Setup Times in Multiserver Systems

Jalani Williams

April 21, 2023

1 Introduction

1.1 Motivation

Multiserver systems matter. Efficient resource provisioning in modern multiserver systems is an important problem. The entire fabric of modern society –from politics, to art, to industry– is interwoven with technology enabled by cloud computing. In the natural sciences, high performance computing forms the backbone of much of our most advanced research. And although computing is perhaps the most prominent example of a multiserver system, our daily lives are filled with them: we wait to purchase things at the supermarket, we wait for a website to load, we wait to be seen at the hospital, etc. In each of these situations, how much time we spend waiting to receive service–the *delay* we experience–can matter a great deal to us. An important determining factor in a multiserver system's delay is its number of servers. As the number of servers increase, the lighter the load each individual server needs to bear, and the shorter a customer must wait before receiving service. As such, one of the primary problems in resource provisioning is deciding how many servers a given multiserver system needs.

Resource provisioning is hard. Unfortunately, determining the right number of servers for a system can be quite difficult, especially since service demand can vary significantly across time [21]. Because demand varies over time, a robust service system must retain enough service capacity that, when demand spikes, quality of service does not suffer. From a datacenter standpoint, this thinking applied at the user-level leads to huge over-subscription of resources, and chronic underutilization of those resources. For examples of both underutilization and demand variation, we examine Google's Borg system in Figure 1.1: utilization rarely reaches above 60%, and there is considerable day-to-day and week-to-week variation in that utilization. Although they have technically been allocated to users, a large fraction of servers are on but not working, leading to incredible energy waste. To mitigate this idling/variability problem, many systems implement some form of dynamic capacity scaling, where the amount of compute allocated to an application are scaled up and down in reaction to its current or estimated utilization.

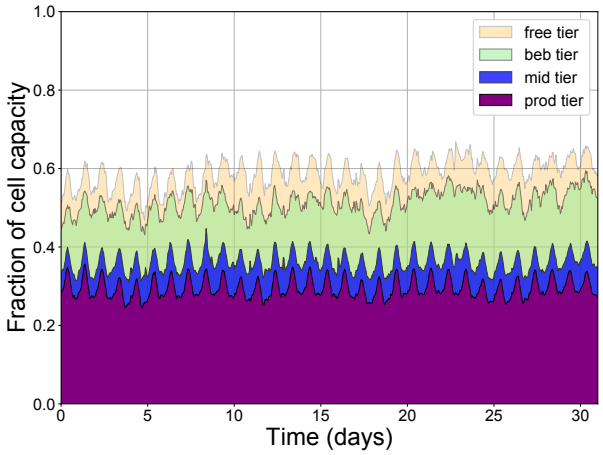

Figure 1: The average utilization over 8 of Google's compute clusters (cells). Note that 1) utilization is low and 2) utilization varies considerably throughout the day. Originally appeared as Figure 2b in [21].

1.2 Problem Statement

While some companies have had great success in increasing the efficiency of their systems by using dynamic scaling, those efficiency gains are necessarily not without performance losses. Because servers do not turn on instantly but require a *setup time* before they are ready to serve jobs, the delay performance of a dynamically-scaled system must be necessarily worse than a system where all servers are kept on at all times. Thus, each dynamic scaling policy poses two unique problems. When using a given policy, 1) "how much worse is the delay performance?" and 2) "how much better is the energy efficiency?" We focus on the first question, asking more generally: **"How do setup times affect the delay performance of modern multiserver systems?"** Given the large space of possible scaling policies and the risk involved in deploying a bad scaling policy, a theoretical understanding of this question could prove quite useful for the prospective provisioner.

1.2.1 Thesis Statement

In this investigation of the negative effect of setup times, I demonstrate that **setup times can not be ignored. Setup times can cause profound increases in waiting time, especially when the distribution of setup time has low variability.**

1.2.2 Main Obstacle

The main obstacle to analyzing multiserver systems with setup times is finding a way to handle their complicated dynamics. Consider the M/M/k queueing system without setup times. The M/M/k is a simple birth-death process where the departure rate is a very simple function of the number of jobs $N(t)$, which is the only state variable. When augmenting the system with setup times, one must separately consider the number of jobs in the queue $Q(t)$ and the number of jobs in service $Z(t)$. Even if one models setup times as memoryless by making the setup time distribution Exponential, the resulting continuous-time Markov Chain still has no known closed-form solution; the state-of-the-art for these problems is an algorithm for steady-state expectations of interest, when analyzing the set of systems with a specific number of servers [11]. In this work, we aim to bound the performance of real multiserver systems with setup, and so must do away with the Exponential assumption, making things apparently even harder. Now, instead of just tracking the jobs in queue $Q(t)$ and the jobs in service $Z(t)$, we must also track the the remaining setup time of each server, the vector $\boldsymbol{W}(t)$.

1.3 Summary of the State-of-the-Art

Summary. Many people have recognized the importance of studying dynamic capacity scaling. The study of the negative effect caused by setup times arguably began with Welch in [22], where he analyzed the delay behavior of the M/G/1 with general setup times. Since then, many works have attempted to progress our understanding of setup times; some by extending Welch's results to new service disciplines[2, 12, 3] and others by investigating the effect of setup in multiserver systems [1, 7, 11, 19, 5, 18, 17, 16, 10, 6]. Unfortunately, previous research around dynamic capacity scaling has had two major shortcomings.

First major shortcoming: setup ignored. First, most research has ignored or elided the effect of setup times. One can easily see why: including a "remaining setup time" value for each server can hugely increase the dimensionality of a service system state space, and make the dynamics hard to analyze. However, there are only two cases where it would be reasonable to ignore the effect of setup times on queueing: if setup times are either extremely small or extremely large. If setup times are extremely small, then they can have only a correspondingly small effect on response time. On the other hand, if setup times are extremely large, then it no longer makes sense to consider the dynamics of setup on the behavior of a system.

However, in modern multiserver systems, neither of these are the case: setup times are short enough that we can reasonably consider dynamically provisioning servers to keep the queue from getting too large, but not so short that if every job had to wait for an entire setup time, then that would be a negligible addition to the delay. In reality, modern setup times can be around a minute while desired

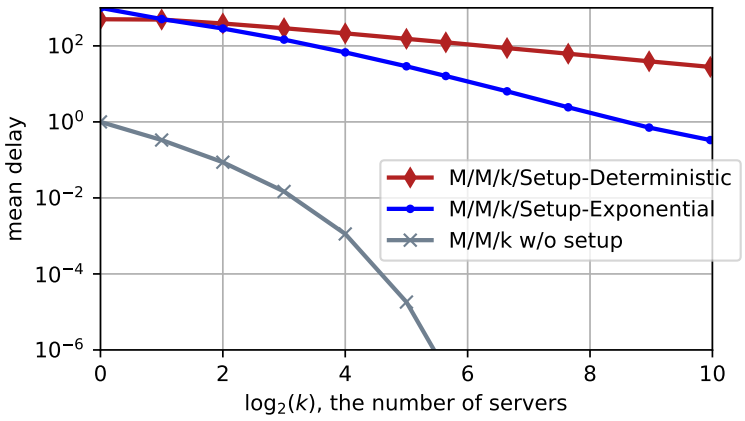

Figure 2: Simulation results for the M/M/k/Setup-Deterministic, M/M/k/Setup-Exponential, M/M/k (no setup), with $\mu = 1$, setup time $\frac{1}{\alpha} = 1000$, and load kept at a constant $\rho = 0.5$. While there's a huge difference between no-setup and setup, there's also a considerable difference in the shape of the curves for Exponential and Deterministic setups.

response times have shrunk to the tens of milliseconds –setup times can now be a *thousand-fold* larger than service times [10, 15, 20].

Moreover, we know from simulation that when the number of servers gets large, systems with setup times have much larger delay than systems without setup. In Figure 2, we hold the load and setup time constant while varying the number of servers k and observe that the delay of systems with setup seem to decay only polynomially with k, while the delay of a system without setup is known to decay exponentially quickly. This difference in delay scaling behavior leads to observed delays which are millions of times larger in systems with setup. Even so, setup times continue to be largely ignored theoretically, due to the dimensionality issue mentioned earlier.

Second major shortcoming: Exponential setup times. To sidestep this dimensionality problem, theoretical works which study the effect of setup times in multiserver systems universally assume that a server's setup time is i.i.d. Exponential; this Exponential assumption, it turns out, is the second major shortcoming of previous research. In reality, setup time distributions are much less variable than the Exponential –a virtual machine's boot time, for example, is essentially Determinstic after conditioning on the boot image size [15]. More to the point, modeling setup times as Exponential can hide the negative effect that setup time can have in large systems. Investigating this via simulation, we simulated the behavior of an M/M/k under a natural scaling policy, and plotted the expected time in queue as the number of servers k increased. Shown

again in Figure 2, our results demonstrate that models using Exponential setup can have expected queueing times which are orders of magnitude smaller than models using Deterministic setup. Consequently, those using the behavior of an Exponential system to predict the behavior of a Deterministic system would *vastly underestimate* the negative effect that setup times have on queueing performance.

1.4 Key Ideas

As mentioned before, the main difficulty in analyzing the M/M/k with setup times lies in keeping track of the complicated dynamics of the remaining setup time vector $\boldsymbol{W}(t)$, and tracking the interaction of $\boldsymbol{W}(t)$ with the number of busy servers $Z(t)$. Before I discuss how our method enables us to bypass this difficulty, I briefly explain our method.

Method: Stopping Time Decomposition. The method we have developed proceeds in three steps. First, one applies the Renewal Reward theorem, which states that, for a particular kind of stopping time X,

$$\mathbb{E}\left[Q(\infty)\right] = \frac{\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right]}{\mathbb{E}\left[X\right]} = \frac{\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right]}{\mathbb{E}\left[\int_0^X 1\mathrm{d}t\right]}, \tag{1}$$

where the final state $\mathcal{S}(X)$ is the same as the initial state $\mathcal{S}(0)$. Thus, to bound $\mathbb{E}\left[Q(\infty)\right]$, it suffices to bound the integrals in the numerator and denominator of (1). In the second step, we take the interval $[0, X)$ and use various stopping times to divide it into even more intervals; we divide our integral over these same stopping times. After making the intervals of interest sufficiently small, we use coupling to bound the dynamics over these small intervals, and use martingale theory to bound the integrals of our coupled processes.

How it helps. Defining our stopping times in the right way allows us to bypass many of the problem's natural difficulties for three reasons. First, we can define our stopping times so that we reduce our focus to only the most critical moments during a renewal cycle, e.g., in an upper bound, we can track the first moment the queue exceeds a certain size. Second, we can define our stopping times so that we gain information about the current state, which allows us to have a tight handle on the system's dynamics in the near-future. Third, since the minimum between two stopping times is itself a stopping time, we can combine both of these approaches to get a very detailed breakdown of a given renewal cycle.

1.5 Key Results

Main result. To support this statement, I provide the first analysis of a dynamically-scaling multiserver system with Deterministic setup times. In par-

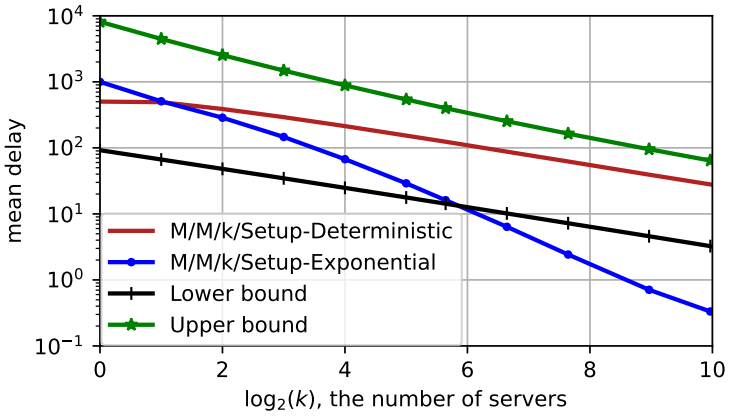

Figure 3: Comparison of our lower and upper bounds for the M/M/k/Setup-Deterministic with simulation results for the M/M/k/Setup-Deterministic and M/M/k/Setup-Exponential, with $\mu = 1$, setup time $\frac{1}{\alpha} = 1000$, and load kept at a constant $\rho = 0.5$. Our bounds clearly follow the shape of the M/M/k/Setup-Deterministic. The shape of the M/M/k/Setup-Exponential is quite different.

ticular, I characterize the expected queueing time $\mathbb{E}[T_Q]$ in the M/M/k/Setup-Deterministic, a variation on the M/M/k queueing system, showing that

Theorem 1 (Simplified scaling version)**.**

$$\mathbb{E}[T_Q] = \Theta\left(\textit{[setup time]}\frac{1}{\sqrt{k\rho}}\right),$$

where the load ρ is the time-average fraction of servers actively working on jobs and where in the scaling of Θ, the load ρ must be kept in the sub-Halfin-Whitt regime while k and [setup time] jointly scale, with $\frac{\text{[setup time]}}{\text{service time}} \geq 4\ln(k\rho)$. For reference, if $\frac{\text{[setup time]}}{\text{service time}} \geq 1000$, then this $\ln(k\rho)$ condition remains satisfied all the way up until $k \geq 10^{100}$.

Beyond the result proven, the method I have developed provides explicit lower and upper bounds on $\mathbb{E}[T_Q]$, as seen in Figure 3. This method uses a combination of renewal theory, martingales, and coupling to reduce the problem of bounding $\mathbb{E}[Q(\infty)]$ to some more basic probability problems. Given its success at solving this long-standing problem, this method itself could be of independent interest.

1.6 Takeaways

I now discuss the takeaways of our theorem, using Figure 3 as an illustrative example. In Figure 3, we plot the mean delay of a job in 1) the M/M/k without

setup, 2) the M/M/k with Exponential setup times, and 3) the M/M/k with Deterministic setup times, along with our proven delay bounds. We scale the number of jobs k while holding the load $\rho = 0.5$, service time $\frac{1}{\mu} = 1$s, and setup time $\frac{1}{\alpha} = 1000$s constant. We use a log-log scale to better illustrate the difference in scaling behavior; in this log-log plot, functions decaying polynomially in k will appear like straight lines (with negative slope), while functions decaying exponentially in k will appear like negative exponentials, since

$$\ln\left(e^{-ck}\right) = -ck = -c2^{\log(k)}.$$

We can plainly see the difference in scaling behavior between the no setup system, the Exponential setup system, and the Deterministic setup system. As expected classically, the mean delay of the no setup system is decaying exponentially in k. And, in confirmation of our bounds, the mean delay of the Deterministic setup system can be seen to decay like $k^{-1/2}$. As predicted, our bounds seem to be off by a constant factor, and our lower bound is, at least in this plot, able to separate the performance of the Exponential and Deterministic systems.

1.7 Impact

The most direct impact of Theorem 1 is on provisioning.For example, suppose we chose to use the M/M/k/Setup policy in our datacenter and we were wondering how many servers we need in order to secure good delay performance; moreover, assume that, as usual, that setup times are deterministic and large in this datacenter. Theorem 1 tells us two things.

First it tells us that, if we keep the offered load $k\rho$ the same while increasing the number of servers k, then the queueing performance of our system *won't actually improve*. Our analysis shows that, instead, the number of busy servers will simply perpetually oscillate around $k\rho$, with most of the servers unused. In some sense, the system is kept perpetually in the Halfin-Whitt regime, with $\rho_{\text{eff}} = 1 - \frac{\text{[service time]}}{\text{[setup time]}} \frac{1}{\sqrt{k\rho}}$; up to a point, it actually *improves* performance to scale up the offered load $k\rho$.

Furthermore, if a system provisioner tried to figure out how big they would need to scale up their system before they reached a desired mean delay, and they modelled their system using either the M/M/k without setup or the M/M/k with Exponential setup, then they would *vastly* underestimate how much their system must grow. We illustrate this point in Figure 4. As before, we set the service time to 100 ms, the load ρ to a constant 0.5, the ratio $\frac{\text{[setup time]}}{\text{[service time]}}$ to 1000, and examine the drop in mean delay as the number of servers k increases. We then check how large the k must become before each system reaches the delay target. Comparing them, the no-setup system reaches the target delay when $k = 2$, the Exponential-setup system reaches the target delay when $k = 1024$, but even the *lower bound* for the Deterministic-setup system only reaches the target delay when $k = 65536$. When it comes to provisioning problems like these, it is absolutely vital that the provisioner models setup times as Deterministic.

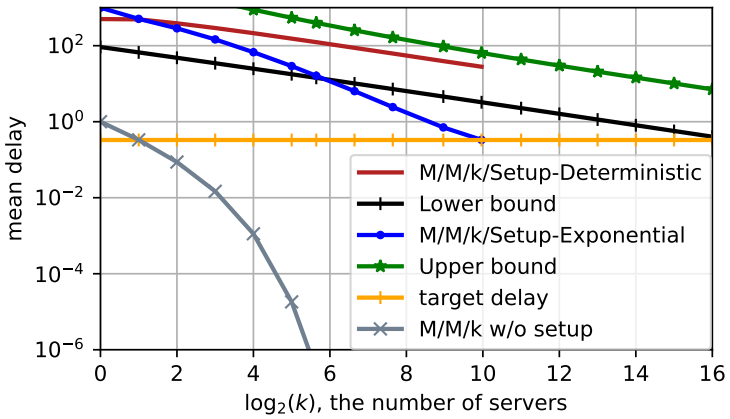

Figure 4: Provisioning comparison of our lower and upper bounds for the M/M/k/Setup-Deterministic with simulation results for the M/M/k/Setup-Deterministic and M/M/k/Setup-Exponential, with [service time] = 100 ms, setup time $\frac{1}{\alpha} = 100$ s, and load kept at a constant $\rho = 0.5$. Comparing how quickly the mean delay reaches the target delay of 33 ms, we find that the no-setup system reaches it at $k = 2$, the Exponential system at $k = 2^{10}$, and the lower bound for the Deterministic system at $k = 2^{16}$. The no-setup and Exponential systems greatly underestimate the needed value of k.

1.8 Outline

In the remainder of this document, I explain my thesis work in more detail, and propose a few additional problems whose solutions I think will greatly enhance the overarching story being told here. To be specific:

- In Section 2, I give an in-depth description of M/M/k/Setup model.
- In Section 3, I give a brief summary of the previous work analyzing multiserver systems with setup times.
- In Section 4, I explain a few of the key techniques used in the proofs of the main results.
- In Sections 5 and 6, I describe the approach we take to prove the lower bound (and upper bound, resp.) on $\mathbb{E}[T_Q]$, and give a brief summary of my status on each project.
- In Section 7, I outline a few problems which I believe would augment my existing work most convincingly.
- In Section 8, I lay out a timeline for completing the remaining work and writing up the final thesis document.

2 Model

We begin by formally describing our model, referred to as the M/M/k/Setup-Deterministic, which is a variant on the M/M/k queueing system. An example is illustrated in Figure 5. Just as in the M/M/k model, in our model there are k servers, indexed by $1, 2, \ldots, k$; jobs arrive following a Poisson process of rate $k\lambda$ into a central queue, and job service times follow an exponential distribution with rate μ. The load of the system is denoted as $\rho \triangleq \frac{\lambda}{\mu}$, and the quantity $R \triangleq k\rho$ is referred to as the offered load of the system following the convention.

To augment the M/M/k with setup times, we make the following adjustments. When a server completes a job and there are no jobs waiting in the queue, the server turns off. Now if we want to turn an off-server back on, it requires a setup time. We assume that the setup times are deterministic with value $\frac{1}{\alpha}$. This is in contrast to the regular M/M/k model, where all the servers are on all the time. With setup times, the dynamics of the system can be described as follows.

- **When a job arrives:** (i) If there are off servers, an off server initializes setup. To avoid ambuguity, we assume that in this case the off server with the smallest index initializes setup. In the example in Figure 5, if a job arrives to the pictured state, server 5 initializes setup. (ii) If all the servers are either on or in setup, their states do not change.

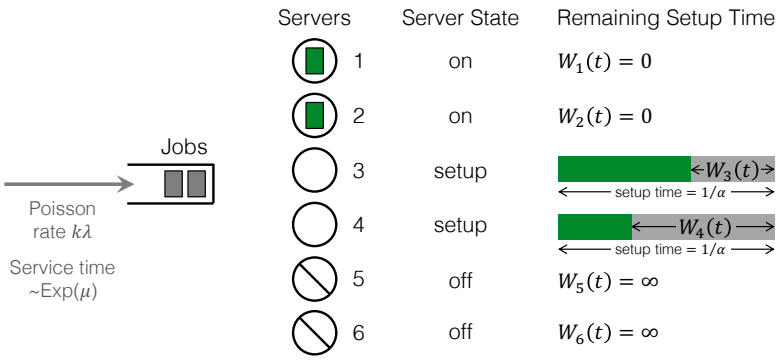

Figure 5: An example of M/M/k/Setup-Deterministic with $k = 6$. The state pictured has $Z(t) = 2$ busy servers, which means there are 2 jobs in service. There are $Q(t) = 2$ jobs in queue, and thus $N(t) = Z(t) + Q(t) = 4$ jobs in system.

- **When a job is completed on a server:** (i) If the queue is empty, the server turns off. (ii) If the queue is nonempty, the server puts the head-of-queue job into service. Then if the number of jobs left in the queue is smaller than the number of servers in setup, i.e., we are setting up more servers than needed, the setup on the server with the largest remaining setup time is canceled, returning the server back to off. In the example in Figure 5, if server 1 completes the job currently in service, server 1 starts serving the head-of-queue job, the setup on server 4 is canceled, and server 4 returns to off.

Now we explain the notation we use to describe the state of the system at time t. An example is again given in Figure 5. Let $Z(t)$ be the number of on servers (busy servers). Then $Z(t)$ is also the number of jobs in service. Let $Q(t)$ be the number of jobs in queue (not including the jobs in service), and $N(t) = Z(t) + Q(t)$ be the total number of jobs in system. We describe the remaining setup times of servers via a size k vector $\boldsymbol{W}(t)$, where, if server i is in setup, the entry $W_i(t)$ is the remaining amount of time that the i-th server needs to finish setting up before turning ON. If server i is not in setup, we set $W_i(t)$ in the following way for convenience: if server i is on, we set $W_i(t) = 0$; if it is off, we set $W_i(t) = \infty$. Then it is clear that the process

$$\Big\{\mathcal{S}(t) \triangleq (Z(t), Q(t), \boldsymbol{W}(t)) \colon t \in \mathbb{R}_+\Big\}$$

is a Markov process. Let $s = (z, q, \boldsymbol{w})$ represent a realization of the state. We drop the time index and simply write $\mathcal{S}, Z, Q, N$ and $\boldsymbol{W}$ to represent the corresponding quantities in steady state. We note that by Little's law, the mean number of busy servers in steady state is equal to the offered load, i.e., $\mathbb{E}[Z] = R = k\rho$.

We now define some notation which will prove useful in our later analyses. For convenience's sake, we use the notation $A(t_1, t_2)$ to denote the number of arrivals which arrival during the interval $[t_1, t_2)$. Likewise, we let $D_j(t_1, t_2)$ denote the number of (potential) departures from the first j servers. Taking Π's to be unit rate Poisson processes, we can define this as

$$A(t_1, t_2) = \Pi_A\left(k\lambda t_2\right) - \Pi_A\left(k\lambda t_1\right)$$

and

$$D_j(t_1, t_2) = \sum_{i=1}^{j}\left[\Pi_i\left(\mu(t_2 - t_1) + \mu\int_0^{t_1}\mathbf{1}_{\{Z(x)\geq i\}}\mathrm{d}x\right) - \Pi_i\left(\mu\int_0^{t_1}\mathbf{1}_{\{Z(x)\geq i\}}\mathrm{d}x\right)\right],$$

where the true number of departures from the system is

$$\mathrm{Dep}(t) = \sum_{i=1}^{k}\Pi_i\left(\mu\int_0^{t}\mathbf{1}_{\{Z(x)\geq i\}}\mathrm{d}x\right).$$

3 Related Work

In this section, I discuss some related works which also analyze the effect of setup time in queueing systems. Although we have found no theoretical work analyzing multiserver systems with deterministic setup times, there is a rich history of work around the analysis of queueing systems with setup.

Single server The case of setup time in a single server has been understood since the 1960's. Welch, in [22], considers a slight generalization of the M/G/1/setup queue where, if a customer arrives while the server is idle, then they have a different service distribution than if they arrive while the server is busy. Welch characterizes the steady-state and transient distributions of the queue length and delay. This important result has been extended in a variety of different directions, both by adjusting the service discipline and by adjusting the arrival process [2, 12, 3].

M/M/k and M/G/k with staggered setup The easiest case of multiserver systems with setup times involves the *staggered setup* model, where at most one server can be in setup at a time, greatly simplifying the analysis. In [1], the authors obtain an expression for the steady-state distribution of queue length for the system when setup times are Exponential, using the method of difference equations. In [7], the authors simplify the solution of the M/M/k with exponential setup times considerably, and prove a decomposition result for mean delay. In [5], the decomposition result is generalized to a hyperexponential job size distribution, and shown to hold approximately for a general job size distribution.

M/M/k/Setup-Exponential, Approximations Most of the results that deal with an M/M/k/Setup system assume Exponential service times and are approximate. In particular, we highlight the work in [18] and [7]. Gandhi et al. [7] seek useful intuitive approximations to the M/M/k/Setup-Exponential system. Their approximations stem from an exact analysis of the M/M/∞/Setup-Exponential system, which they then modify in various ways to capture the finite server case. The approximations in [7] work well, except when both load and setup times are moderately high ($\rho > 0.5$ and $\frac{\mu}{\alpha} > 10$).

Pender and Phung-Duc [18] consider a generalization of the M/M/k/Setup-Exponential model which includes non-stationary arrival rate and customer abandonment. Within this model, they derive a mean field approximation for the system dynamics, which they prove converges as the number of servers, k, approaches infinity.

Unlike our work, neither Pender and Phung-Duc [18] nor Gandhi et al. [7] provide explicit bounds on the delay. The approximations themselves are also not stated as an explicit function of the system parameters. Finally, neither considers Deterministic setup times.

M/M/k/Setup-Exponential, Exact Analysis There are only a few results that deal with the exact analysis of the M/M/k with Exponential setup times. The most well-known are [11] and [19]. Gandhi et al. [11] give the first exact analysis of the M/M/k/Setup-Exponential system. To do this, they develop the *Recursive Renewal Reward (RRR)* technique for solving the corresponding Markov chain, algorithmically. Gandhi et al. [11] use RRR to obtain the Laplace transform of delay for any particular value of k, but do not provide a formula as a function of k. Phung-Duc [19] rederives the exact solutions from [11] using generating functions and matrix-analytic methods.

While [11] and [19] are important in that they provide the first exact analysis, their algorithms actually take $O(k^2)$ time to compute. They also do not provide good intuition for the structure of the *solution*, i.e., how the different system parameters (mean setup time, mean service time, arrival rate, number of servers) affect the delay behavior of the system.

Distributed setting Setup times have also been looked at in distributed systems where a dispatcher routes each incoming job to one of several queues. Mukherjee et al. [17] describe a token-based load balancing and scaling scheme called TABS that takes into account of setup times on the individual queues. They prove that the performance of TABS (as $k \to \infty$) is asymptotically optimal. While [17] assumes that the queues have finite buffers, Mukherjee and Stolyar [16] generalize their results to infinite buffers. The nature of the questions being asked and answered in [17] and [16] are very different from the central queue-based work we discuss.

M/G/2/Setup-Deterministic, with dispatching In the control literature, deterministic setup times have been incorporated into models in order

to enhance realism. Hyytiä et al. [13] consider a dispatching version of the M/G/2/Setup-Deterministic model, and attempt to build near-optimal policies for the joint control of setup initiation and the dispatching of jobs. We hope that our analysis here could open the door to more fine-grained stochastic analysis of such control policies.

M/M/k/Setup-Deterministic, simulation only The only work we have found which discusses the M/M/k/Setup-Deterministic model explicitly is a simulation-based thesis by Kara [14]. Their simulation results corroborate the argument we make in Section 1. In particular, they observe that the mean delay in the M/M/k/Setup-Deterministic is consistently larger than that of the M/M/k/Setup-Exponential, and, as the mean setup time $\frac{1}{\alpha}$ increases, the relative increase in mean delay between the M/M/k/Setup-Deterministic and the M/M/k/Setup-Exponential also increases.

Algorithms for reducing the effect of setup times on delay and energy usage Setup times are both a problem from a delay perspective and also from an energy perspective (servers utilize peak power while in setup [10]). One can of course avoid setup times altogether by always leaving servers on, but this results in wasted energy as well, since a server which is on, but idle, utilizes 60-70% of peak energy [10]. To manage power efficiently, several algorithms have been developed to reduce the costly effects of setup times. One idea is *DelayedOff*, whereby a one waits some time before turning off a server, so as to avoid a future setup time [6, 7, 10, 18]. Another idea is routing jobs to the *Most Recently Busy server (MRB)*, so as to minimize the size of the pool of servers that are turning on and off [6]. Similar to MRB is the idea of creating a rank ordering of all servers and always sending each job to the *lowest-numbered server in the rank* [10]. The goal of all such algorithms is to minimize the Energy-Response-time-Product (ERP) [6], maximize the Normalized-Performance-Per-Watt (NPPW) [4], or minimize energy given a fixed tail cutoff for response time [10]. Other ideas for minimizing delay and energy involve utilizing sleep states in servers, which require more power than being off, but have a lower setup time [8, 9].

4 Key Techniques

In this section, I discuss the general method used to derive our main results, the upper and lower bounds on $\mathbb{E}[T_Q]$. The method I have developed can be broken into three steps:

1. First, we find a useful renewal cycle and applies the Renewal Reward Theorem, which states that, for example,

$$\mathbb{E}[Q(\infty)] = \frac{\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right]}{\mathbb{E}[X]} = \frac{\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right]}{\mathbb{E}\left[\int_0^X 1\mathrm{d}t\right]},$$

where by convention this renewal cycle begins at time 0 and ends at time X. Note that from here, to prove a bound on $\mathbb{E}[Q(\infty)]$, it suffices to provide the numerator and denominator separately, which can both be framed as time integrals up to a stopping time.

2. Second, we decompose the renewal period $[0, X)$ into further sequences of stopping times, leaving intervals of the form $[\tau, \gamma)$ for stopping times τ and γ.

3. Third, we use coupling to take the problem of bounding $\mathbb{E}\left[\int_\tau^\gamma Q(t)\mathrm{d}t\right]$ and reduce that bounding problem to a more basic probabilistic problem.

With the steps stated, I describe each step in turn.

4.1 The First Step: Applying the Renewal Reward Theorem

The first step in this method, applying the Renewal Reward Theorem, is not in-and-of-itself very difficult, but should nevertheless not be taken lightly. The choice of renewal cycle here has a very large impact on the difficulty of the next two steps. As far as I can see, our method works best for renewal cycles whose lengths are both not too long and not too short. Because the stopping time decomposition in the next step is essentially conditioning on what happens during the cycle, renewal cycles which are too long require very complex reasoning about all the different possible trajectories of the system over a long timescale. For example, it would be hard to reason about the behavior of the system in between moments where the system empties, i.e. the number of jobs $N(t)$ becomes 0. On the other hand, if the renewal cycle is too short, then it will be hard to deduce anything about what has happened in between renewal points. For example, if we set our renewal points to be the moments where the queue is empty and the number of busy servers $Z(t) = R$, then we could both 1) have a job arrival and a job departure in rapid succession, and 2) have a departure next, turning off the R-th server and thus requiring at least a full $\frac{1}{\alpha}$ before the renewal cycle is over. In my work, we set the renewal point $X \triangleq \min\{t > 0 : Z(t^-) = R + 1, Z(t) = R\}$ to be the first moment that $(R + 1)$-th server turns off. This turns out to be a happy medium, where the system renews consistently and the dynamic behavior between renewals varies only slightly. We show an idealized depiction of one of our renewal cycles in Figure 6.

4.2 The Second Step: Stopping Time Decomposition

The second step in this method is stopping time decomposition, which one might also call conditioning across time. In this step, we decompose the interval $[0, X)$ into a sequence of intervals $\mathcal{I}_i$, with the goal of deriving an accurate understanding of the behavior of the system within each interval. This decomposition happens along certain critical moments during a renewal cycle, which

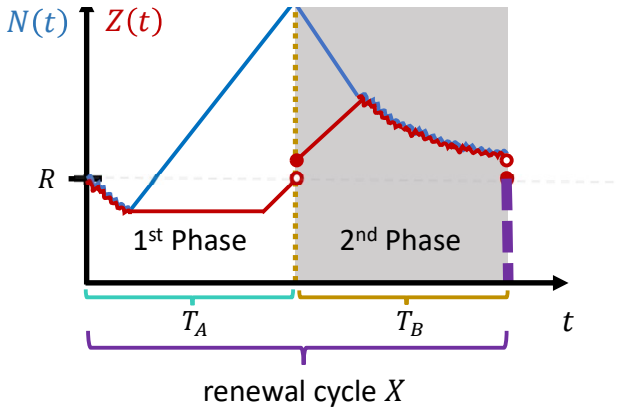

Figure 6: Idealized depiction of an M/M/k/Setup-Deterministic renewal cycle.

we usually define by entrance into or exit out of a particular region of state space. For example, since the end of a cycle X is marked by the shutoff of the $(R+1)$-th server, and the $(R+1)$-th server is off at time 0, a natural critical moment is the moment the $(R+1)$-th server turns on. This point, which we call T_A, represents a critical phase transition in the dynamics of the system: before time T_A, the departure rate of the system is at most the arrival rate, but after time T_A, the departure rate is strictly larger than the arrival rate. In other words, before T_A, jobs accumulate, and after T_A, they drain. This observation turns out to be incredibly useful in deriving these bounds.

First criterion. Before discussing the third step, I will describe in more detail how one should choose these "critical moments" by which we decompose the renewal cycle. In my work, there are two broad criteria by which I select stopping times. First, we want stopping times that imply something about future dynamics, i.e. the behavior of $Z(t)$, the number of busy servers. For example, in both bounds we define

$$\tau_j \triangleq \min\{t > 0 : N(t) = R - j\}.$$

Since we begin with $Z(t) = R$ busy servers, then at time τ_j we know that the number of busy servers $Z(t) = N(t) = R - j$, i.e., we know the exact state at time τ_j. Thus, the definition of τ_j immediately gives us two facts:

1. For the next $\frac{1}{\alpha}$ after time τ_j, the number of busy servers $Z(t) \leq R - j$.
2. Until time τ_{j+1}, the number of busy servers $Z(t) \geq R - j$.

These two statements turn out to be key in establishing our bounds.

Second criterion. The second criterion is that the number of these stopping times which happen before a renewal point X should be bounded in some way.

In our case, we define stopping times based on the moments when the system enters a "bad" region of state space, e.g., when the number of jobs $N(t)$ grows larger than some fixed boundary $R + M$. By tuning the size of this boundary appropriately, we can both 1) bound the number of visits to this "bad" region, and 2) bound the time integral while within this bad region. For example, in our proof of the upper bound on $\mathbb{E}[T_Q]$, we call the interval from $[\tau_j, \tau_{j+1} \wedge T_A)$ the j-th epoch, and we further break each epoch into a set of up-crossings followed by down-crossings, and show that the number of up-crossings is stochastically dominated by a certain Geometric random variable. Optimally, we can argue jointly about the dynamics after entering a region of state space and the integral while spent in that region.

4.3 The Third Step: Bounding

In the second step, we decomposed time into small intervals for which one has a tight bound on the dynamics; in the third and final step, we explicitly translate those dynamics-bounds into integral-bounds using, in our case, a combination of coupling arguments and martingale theory. If the previous steps have gone well, then all the reduced problems at this point are very basic probability problems.

First example: hitting probability. For example, in both my upper bound and lower bound of $\mathbb{E}[T_Q]$, I consider the following problem:

Problem 4.1. *Suppose that epoch j, which begins at time τ_j, ends when either the next epoch starts (at τ_{j+1}) or the $(R+1)$-th server turns on (at the stopping time we call T_A). What is the probability that epoch j lasts longer than a setup time?*

After a bit of work, we can reduce this problem to:

Problem 4.2. *Suppose we have a biased discrete random walk $V(x)$ which begins at $V(0) = 1$ and in each step $V(x)$ moves upward with probability p and downward with probability $q \leq p$. What is the probability that we reach $V(x) = 0$ in T steps?*

which can be solved in a variety of ways (combinatorics, the Optional Stopping Theorem, etc.).

4.3.1 Second example: escape probability

As another example, when proving that a number of up-crossings is stochastically dominated by a Geometric random variable, we encounter the following problem:

Problem 4.3. *Given a constant c, suppose that at some time u, the number of jobs $N(u) = R + c\sigma$, where $\sigma = \sqrt{\frac{k\lambda}{\alpha}}$, and the number of busy servers $Z(t) \leq R$. Give an upper bound on the probability that $N(t)$ becomes $\leq R$ within the next $\frac{1}{\alpha}$ time.*

From a coupling argument, this reduces to

Problem 4.4. *Given two independent Poisson processes $Y_A(t)$ and $Y_D(t)$ of rate $k\lambda$, give an upper bound on* $\Pr\left(\sup_{t\in[0,\frac{1}{\alpha}]} Y_A(t) - Y_D(t) \geq c\sigma\right)$.

which can also be solved in a number of ways (combinatorics, the Optional Stopping Theorem, the Berry-Esseen bound, etc). By solving these basic problems, we obtain small facts which, when assembled in the right way, give the desired bound.

5 Lower Bound

Section 5 focuses on the following lower bound for $\mathbb{E}[T_Q]$ in an M/M/k/Setup-deterministic system.

I first state the theorem, then describe the approach I have taken to prove it, then give an update on my progress towards proving it.

Theorem 2 (Explicit Lower Bound)**.** *Consider an M/M/k/Setup-Deterministic system with load $\rho = \frac{\lambda}{\mu}$ and setup time $\frac{1}{\alpha}$. If the ratio between the setup time and the service time satisfies that $\frac{1/\alpha}{1/\mu} \geq 1000$, the offered load $R = k\rho \geq 128$, and $\frac{1/\alpha}{1/\mu} \geq \log^2(k\rho)$, then the expected delay in steady state is lower bounded as*

$$\mathbb{E}[T_Q] \geq \frac{1}{k\lambda} \frac{\frac{1}{2}\left(\frac{1}{\alpha}\right)^2 \frac{\mu\sqrt{R}}{2} + g\left(\left[\left(\frac{\mu}{\alpha}-1\right)\frac{\sqrt{R}}{2} - k(1-\rho)\right]^+\right)}{C_1\left(\frac{1}{\alpha}+\frac{1}{\mu}\right) + \frac{1}{\alpha} + h\left(C_2\frac{\mu\sqrt{k\rho}}{\alpha}\right) + \frac{C_3}{\mu}\log\left(C_2\frac{\mu\sqrt{k\rho}}{\alpha}\right)},$$

where C_1, C_2, and C_3 are constants independent of system parameters k, λ, μ, and α, and $g(x) \triangleq \left[\frac{x-1}{2} + \frac{1}{1-\rho}\right]\frac{x}{k\mu(1-\rho)}$.

5.1 Approach

To prove this theorem, I follow the procedure outlined in Section 4. Let $X \triangleq \min\{t > 0 : Z(t^-) = R+1, Z(t) = R\}$ refer to the next renewal point. Applying the Renewal Reward Theorem, we obtain

$$\mathbb{E}[Q(\infty)] = \frac{\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right]}{\mathbb{E}[X]}.$$

Our two main lemmas will be a lower bound on the numerator and an upper bound on the denominator. We discuss each bound separately.

5.1.1 Lower bound on $\mathbb{E}\left[\int_0^X Q(t)\mathbf{d}t\right]$

To prove the lower bound on the numerator, we first break the initial portion of a renewal cycle into epochs; see Figure 7 for a visual reference. Recall that we

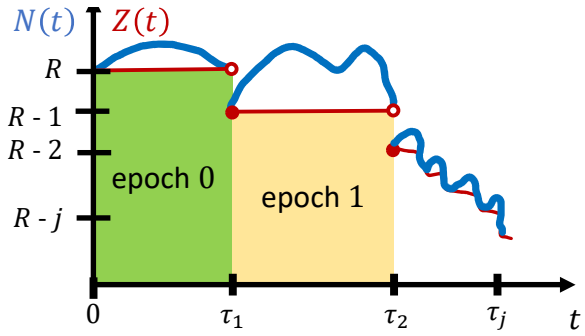

Figure 7: A depiction of a few epochs of the first phase.

define $\tau_j \triangleq \min\{t > 0 : N(t) = R - j\}$ as the first time the number of jobs falls to $R - j$, define T_A as the first time the $(R+1)$-th server turns on, and call the period $[\tau_j, \tau_{j+1} \wedge T_A)$ the j-th epoch. Call an epoch long if its length is longer than a setup time, and let L be the index of the first long epoch. Note that

1. An epoch j occurs during this renewal cycle if and only if $\tau_j < T_A$,
2. The end of the first phase T_A can not occur until a long epoch has happened (since no servers can set up during a short epoch), and furthermore must occur after $\tau_L + \frac{1}{\alpha}$.
3. The moment where epoch L is identified, $\tau_L + \frac{1}{\alpha}$, is a stopping time.

It follows that

$$\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right] = \mathbb{E}\left[\int_0^{\tau_L+\frac{1}{\alpha}} Q(t)\mathrm{d}t\right] + \mathbb{E}\left[\int_{\tau_L+\frac{1}{\alpha}}^X Q(t)\mathrm{d}t\right]. \tag{2}$$

We analyze each of the terms in the right hand side of (2) separately.

Bound on the first term. To bound the first term in (2), we use the observation that the departure rate is fixed at $\mu(R-j)$ at the beginning of epoch j together with the Optional Stopping Theorem to show that

$$\mathbb{E}\left[\int_0^{\tau_L+\frac{1}{\alpha}} Q(t)\mathrm{d}t\right] \geq \frac{\mu}{\alpha}\mathbb{E}[L].$$

To bound $\mathbb{E}[L]$, we directly bound the probability that epoch j lasts longer than a setup time; we reduce this to a problem about biased random walks, as discussed in Problem 4.2. In the end, we get a bound which says

$$\mathbb{E}[L] \geq \frac{1}{2}\sqrt{R}, \tag{3}$$

which completes our bound of the first term.

Bound on the second term. To bound the second term in (2), we couple the original system to an M/M/1 with arrival rate $k\lambda$ and departure rate $k\mu$. In particular, define the coupled number of jobs in queue as

$$\hat{Q}(t) = A(\tau_L + \frac{1}{\alpha}, t) - D_k(\tau_L + \frac{1}{\alpha}, t) + \left[N(\tau_L + \frac{1}{\alpha}) - k\right]^+.$$

Let $\text{BP} \triangleq \min\left\{t > \tau_L + \frac{1}{\alpha} : \hat{Q} \leq 0\right\}$ be the next time the coupled queue is empty. Since $\hat{Q}(t) \leq Q(t)$ for all $t \in [\tau_L + \frac{1}{\alpha}, \text{BP}]$, it follows that

$$\text{BP} < X$$

and, thus that

$$\int_{\left(\tau_L + \frac{1}{\alpha}\right)}^{X} Q(t)\mathrm{d}t \geq \int_{\left(\tau_L + \frac{1}{\alpha}\right)}^{\text{BP}} Q(t)\mathrm{d}t,$$

where one should note that the expected value of the right hand side is simply the expected time integral of an M/M/1 with arrival rate $k\lambda$ and departure rate $k\mu$ over a busy period started by $[N\left(\tau_L + \frac{1}{\alpha}\right) - k]^+$, which is known classically. For convenience, we write this value as $g(x) \triangleq \left[\frac{x-1}{2} + \frac{1}{1-\rho}\right] \frac{x}{k\mu(1-\rho)}$. Applying Jensen's inequality, we find

$$\begin{aligned}
\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right] &= \mathbb{E}\left[\int_0^{\tau_L + \frac{1}{\alpha}} Q(t)\mathrm{d}t\right] + \mathbb{E}\left[\int_{\tau_L + \frac{1}{\alpha}}^{X} Q(t)\mathrm{d}t\right] \\
&\geq \frac{\mu}{\alpha}\mathbb{E}\left[L\right] + \mathbb{E}\left[g\left(\left[N\left(\tau_L + \frac{1}{\alpha}\right) - k\right]^+\right)\right] \\
&\geq \frac{\mu}{\alpha}\mathbb{E}\left[L\right] + g\left(\left[\mathbb{E}\left[N\left(\tau_L + \frac{1}{\alpha}\right) - R\right] - k(1-\rho)\right]^+\right).
\end{aligned}$$

To bound $\mathbb{E}\left[N\left(\tau_L + \frac{1}{\alpha}\right) - R\right]$, we again make a martingale argument, which shows that

$$\mathbb{E}\left[N\left(\tau_L + \frac{1}{\alpha}\right) - R\right] \geq \left(\frac{\mu}{\alpha} - 1\right)\mathbb{E}\left[L\right].$$

Using the bound on $\mathbb{E}\left[L\right]$ from (3), we obtain that

$$\mathbb{E}\left[\int_0^X Q(t)\mathrm{d}t\right] = \frac{1}{2}\left(\frac{1}{\alpha}\right)^2 \frac{\mu\sqrt{R}}{2} + g\left(\left[\left(\frac{\mu}{\alpha} - 1\right)\frac{\sqrt{R}}{2} - k(1-\rho)\right]^+\right),$$

as desired.

5.1.2 Upper bound on $\mathbb{E}\left[X\right]$

To upper bound the expected length of a renewal cycle $\mathbb{E}\left[X\right]$, we analyze the two phases of the system separately, i.e. we bound $\mathbb{E}\left[T_A\right]$ and $\mathbb{E}\left[T_B\right] \triangleq \mathbb{E}\left[X - T_A\right]$ separately.

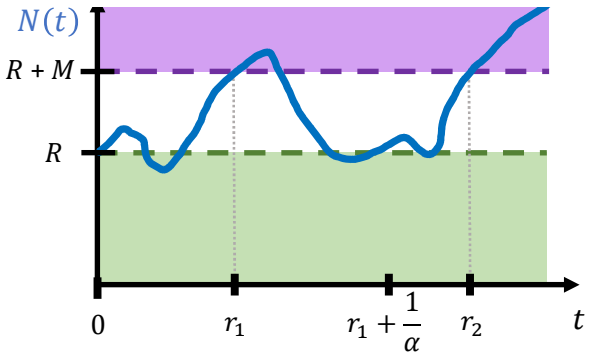

Figure 8: A depiction of the first few "separated visits" to the region $N(t) \geq R + M$.

Bound on $\mathbb{E}[T_A]$. To bound $\mathbb{E}[T_A]$, we make an up-crossing argument; for a visual representation, see Figure 8. For our up-crossings, we define a sequence of separated visits to the region $N(t) \geq R + M$, where

$$r_1 \triangleq \min\{t > 0 : N(t) \geq R + M\}$$

and

$$r_{i+1} \triangleq \min\left\{t > r_i + \frac{1}{\alpha} : N(t) \geq R + M\right\}.$$

Let n_r be the random number of these separated visits which occur before time T_A. From Wald's equation, we see that

$$\begin{aligned}
\mathbb{E}[T_A] &= \mathbb{E}\left[T_A - r_{n_r} + \sum_{i=0}^{n_r-1} r_{i+1} - r_i\right] \\
&= \mathbb{E}\left[\sum_{i=0}^{n_r} r_{i+1} \wedge T_A - r_i\right] \\
&= \sum_{i=0}^{\infty} \Pr(n_r \geq i)\, \mathbb{E}[r_{i+1} \wedge T_A - r_i | n_r \geq i].
\end{aligned}$$

We then show that, if $M \approx \sqrt{\frac{k\lambda}{\alpha}}$, then the number of visits n_r is stochastically dominated by a Geometric random variable, and the time between visits is bounded by

$$\mathbb{E}[r_{i+1} \wedge T_A - r_i | n_r \geq i] \leq 2\left(\frac{1}{\alpha} + \frac{1}{\mu}\right)$$

This, in turn, tells us that

$$\mathbb{E}[T_A] = \sum_{i=0}^{\infty} \Pr(n_r \geq i)\, \mathbb{E}[r_{i+1} \wedge T_A - r_i | n_r \geq i]$$

$$\leq 2\left(\frac{1}{\alpha}+\frac{1}{\mu}\right)\mathbb{E}\left[n_r\right]$$
$$\leq C_1\left(\frac{1}{\alpha}+\frac{1}{\mu}\right),$$

as desired.

Bound on $\mathbb{E}[T_B]$. To bound $\mathbb{E}[T_B] = \mathbb{E}[X - T_A]$, we again make an M/M/1 coupling argument, except that here we couple the original system to something we call the shutoff system, where the dynamics of the two systems are the same, except that shutoff system does not turn on servers. Specifically, we let the system turn on and off servers for $\frac{1}{\alpha}$ after time T_A, and then, at time $T_A + \frac{1}{\alpha}$, we migrate our jobs and busy servers to the shutoff system and allow it to run until $N_{\text{SHUT}}(t) = R$. After analyzing the shutoff system as a sequence of M/M/1 queues, we obtain that

$$\mathbb{E}[X - T_A] \leq \frac{1}{\alpha} + \frac{\mathbb{E}[Q(T_A)]}{k\mu(1-\rho)} + \frac{C_1}{\mu}\log\left(\mathbb{E}[Q(T_A)]\right)$$

To finish our proof, we bound $\mathbb{E}[Q(T_A)]$; to do so, it suffices to bound the conditional expectation after an up-crossing. We do so using martingales and the notion of epochs and up-crossings, an idea explained more thoroughly in Section 6.

5.2 Status

This part of the project is essentially 100% complete, and a paper on the subject matter has been published in SIGMETRICS 2023. As I've gained insight into the problem more, there are a few pieces I would like to rewrite for clarity/cohesiveness and a few places I think the analysis can easily be tightened, but, aside from that, I am done with it.

6 Upper Bound

In Section 6, I discuss the ideas that go into proving the upper bound, and give an update on this part of the project's status. First, the theorem statement:

Theorem 3. *Consider an M/M/k/Setup-Deterministic system with load $\rho = \frac{\lambda}{\mu}$ and setup time $\frac{1}{\alpha}$. If the ratio between the setup time and the service time satisfies that $\frac{1/\alpha}{1/\mu} \geq 1000$, the offered load $R = k\rho \geq 128$, and $\frac{1/\alpha}{1/\mu} \geq \log^2(k\rho)$, then the expected delay in steady state is upper bounded as*

$$\mathbb{E}[T_Q] \leq \frac{1}{k\lambda}\left[\frac{\left(B_1\sqrt{\frac{k\lambda}{\alpha}} + B_2\frac{\mu\sqrt{R}}{\alpha}\right)\frac{1}{\alpha} + \frac{\left(B_3\sqrt{\frac{k\lambda}{\alpha}} + B_4\frac{\mu}{\alpha}\sqrt{R}\right)^2}{\mu k(1-\rho)} + \frac{1}{\alpha}\frac{\rho}{1-\rho}}{\frac{1}{\alpha} + \frac{A_1\frac{\mu\sqrt{R}}{\alpha}}{k\mu(1-\rho)}} + B_5\sqrt{\frac{k\lambda}{\alpha}}\right],$$

where all the B_i's and A_1 are constants independent of system parameters k, λ, μ, and α.

6.1 Approach

Following the method outlined in Section 4, we reduce the problem of upper bounding $\mathbb{E}[T_Q]$ to upper bounding $\mathbb{E}\left[\int_0^X [N(t) - R]\mathrm{d}t\right]$ and lower bounding $\mathbb{E}[X]$; note the change in integrand made here, compared to the usualy $Q(t)$. We address the two bounds separately, in some places recalling ideas from the lower bound of Section 5.

6.1.1 Upper bound on $\mathbb{E}\left[\int_0^X [N(t) - R]\mathbf{d}t\right]$

We again split our analysis into two phases. Recall that

$$T_A \triangleq \min\{t > 0 : Z(t) = R + 1\},$$

and call the interval from $[0, T_A)$ the first phase, and $[T_A, X)$ the second phase, as in Figure 6.

Bound on $\mathbb{E}\left[\int_0^{T_A} [N(t) - R]\mathbf{d}t\right]$. To bound this integral, we again split the first phase up into epochs, but now we further decompose the epochs into up-crossings and down-crossings; see Figure 9 for details. Here, the zeroth down-crossing is at time τ_j, the first up-crossing is at time

$$u_1^{(j)} \triangleq \min\{t > \tau_j : N(t) \geq R + M\},$$

and, in general, the i-th down-crossing occurs at time

$$d_i^{(j)} \triangleq \min\left\{t \geq u_i^{(j)} : N(t) \leq R\right\}.$$

Likewise, the $(i+1)$-th up-crossing occurs at

$$u_{i+1}^{(j)} \triangleq \min\left\{t \geq d_i^{(j)} : N(t) \geq R + M\right\}.$$

Furthermore, we call the interval $[u_i^{(j)}, d_i^(j))$ the i-th fall in epoch j and the interval $[d_i^{(j)}, u_{i+1}^j)$ the $(i+1)$-th rise in epoch j, let n_e be the number of epochs before time T_A, and also let $n_u^{(j)}$ be the number of up-crossings which occur during epoch j.

We separate our analysis of rises and falls. During a rise, by definition, we have the bound $N(t) - R \leq M$. It follows that the time integral over the i-th rise is always $\leq M[u_i^{(j)} \wedge T_A - d_i^{(j)}]$. Likewise, during epoch j, we have a lower bound on the number of departure rate of jobs ($\mu(R - j)$), which, via a simple

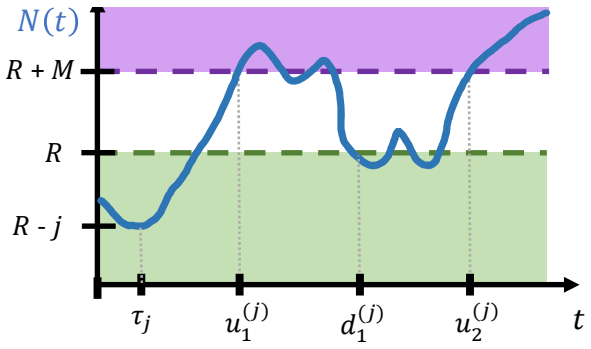

Figure 9: A depiction of the first few up-crossings and down-crossings in epoch j.

coupling + martingales argument, gives way to a simple bound on the time integral over a fall

$$\mathbb{E}\left[\int_{u_i^{(j)}}^{T_A \wedge d_i^{(j)}} [N(t) - R]\mathrm{d}t \middle| n_u^{(j)} \geq i\right] \leq 2M\mathbb{E}\left[T_A \wedge d_i^{(j)} - u_i^{(j)} \middle| n_u^{(j)} \geq i\right] + \frac{1}{2}\left(\frac{1}{\alpha}\right)^2 \mu j,$$

where here we have set $M = \sqrt{\frac{k\lambda}{\alpha}}$. Applying Wald's equation and performing some algebra, we obtain

$$\mathbb{E}\left[\int_0^{T_A} [N(t) - R]\mathrm{d}t\right] \leq 2M\mathbb{E}\left[T_A\right] + \frac{1}{2}\left(\frac{1}{\alpha}\right)^2 \mu \sum_{j=0}^{\infty} \Pr\left(n_e \geq j\right) j.$$

We bound this final summation by proving that, for $j > \sqrt{2}\sqrt{R}$,

$$\Pr\left(n_e = j | n_e \geq j\right) \geq \frac{\sqrt{2}}{\sqrt{R}},$$

i.e. by lower bounding the probability that the first phase ends before the next epoch begins.

Bound on $\mathbb{E}\left[\int_{T_A}^{X} [N(t) - R]\mathbf{d}t\right]$. To bound this integral, we again make use of the separated visits idea from Section 5, except that now we are interested in down-crossings, letting our first visit occur at time

$$v_1 \triangleq \min\left\{t \geq T_A : N(t) \leq R + M_2\right\}$$

and letting successive separated visits occur at

$$v_{i+1} \triangleq \min\left\{t \geq v_i + \frac{1}{\alpha} : N(t) \leq R + M_2\right\};$$

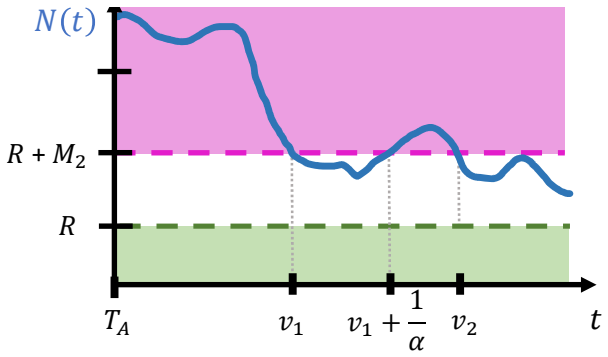

Figure 10: A depiction of the first few visits to the region $\{N(t) \leq R + M_2\}$.

see Figure 10 for more details. For convenience, let n_v be the number of visits which occur before time X.

Using the same idea of allowing the system to turn on servers for a $\frac{1}{\alpha}$ period, then converting our system into a SHUTOFF system (a system where servers can only turn off; see Section 5 for more details), we can obtain bounds on the integral until the first visit $\mathbb{E}\left[\int_{T_A}^{v_1}\right]$, the integral between successive visits $\mathbb{E}\left[\int_{v_i}^{v_{i+1}}\right]$, and the probability $\Pr(n_v \geq i)$. We omit the exact statement of these bounds, due to their complexity. Combining these bounds using Wald's equation, we find that

$$\mathbb{E}\left[\int_{T_A}^{X}[N(t)-R]\mathrm{d}t\right] \leq \left(B_6\sqrt{\frac{k\lambda}{\alpha}} + B_7\frac{\mu\sqrt{R}}{\alpha}\right)\frac{1}{\alpha} + \frac{\left(B_3\sqrt{\frac{k\lambda}{\alpha}} + B_4\frac{\mu}{\alpha}\sqrt{R}\right)^2}{\mu k(1-\rho)} + \frac{1}{\alpha}\frac{\rho}{1-\rho},$$

as desired.

6.1.2 Lower bound on $\mathbb{E}[X]$

To prove a lower bound on the cycle length, we again make use of the "long" epoch idea, where an epoch is long if it lasts longer than $\frac{1}{\alpha}$ time, and L is the index of the first long epoch. Since at least one epoch must be long, we have $\mathbb{E}\left[\tau_L + \frac{1}{\alpha}\right] \geq \frac{1}{\alpha}$. Moreover, by coupling the system to an M/M/1 with arrival rate $k\lambda$ and departure rate $k\mu$ at time $\tau_L + \frac{1}{\alpha}$, we obtain that

$$\begin{aligned}\mathbb{E}[X] &= \mathbb{E}\left[\tau_L + \frac{1}{\alpha}\right] + \mathbb{E}\left[X - \left(\tau_L + \frac{1}{\alpha}\right)\right] \\ &\geq \frac{1}{\alpha} + \frac{\mathbb{E}\left[N\left(\tau_L + \frac{1}{\alpha}\right) - R\right]}{k\mu(1-\rho)}.\end{aligned}$$

Just as in Section 5, applying a stopped martingale analysis, we find that

$$\mathbb{E}\left[N\left(\tau_L + \frac{1}{\alpha}\right) - R\right] \geq \frac{\mu}{\alpha}\mathbb{E}\left[L\right] \geq \frac{1}{2}\frac{\mu}{\alpha}\sqrt{R} \triangleq A_1\frac{\mu}{\alpha}\sqrt{R},$$

which where we once again bound $\mathbb{E}\left[L\right]$ by lower bounding the probability that an epoch lasts $\frac{1}{\alpha}$ time.

6.2 Status

I would estimate this project is around 90% done: a very rough draft of the technical results has been written, and my goal is to prepare and submit a manuscript to either Performance or Operations Research (OR) within the next few months.

7 Future Work

In this section, we describe a few possible next steps for this project.

7.1 Analyzing the Gap (in mean delay between the Exponential and Deterministic models)

Perhaps the clearest next step to take is to attempt to prove that there's a large gap between the Exponential setup model and the Deterministic setup model.There are two reasons why I find this direction compelling.

The first reason is that showing that this gap is large would provide a strong theoretical justification that the setup distribution can not be ignored when modeling multiserver systems with setup. In fact, looking at our experimental results (Figure 2), there is a natural conjecture:

Conjecture 7.1. *As the number of servers k and the setup time $\frac{1}{\alpha}$ both scale to ∞, the limit*

$$\frac{\mathbb{E}\left[T_Q^{(Exp)}\right]}{\mathbb{E}\left[T_Q^{(Det)}\right]} \to 0.$$

Although Conjecture 7.1 seems to be the case, at the moment, there are no closed form bounds for $\mathbb{E}\left[T_Q^{(\text{Exp})}\right]$. The state-of-the-art when it comes to the analysis of the M/M/k/Setup-Exponential are computational methods [11, 19] and limit theorems [18]; to date, no one has produced a provably correct formula bounding $\mathbb{E}\left[T_Q^{(\text{Exp})}\right]$ which captures its scaling behavior with the setup time $\frac{1}{\alpha}$.

Beyond cementing the separation between Deterministic and Exponential setup systems, analyzing the Exponential system would also present unique challenges to the methods I have developed to analyze the Deterministic system. In the Deterministic system, we have almost sure lower bounds on when a

server will next turn on, and questions about the system dynamics can easily be phrased in terms of straightforward stopping times. In the Exponential system, the moment-to-moment dynamics of the system end up affecting the time until setup occurs; how should we modify our method to account for this? At the moment, the modification we should make is unclear, but, once understood, could pave the way to making this method much more broadly useful.

In particular, I am excited to find out how to blend existing steady-state analysis techniques with my method which focuses on integration. For example, state-space collapse results show that a system with a high-dimensional state space has a steady state distribution with much fewer dimensions, e.g. a dispatching system using a Join-the-Shortest-Queue assignment discipline usually has very balanced queues. Is there a nice state-space collapse result for the Exponential setup system which we can use to help us bound the expected time integrals that my method produces? Answering questions like these would clarify the strengths and potential synergies between the method I introduced here and others.

7.2 Analyzing Other Policies

Given that we have demonstrated that setup times can have a large negative effect on queueing, another compelling direction is to show that there exists a policy which can mitigate that large negative effect, and, if possible, characterizing the delicate balance that exists between the properties of performance, robustness, and efficiency. There are a number of possible policies to analyze. One example is the Delayed-Off policy, a policy where the system behaves in the same way as the M/M/k/Setup, except that idle servers are now turned off at some rate β instead of instantly turning off. This parameter β provides a tunable interpolation between the usual M/M/k/Setup policy and the policy which keeps all servers on all the time (when $\beta \to \infty$ and $\beta \to 0$, respectively). It would be interesting to deduce a systematic way of setting β to ensure a system achieves its desired quality of service.

8 Timeline

In this section, I lay out a timeline for completing my thesis work.

May-July 2023:

- Prepare a submission for Performance or OR on the Upper Bound (Section 6).
- Write thesis chapter on the Lower Bound (Section 5).
- Begin work on Analyzing the Gap (Section 7.1).

August-October 2023:

- Begin work on Analyzing Other Policies (Section 7.2).
- Write thesis chapter on Impact (Section 1.7).
- Prepare a submission for the SIGMETRICS Fall deadline on Analyzing the Gap (Section7.1).

November 2023-January 2024:

- Write thesis chapter on the Upper Bound (Section 6).
- Write thesis chapter on Analyzing the Gap (Section 7.1.
- Prepare a submission for the SIGMETRICS Winter deadline on Analyzing Other Policies (Section 7.2).

February-April 2024

- Write thesis chapter on Analyzing Other Policies (Section 7.2).
- Finish writing thesis and complete outstanding projects.

References

[1] J. R. Artalejo, A. Economou, and M. J. Lopez-Herrero. Analysis of a Multiserver Queue with Setup Times. *Queueing Syst.*, 51(1):53–76, 2005.

[2] W. Bischof. Analysis of M/G/1-Queues with Setup Times and Vacations under Six Different Service Disciplines. *Queueing Syst.*, 39(4):265–301, 2001.

[3] G. Choudhury. On a batch arrival Poisson queue with a random setup time and vacation period. *Comp. & Oper. Res.*, 25(12):1013–1026, 1998.

[4] A. Gandhi and M. Harchol-Balter. How Data Center Size Impacts the Effectiveness of Dynamic Power Management. In *Proc. Ann. Allerton Conf. Communication, Control and Computing*, pages 1164–1169, Urbana-Champaign, IL, September 2011.

[5] A. Gandhi and M. Harchol-Balter. M/G/k with staggered setup. *Oper. Res. Lett.*, 41(4):317–320, 2013.

[6] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. Kozuch. Optimality analysis of energy-performance trade-off for server farm management. In *Proc. Int. Symp. Computer Performance, Modeling, Measurements and Evaluation (IFIP Performance)*, Namur, Belgium, November 2010.

[7] A. Gandhi, M. Harchol-Balter, and I. Adan. Server farms with setup costs. *Performance Evaluation*, 67(11):1123–1138, 2010.

[8] A. Gandhi, M. Harchol-Balter, and M. Kozuch. The case for sleep states in servers. In *SOSP Workshop on Power-Aware Computing and Systems (HotPower)*, pages 1–5, Cascais, Portugal, October 2011.

[9] A. Gandhi, M. Harchol-Balter, and M. Kozuch. Are sleep states effective in data centers? In *Int. Conf. Green Computing (IGCC)*, pages 1–10, San Jose, CA, 2012.

[10] A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M. A. Kozuch. AutoScale: Dynamic, Robust Capacity Management for Multi-Tier Data Centers. *ACM Trans. Comput. Syst.*, 30(4):1–26, 2012.

[11] A. Gandhi, S. Doroudi, M. Harchol-Balter, and A. Scheller-Wolf. Exact analysis of the M/M/k/setup class of Markov chains via Recursive Renewal Reward. In *Queueing Syst.*, pages 153–166, 2013.

[12] Q.-M. He and E. Jewkes. Flow time in the MAP/G/1 queue with customer batching and setup times. *Stochastic Models*, 11(4):691–711, 1995.

[13] E. Hyytiä, D. Down, P. Lassila, and S. Aalto. Dynamic Control of Running Servers. In *Int. Conf. Measurement, Modelling and Evaluation of Comput. Systems*, pages 127–141, Erlangen, Germany, 2018. Springer.

[14] A. Kara. Energy Consumption in Data Centers with Deterministic Setup Times. Master's thesis, Middle East Technical University, 2017.

[15] M. Mao and M. Humphrey. A Performance Study on the VM Startup Time in the Cloud. In *IEEE Int. Conf. Cloud Computing (CLOUD)*, pages 423–430, Honolulu, HI, 2012.

[16] D. Mukherjee and A. Stolyar. Join Idle Queue with Service Elasticity: Large-Scale Asymptotics of a Nonmonotone System. *Stoch. Syst.*, 9(4): 338–358, 2019.

[17] D. Mukherjee, S. Dhara, S. C. Borst, and J. S. van Leeuwaarden. Optimal Service Elasticity in Large-Scale Distributed Systems. *Proc. ACM SIGMETRICS Int. Conf. Measurement and Modeling of Computer Systems*, 1: 1–28, 2017.

[18] J. Pender and T. Phung-Duc. A law of large numbers for m/m/c/delayoff-setup queues with nonstationary arrivals. In *Int. Conf. on Analytical and Stochastic Modeling Techniques and Applications*, pages 253–268, Cardiff, UK, 2016. Springer.

[19] T. Phung-Duc. Exact solutions for M/M/c/setup queues. *Telecommun. Syst.*, 64(2):309–324, 2017.

[20] K. Rzadca, P. Findeisen, J. Swiderski, P. Zych, P. Broniek, J. Kusmierek, P. Nowak, B. Strack, P. Witusowski, S. Hand, et al. Autopilot: workload autoscaling at Google. In *Proc. European Conf. Computer Systems (EuroSys)*, pages 1–16, Heraklion, Crete, Greece, 2020.

[21] M. Tirmazi, A. Barker, N. Deng, M. E. Haque, Z. G. Qin, S. Hand, M. Harchol-Balter, and J. Wilkes. Borg: the next generation. In *Proc. European Conf. Computer Systems (EuroSys)*, pages 1–14, 2020.

[22] P. D. Welch. On a Generalized M/G/1 Queuing Process in Which the First Customer of Each Busy Period Receives Exceptional Service. *Oper. Res.*, 12(5):736–752, 1964.