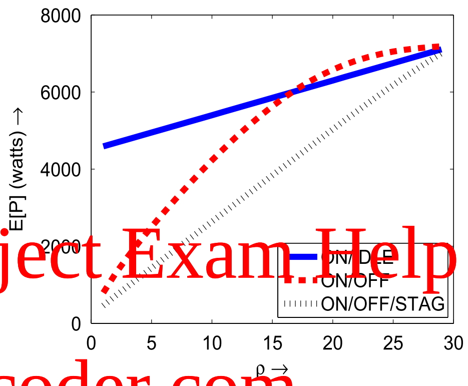
(a) $\mathbf{E}[T]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).



(b) $\mathbf{E}[T]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).



(c) $\mathbf{E}[P]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).



(d) $\mathbf{E}[P]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).

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

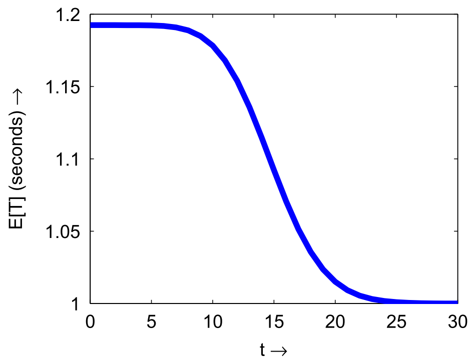
However, in the regime where *both* the setup time and the load are high, $\mathbf{E}\left[P_{ON/OFF}\right] > \mathbf{E}\left[P_{ON/IDLE}\right]$, since in this regime, the *ON/OFF* policy wastes a lot of power in turning servers *on*, confirming our intuition from Section 6.

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 $\mathbf{E}[T]$ over $\mathbf{E}[P]$.

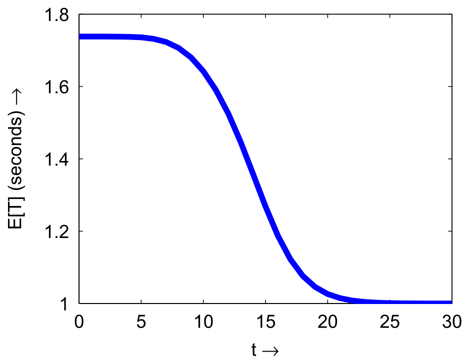
8.2. *Mixed strategies: achieving the best of both worlds*

By the above discussions, *ON/IDLE* is superior for reducing $\mathbf{E}[T]$, whereas *ON/OFF* is often superior for reducing $\mathbf{E}[P]$. 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.

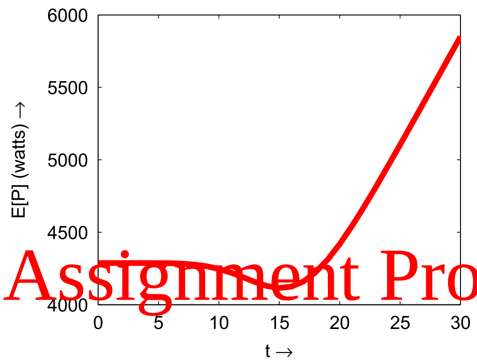
Fig. 11 shows the effect of t on $\mathbf{E}[T]$ and $\mathbf{E}[P]$. As t increases from 0 to k, $\mathbf{E}[T]$ decreases monotonically from $\mathbf{E}\left[T_{ON/OFF}\right]$ to $\mathbf{E}\left[T_{ON/IDLE}\right]$, as expected. By contrast, under the right setting of t, $\mathbf{E}[P]$ for the *ON/IDLE*(t) policy can be significantly lower than both $\mathbf{E}\left[P_{ON/OFF}\right]$ and $\mathbf{E}\left[P_{ON/IDLE}\right]$, as shown in Fig. 11(d), where the power savings are 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*.



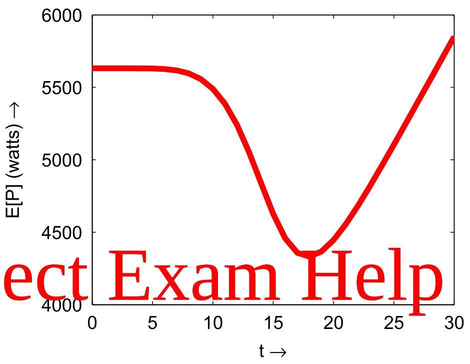
(a) $\mathbf{E}[T]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).



(b) $\mathbf{E}[T]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).



(c) $\mathbf{E}[P]$ for low setup time ($\alpha = 1\ \mathrm{s}^{-1}$).



(d) $\mathbf{E}[P]$ for high setup time ($\alpha = 0.1\ \mathrm{s}^{-1}$).

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

8.3. *Large server farms*

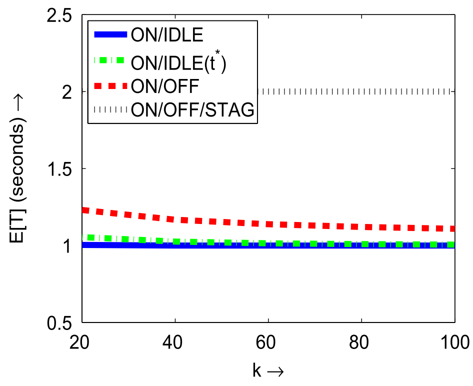
It is interesting to ask whether the tradeoff between $ON/IDLE$ and ON/OFF that we witnessed in Section 8.1 becomes more or less exaggerated as the size of the server farm (k) increases. Using Matrix analytic methods to analyze cases where k is very large, say $k = 100$, is cumbersome and time consuming, 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 $\mathbf{E}[T]$ and $\mathbf{E}[P]$ 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 $\mathbf{E}\left[P_{ON/OFF}\right]$ and $\mathbf{E}\left[P_{ON/IDLE}\right]$ increases with k, and the difference between $\mathbf{E}[T_{ON}/OFF]$ and $\mathbf{E}\left[T_{ON/IDLE}\right]$ 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 $\mathbf{E}[T]$ and $\mathbf{E}[P]$. 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 $\mathbf{E}[T]$ for $ON/OFF/STAG$ is much worse than the other policies, and the $\mathbf{E}[P]$ for $ON/OFF/STAG$ is only slightly better than the $ON/IDLE(t^*)$ policy.

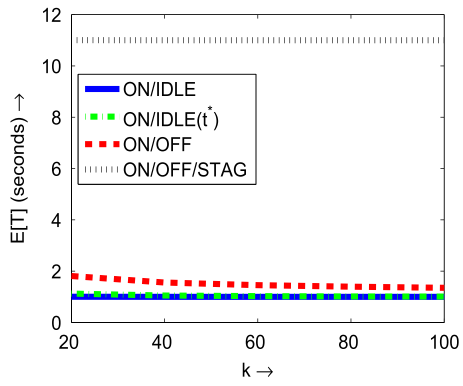
9. Conclusion

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 costs), 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.

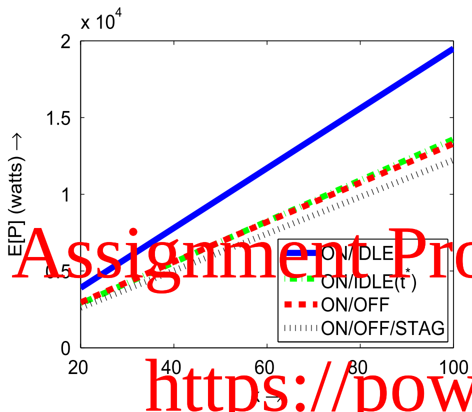
We provide the first analysis of server farms with setup costs, resulting in simple closed-form solutions and approximations for the mean response time and the mean power consumption. We also consider variants of server farms with setup costs, such as server farms with staggered boot up, where at most one server can be in *setup* at any time. For this



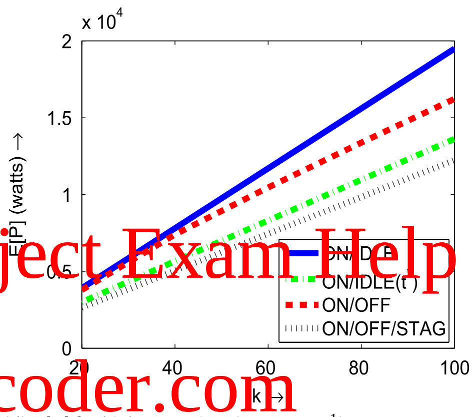
(a) $\mathbf{E}[T]$ for low setup time ($\alpha = 1\ \text{s}^{-1}$).



(b) $\mathbf{E}[T]$ for high setup time ($\alpha = 0.1\ \text{s}^{-1}$).



(c) $\mathbf{E}[P]$ for low setup time ($\alpha = 1\ \text{s}^{-1}$).



(d) $\mathbf{E}[P]$ for high setup time ($\alpha = 0.1\ \text{s}^{-1}$).

Fig. 12. Comparison of $\mathbf{E}[T]$ and $\mathbf{E}[P]$ for the policies *ON/IDLE*, *ON/OFF*, *ON/OFF/STAG*, and *ON/IDLE*(t^*) as a function of k. Throughout, we set $\mu = 1$ job/s, $\frac{\rho}{k} = 0.5$ and $P_{on} = 240$ W.

variant, we prove that the distribution of response time can be decomposed into the sum of the response time for a server farm without setup, 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 the 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. We also find that *mixed* 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 the 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

[1] L.A. Barroso, U. Hölzle, The case for energy-proportional computing, Computer 40 (12) (2007) 33–37. doi: http://dx.doi.org/10.1109/MC.2007.443.
[2] I. Corporation, Serial ATA staggered spin-up, White Paper, September 2004.
[3] M.W. Storer, K.M. Greenan, E.L. Miller, K. Voruganti, Pergamum: replacing tape with energy efficient, reliable, disk-based archival storage, in: FAST'08: Proceedings of the 6th USENIX Conference on File and Storage Technologies, USENIX Association, Berkeley, CA, USA, 2008, pp. 1–16.
[4] CNET news, Google spotlights data center inner workings, 2008. http://news.cnet.com/8301-10784_3-9955184-7.html.
[5] D.C. Knowledge, Who has the most web servers?, 2009. http://www.datacenterknowledge.com/archives/2009/05/14/whos-got-the-most-web-servers.
[6] A. Gandhi, M. Harchol-Balter, $M/G/k$ with exponential setup, Tech. Rep. CMU-CS-09-166, School of Computer Science, Carnegie Mellon University, 2009.
[7] P. Welch, On a generalized $M/G/1$ queueing process in which the first customer of each busy period receives exceptional service, Oper. Res. 12 (1964) 736–752.
[8] H. Takagi, Priority queues with setup times, Oper. Res. 38 (4) (1990) 667–677.
[9] W. Bischof, Analysis of $M/G/1$-queues with setup times and vacations under six different service disciplines, Queueing Syst. 39 (4) (2001) 265–301.

[10] G. Choudhury, On a batch arrival Poisson queue with a random setup time and vacation period, Comput. Oper. Res. 25 (12) (1998) 1013–1026.
[11] G. Choudhury, An $M^X/G/1$ queueing system with a setup period and a vacation period, Queueing Syst. 36 (1–3) (2000) 23–38.
[12] S. Hur, S.-J. Paik, The effect of different arrival rates on the N-policy of $M/G/1$ with server setup, Appl. Math. Model. 23 (4) (1999) 289–299.
[13] J.R. Artalejo, A. Economou, M.J. Lopez-Herrero, Analysis of a multiserver queue with setup times, Queueing Syst. 51 (1–2) (2005) 53–76.
[14] I. Adan, J. van der Wal, Combining make to order and make to stock, OR Spektrum 20 (1998) 73–81.
[15] I. Adan, J. van der Wal, Difference and differential equations in stochastic operations research, 1998. www.win.tue.nl/~iadan/difcourse.ps.
[16] M. Abramowitz, I.A. Stegun, Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, New York, 1964.
[17] Intel Corp., Intel Math Kernel Library 10.0—LINPACK, 2007. http://www.intel.com/cd/software/products/asmo-na/eng/266857.htm.
[18] I. Mitrani, D. Mitra, A spectral expansion method for random walks on semi-infinite strips, in: Iterative Methods in Linear Algebra, 1992, pp. 141–149.

Anshul Gandhi is a Ph.D. student in the Computer Science Department at Carnegie Mellon University, under the direction of Mor Harchol-Balter. His research involves designing and implementing power management policies for datacenters as well as general performance modeling of computer systems.

Mor Harchol-Balter is an Associate Professor of Computer Science at Carnegie Mellon University and also serves as the Associate Department Head for the Computer Science Department. She is heavily involved in the ACM SIGMETRICS/Performance research community and recently served as Technical Program Chair for SIGMETRICS. Mor's work focuses on designing new resource allocation policies (load balancing policies, power management policies, and scheduling policies) for server farms and distributed systems in general. Her research spans both queueing analysis and systems implementation.

Ivo Adan is an associate professor in the department of Mathematics and Computer Science of the Eindhoven University of Technology. Since 2009, he also works as a part-time full professor at the Operations Research and Management group at the University of Amsterdam. His current research interests are in the analysis of multi-dimensional Markov processes and queueing models, and in the performance evaluation of communication, production and warehousing systems. His email address is iadan@win.tue.nl.