Robustness and performance of threshold-based resource allocation policies

Takayuki Osogami   Mor Harchol-Balter   Alan Scheller-Wolf

Computer Science Department, Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15213, USA, {osogami,harchol}@cs.cmu.edu,
Tepper School of Business, Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA 15213, USA, awolf@andrew.cmu.edu.

Area of review: Manufacturing, Service, and Supply Chain Optimization.

Subject Classifications: Production/scheduling: Flexible manufacturing/line balancing. Queues: Markovian. Dynamic programming / optimal control: Markov: Infinite state.

Abstract

We provide the first analytical study of the mean response time and robustness of a wide range of threshold-based resource allocation policies for a multiserver queueing system such as those commonly used in modeling call centers. We introduce two different types of robustness: static robustness and dynamic robustness. Static robustness measures robustness against misestimation of load (i.e., constant load differing from that predicted), while dynamic robustness measures robustness against fluctuations in load (i.e., alternating high and low loads, or burstiness). We find that using multiple thresholds can have significant benefit over using only a single threshold with respect to static robustness, but that multiple thresholds surprisingly offer only small advantage with respect to dynamic robustness and mean response time. A careful evaluation of load conditions allows us to establish guidelines for choosing a good resource allocation policy, with respect to simplicity, robustness, and mean response time. Finally, we evaluate the effectiveness of our guidelines in designing resource allocation policies at a call center.

1 Introduction

A common problem in multiserver systems is deciding how to allocate resources (e.g. operators, CPU time, and bandwidth) among jobs to maximize system performance, e.g. with respect to mean response time or throughput. Since good parameter settings typically depend on environmental conditions such as system loads, an allocation policy that is optimal in one environment may provide poor performance when the environment changes, or when the estimation of the environment is wrong. In other words, the policy may not be *robust*. In this paper, we design several allocation policies for multiserver systems, quantifying their performance with respect to mean response time and robustness, providing insights into which types of policies perform well in different operating environments.

1.1 Model and metric

We consider a multiserver model that consists of two servers and two queues (Beneficiary-Donor model), as shown in Figure 1. Jobs arrive at queue 1 and queue 2 according to (possibly Markov modulated) Poisson processes with average arrival rates λ_1 and λ_2, respectively. Jobs have exponentially distributed service demands; however, the running time of a job may also depend on the

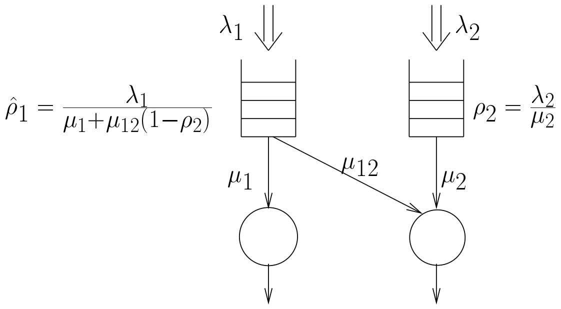


Figure 1: *Beneficiary-Donor model.*

affinity between the particular server and the particular job/queue. Hence, we assume that server 1 (beneficiary server) processes jobs in queue 1 (type 1 jobs) with rate μ_1, while server 2 (donor server) can process type 1 jobs with rate μ_{12}, and can process jobs in queue 2 (type 2 jobs) with rate μ_2. We define $\rho_1 = \lambda_1/\mu_1$, $\rho_2 = \lambda_2/\mu_2$, and $\hat{\rho}_1 = \lambda_1/(\mu_1 + \mu_{12}(1 - \rho_2))$. Note that $\rho_2 < 1$ and $\hat{\rho}_1 < 1$ are necessary for the queues to be stable under any allocation policy, since the maximum rate at which type 1 jobs can be processed is μ_1, from server 1, plus $\mu_{12}(1 - \rho_2)$, from server 2.

The Beneficiary-Donor model has a wide range of applications in service facilities such as call centers and repair facilities. For example, in call centers, the donor server may be a bilingual operator, and the beneficiary server may be a monolingual operator (Shumsky, 2004; Stanford and Grassmann, 1993, 2000), or the donor server may be a cross-trained or experienced generalist who can handle all types of calls, and the beneficiary server may be a specialized operator who is only trained to handle a specific type of calls (Shumsky, 2004). In a repair facility, the donor server may be a technician who can handle jobs of any difficulty, and the beneficiary server may be a technician with limited expertise (Green, 1985).

We design and evaluate allocation policies for the Beneficiary-Donor model with respect to three objectives. First, as is standard in the literature, we seek to minimize the overall weighted mean response time, $c_1 p_1 E[R_1] + c_2 p_2 E[R_2]$, where c_i is the weight (importance) of type i jobs, $p_i = \lambda_i/(\lambda_1 + \lambda_2)$ is the fraction of type i jobs, and $E[R_i]$ is the mean response time of type i jobs, for $i = 1, 2$. Here, response time refers to the total time a job spends in the system. Below, we refer to overall weighted mean response time simply as mean response time.

In addition to mean response time, we consider an additional metric, *robustness*, introducing two types of robustness: *static robustness* and *dynamic robustness*. Static robustness measures robustness against *misestimation* of load; to evaluate static robustness, we analyze the mean response time of allocation polices for a range of loads to see how a policy tuned for *one* load behaves under different loads. Dynamic robustness measures the robustness against *fluctuations* in load; to evaluate dynamic robustness, we analyze the mean response time of allocation policies under Markov modulated Poisson processes, where arrivals follow a Poisson process at each moment, but the arrival rate changes over time.

1.2 Prior work

There has been a large amount of prior work on the Beneficiary-Donor model, the majority of which focused on proving the optimality of allocation policies in limiting or special cases. With respect to calculating mean response times, only coarse approximations exist for most of the allocation policies in our model. We provide a nearly exact analysis of these, as well as other allocation policies, while also investigating static and dynamic robustness.

One common allocation policy is the $c\mu$ rule (Cox and Smith, 1971), which biases in favor of jobs with high c (high importance) and high μ (small expected size). Applying the cμ rule to our setting, server 2 serves type 1 jobs (rather than type 2 jobs) if $c_1\mu_{12} > c_2\mu_2$, or queue 2 is empty. The $c\mu$ rule is provably optimal when server 1 does not exist (Cox and Smith, 1971) or in the fluid limit (Meyn, 2001; Squillante et al., 2002). However Squillante et al. (2001) as well as Harrison (1998) have shown that $c\mu$ rule may lead to instability (queue length growing unboundedly) even if $\hat{\rho}_1 < 1$ and $\rho_2 < 1$. More recently, Mandelbaum and Stolyar (2004) and Van Mieghem (1995) have introduced and analyzed the *generalized $c\mu$ rule*. However, in our model, the generalized $c\mu$ rule reduces to the $c\mu$ rule and hence has the same stability issues.

In light of this instability, Squillante et al. (2001) and Williams (2000) independently proposed a threshold-based policy that, under the right choice of threshold value, improves upon the $c\mu$ rule with respect to mean response time, guaranteeing stability whenever $\hat{\rho}_1 < 1$ and $\rho_2 < 1$. We refer to this threshold-based policy as the T1 policy, since it places a threshold value, t_1, on queue 1, so that server 2 processes type 1 jobs only when there are at least t_1 jobs of type 1, or if queue 2 is empty. The rest of the time server 2 works on type 2 jobs. This "reserves" a certain amount of work for server 1, preventing server 1 from being under-utilized and server 2 from becoming overloaded, as can happen under the $c\mu$ rule. Bell and Williams (2001) prove the optimality of the T1 policy for a model closely related to ours in the heavy traffic limit.

However, studies by Meyn (2001) and Ahn et al. (2004) suggest that the T1 policy is *not* optimal in general. Meyn obtains, via a numerical approach, the optimal allocation policy when both queues have finite buffers. Although not proven, the optimal policy appears to be a "flexible" T1 policy that allows a continuum of T1 thresholds, $\{t_1^{(i)}\}$, where threshold $t_1^{(i)}$ is used when the length of queue 2 is i. Ahn et al. characterize the optimal policy with respect to minimizing the total holding cost until all the jobs in the system at time zero leave the system, assuming that there are no arrivals after time zero. They also find that the optimal policy is in general a "flexible" T1 policy.

All of the work above investigates a class of allocation policies that are optimal in limiting or special cases. In contrast, there has been little work on the *analysis and evaluation* of the mean response time of general allocation policies in our model, and no work evaluating robustness. Complicating this problem is the fact that the state space required to capture the system behavior grows infinitely in two dimensions; i.e., we need to track both the number of type 1 jobs and the number of type 2 jobs. Hence, only approximate analyses exist for most of allocation policies in our model. For example, Squillante et al. (2001) derive a coarse approximation for the mean response time of the T1 policy under Poisson arrivals based on vacation models. The mean response time of other simple allocation policies (in more general models) such as (idle) cycle stealing, where server 2 works on type 1 jobs when queue 2 is empty, have also been analyzed (with approximation) either by matrix analytic methods with state space truncation (Green, 1985; Stanford and Grassmann, 1993, 2000) or by approximate solutions of a 2D-infinite Markov chain via state space decomposition (Shumsky, 2004). Recently, we have introduced the first nearly exact analysis of the mean response time under a wide range of allocation policies for the Beneficiary-Donor model (Osogami et al., 2004). However, the analysis in (Osogami et al., 2004) is limited to Poisson arrivals.

1.3 Contributions of the paper

- In this paper, we extend the analysis in (Osogami et al., 2004) to more general arrival processes, which allows us to investigate static and dynamic robustness. Our analysis is based on the approach of dimensionality reduction, DR (see for example Osogami, 2005).

DR reduces a two dimensionally (2D) infinite Markov chain to a 1D-infinite Markov chain, which closely approximates the 2D-infinite Markov chain. In particular, DR allows us to evaluate the mean response time under the T1 policy, and a similar policy called the T2 policy which places a threshold on queue 2.

- We introduce two types of robustness: static robustness and dynamic robustness, and analytically study a wide range of threshold-based allocation policies with respect to both types of robustness. Surprisingly, we will see that policies that excel in static robustness do not necessarily excel in dynamic robustness.

- Specifically, we find that an allocation policy with multiple thresholds can experience significant benefit over allocation policies with a single threshold with respect to *static* robustness. Illustrating this, we introduce the adaptive dual threshold (ADT) policy, which places two thresholds on queue 1, and show this has significant advantage over single threshold allocation policies with respect to static robustness. The ADT policy operates like a T1 policy, but the threshold value is self-adapted to the load.

- In contrast to this, we find that multiple thresholds surprisingly offer only small advantage over a single threshold with respect to mean response time and *dynamic* robustness.

- We apply the principles learned to designing allocation policies for call centers: based on the characterization of a call center's operational data, we identify effective allocation policies. We then evaluate our recommended policies via trace driven simulation. Results suggest that our policies can reduce the mean response time by orders of magnitude.

The rest of the paper is organized as follows. Section 2 discusses single threshold allocation policies, and Section 3 discusses multiple threshold allocation policies. In Sections 2-3, we evaluate the policies with respect to mean response time and static robustness. In Section 4, we shift our interest to dynamic robustness. In Section 5, we study a real-world call center fitting our model.

2 Analysis of single threshold allocation policies

In this section, we analytically study the mean response time and static robustness of two single threshold allocation policies. The T1 policy (Section 2.1) places a threshold, t_1, on queue 1, whereby server 2 serves type 1 jobs whenever the length of queue 1 is at least t_1. Thus, under T1, the *beneficiary* queue (queue 1) has control. Our second policy, the T2 policy (Section 2.2), places a threshold, t_2, on queue 2, whereby server 2 serves type 1 jobs whenever the length of queue 2 is *below* t_2. In this policy, the donor queue (queue 2) has control.

In Section 2.1.2, we introduce a nearly exact analysis of the T1 policy based on DR. (DR also enables the analysis of the T2 policy and the ADT policy.) Our analysis will show that the T1 policy is superior to the T2 policy with respect to minimizing the mean response time, but that the T2 policy is superior with respect to static robustness.

2.1 T1 policy

The T1 policy is formally defined as follows:

Definition 1 *Let N_1 (respectively, N_2) denote the number of jobs at queue 1 (respectively, queue 2). The T1 policy with parameter t_1, the T1(t_1) policy, is characterized by the following set of rules, all of which are enforced preemptively (preemptive-resume):*

- *Server 1 serves only its own jobs.*
- *Server 2 serves jobs from queue 1 if either (i) $N_1 \geq t_1$ or (ii) $N_2 = 0$ & $N_1 \geq 2$. Otherwise, server 2 serves jobs from queue 2.*

To achieve maximal efficiency, we assume the following exceptions. When $N_1 = 1$ and $N_2 = 0$, the job is processed by server 2 if and only if $\mu_1 < \mu_{12}$. Also, when $t_1 = 1$ and $N_1 = 1$, the job in queue 1 is processed by server 2 if and only if $\mu_1 < \mu_{12}$ regardless of the number of type 2 jobs.

Note that we will discuss the nonpreemptive case in Section 5.

Figure 2 shows the jobs processed by server 2 as a function of N_1 and N_2 under the T1 policy. Observe that the T1(1) policy is the cμ rule when $c_1\mu_{12} > c_2\mu_2$, and the T1(∞) policy is the cμ rule when $c_1\mu_{12} \leq c_2\mu_2$; thus the $c\mu$ rule falls within the broader class of T1 policies.

2.1.1 Stability under the T1 policy

In the T1 policy, higher t_1 values yield the larger stability region, and in the limit as $t_1 \to \infty$, the queues under the T1 policy are stable as long as $\hat{\rho}_1 < 1$ and $\rho_2 < 1$. More formally,

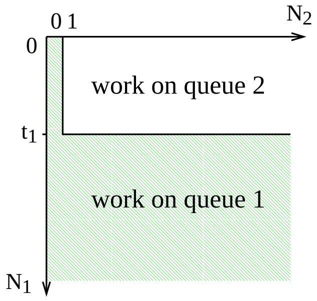


Figure 2: *Figure shows whether server 2 works on jobs from queue 1 or queue 2 as a function of N_1 and N_2, under the T1 policy with parameter t_1.*

Theorem 1 *Under the T1 policy with parameter $t_1 < \infty$, queue 1 is stable if and only if $\lambda_1 < \mu_1 + \mu_{12}$. Stability of queue 2 is given by the following conditions:*

- *For $1 < t_1 < \infty$, queue 2 is stable if and only if*

$$\rho_2 \quad < \quad \begin{cases} \frac{1-\rho_1^{t_1}}{1-\rho_1^{t_1}+\frac{(1-\rho_1)\rho_1^{t_1}}{1-\rho_1+\mu_{12}/\mu_1}} & \text{if } \rho_1 \neq 1, \\ \\ \frac{t_1}{t_1+\lambda_1/\mu_{12}} & \text{if } \rho_1 = 1. \end{cases} \tag{1}$$

- *For $t_1 = 1$, if $\mu_1 \geq \mu_{12}$, queue 2 is stable if and only if equation (1) holds with $t_1 = 2$.*

- *For $t_1 = 1$, if $\mu_1 < \mu_{12}$, queue 2 is stable if and only if*

$$\rho_2 < \frac{1}{1+\frac{\rho_1+\lambda_1/\mu_{12}}{1-\rho_1+\mu_{12}/\mu_1}}.$$

Proof: We prove only the case when $t_1 > 1$ and $\rho_1 \neq 1$. The case when $t_1 = 1$ or $\rho_1 = 1$ can be proved in a similar way. Let $N = (N_1, N_2)$ be the joint process of the number of jobs in queue 1 and queue 2, respectively. The expected length of a "busy period," during which $N_1 \geq t_1$, is finite if and only if $\lambda_1 < \mu_1 + \mu_{12}$. This proves the stability condition for queue 1.

Based on the strong law of large numbers, the necessary and sufficient condition for stability of queue 2 is $\rho_2 < F$, where F is the time average fraction of time that server 2 processes type 2 jobs given $N_2 > 0$. Below, we derive F. Let $\tilde{N} = (\tilde{N}_1, \tilde{N}_2)$ be a process in which $\tilde{N}$ behaves the same as N except that it has no transition from $\tilde{N}_2 = 1$ to $\tilde{N}_2 = 0$. Consider a semi-Markov process of $\tilde{N}_1$, where the state space is $(0,1,2,...,t_1 - 1, t_1^+)$. The state n denotes there are n jobs in queue 1 for $n = 0, 1, ..., t_1 - 1$, and the state t_1^+ denotes there are *at least* t_1 jobs in queue 1. The expected sojourn time is $1/\lambda_1$ for state 0, $1/(\lambda_1 + \mu_1)$ for states $n = 1, ..., t_1 - 1$, and $b = \frac{1/(\mu_1+\mu_{12})}{1-\lambda_1/(\mu_1+\mu_{12})}$ for state t_1^+, where b is the mean duration of the busy period in an M/M/1 queue with arrival rate λ_1 and service rate $\mu_1 + \mu_{12}$. The limiting probabilities for the corresponding *embedded* discrete time Markov chain are $\pi_n = (1 + \rho_1)\rho_1^{n-1}\pi_0$ for $n = 1, ..., t_1 - 1$ and $\pi_{t_1^+} = \rho_1^{t_1}\pi_0$, where

$$\pi_0 = \frac{1 - \rho_1}{(1 + \rho_1^{t_1-1})(1 - \rho_1) + (1 + \rho_1)(1 - \rho_1^{t_1-1})}.$$

As server 2 can work on queue 2 if and only if $\tilde{N}_1 < t_1$, the fraction of time that server 2 can work on queue 2 is

$$F = \frac{\pi_0/\lambda_1 + (1 - \pi_0 - \pi_{t_1^+})/(\lambda_1 + \mu_1)}{\pi_0/\lambda_1 + (1 - \pi_0 - \pi_{t_1^+})/(\lambda_1 + \mu_1) + b\pi_{t_1^+}} = \frac{1 - \rho_1^{t_1}}{1 - \rho_1^{t_1} + \frac{(1-\rho_1)\rho_1^{t_1}}{1-\rho_1+\mu_{12}/\mu_1}}.$$

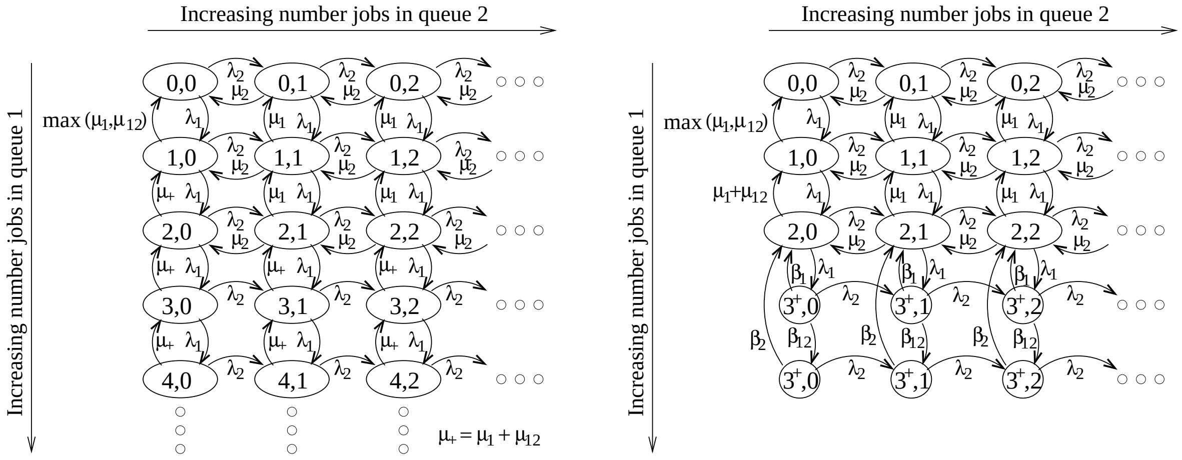
■

The following corollary is an immediate consequence of Theorem 1.

Corollary 1 *Under the T1 policy, the stability region increases with t_1 (i.e., the right hand side of equation (1) is an increasing function of t_1).*

2.1.2 Analysis of the T1 policy

Our analysis of the T1 and other threshold-based policies is based on dimensionality reduction, DR (see for example Osogami, 2005). Advantages of DR include computational efficiency, accuracy, and simplicity; these allow us to extensively investigate the performance characteristics of the allocation policies. DR reduces a 2D-infinite Markov chain (see Figure 3(a)) to a 1D-infinite Markov chain (see Figure 3(b)), which closely approximates the 2D-infinite Markov chain. To derive the mean response time of T1, the 1D-infinite Markov chain tracks the exact number of type 2 jobs, but tracks the number of type 1 jobs only up to the point $t_1 - 1$. At this point a type 1 arrival starts a "busy period," during which both servers are working on type 1 jobs, and



(a) 2D infinite Markov chain

(b) 1D infinite Markov chain

Figure 3: *Markov chains that model the behavior under the T1(3) policy. In figures, (i, j) represents the state where there are i jobs of type 1 and j jobs of type 2. In (a), $\mu_+ \equiv \mu_1 + \mu_{12}$.*

type 2 jobs receive *no* service. This "busy period" ends when there are once again $t_1 - 1$ jobs of type 1. State (t_1^+, j) denotes there are *at least* t_1 jobs of type 1 and there are j jobs of type 2 for $j \geq 0$. The key point is that there is no need to track the exact number of type 1 jobs during this busy period. We approximate the duration of this busy period with a two-phase phase type (PH) distribution with parameters $(\beta_1, \beta_2, \beta_{12})$, matching the first three moments (Osogami, 2005).

We use the limiting probabilities of the Markov chain in Figure 3(b) to calculate the mean number of jobs of each type, $E[N_1]$ and $E[N_2]$, which in turn gives their mean response time via Little's law. These limiting probabilities can be obtained efficiently via matrix analytic methods (Latouche and Ramaswami, 1999). Deriving $E[N_2]$ from the limiting probabilities is straightforward, since we track the exact number of type 2 jobs. We derive $E[N_1]$ by conditioning on the state of the chain. Let $E[N_1]_{ij}$ denote the expected number of type 1 jobs given that the chain is in state (i, j). For $i = 0, ..., t_1 - 1$, $E[N_1]_{ij} = i$ for all j. For $i = t_1^+$, $E[N_1]_{t_1^+, j}$ is the mean number of jobs in an M/M/1 system given that the service rate is the sum of the two servers, $\mu_1 + \mu_{12}$, and given that the system is busy, plus an additional t_1 jobs.

We find that the mean response time computed via DR is usually within two percent of the simulated value (Osogami, 2005). The high accuracy of DR stems from the fact that the state space of the 2D-infinite Markov chain in Figure 3(a) is not simply truncated. Rather, two rows representing 3^+ jobs of type 1 in the 1D-infinite Markov chain in Figure 3(b) capture the infinite number of rows (row 4, 5, ...) in the 2D-infinite Markov chain in such a way that the first three moments of the sojourn time distribution in these two regions agree.

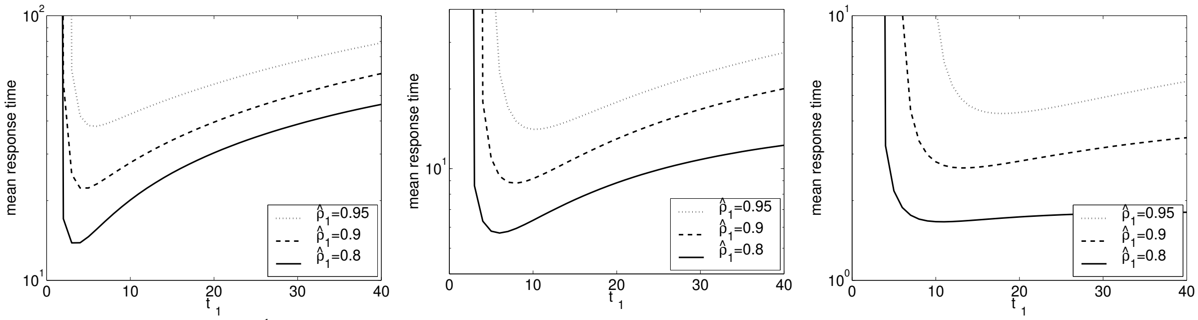
2.1.3 Characterizing the performance of the T1 policy

Our analysis of Section 2.1.2 allows an efficient and accurate analysis of the T1 policy. In this section, we characterize this performance. We find that the behavior of the T1 policy is quite different depending on whether queue 2 prefers type 1 or type 2 ($c_1\mu_{12} \le c_2\mu_2$ or $c_1\mu_{12} > c_2\mu_2$). The optimal t_1 threshold is typically finite when $c_1\mu_{12} > c_2\mu_2$, and typically infinite when $c_1\mu_{12} \le c_2\mu_2$, where the optimality is with respect to minimizing the mean response time. In fact, when $c_1 = c_2$ and $c_1\mu_{12} \le c_2\mu_2$, the following theorem holds (the theorem *may* extend to the case of $c_1\mu_{12} \le c_2\mu_2$ with general c_1 and c_2, but the general case is not proved).

Theorem 2 *If $c_1 = c_2$ and $c_1\mu_{12} \le c_2\mu_2$, the mean response time of the T1 policy is minimized at $t_1 = \infty$ (i.e., the $c\mu$-rule is optimal).*

Proof: Due to Little's law, it is sufficient to prove that the number of jobs completed under the T1(∞) policy is stochastically larger than those completed under the T1 policy with $t_1 < \infty$ at any moment. Let $N^{\mathrm{inf}}(t) = (N_1^{\mathrm{inf}}(t), N_2^{\mathrm{inf}}(t))$ be the joint process of the number of jobs in queue 1 and queue 2, respectively, at time t when $t_1 = \infty$. Let $N^{\mathrm{fin}}(t) = (N_1^{\mathrm{fin}}(t), N_2^{\mathrm{fin}}(t))$ be defined analogously for $t_1 < \infty$. With $t_1 = \infty$, server 2 processes type 2 jobs as long as there are type 2 jobs, and thus $N_1^{\mathrm{inf}}(t)$ is stochastically larger than $N_1^{\mathrm{fin}}(t)$ for all t. Consequently, the number of jobs completed by sever 1 is stochastically smaller when $t_1 < \infty$ than when $t_1 = \infty$ at any moment, since server 1 is work-conserving.

As long as server 2 is busy, the number of jobs completed by sever 2 is stochastically smaller when $t_1 < \infty$ than when $t_1 = \infty$, since $\mu_{12} \le \mu_2$. Also, using coupling argument, we can show that the system with $t_1 = \infty$ has server 2 go idle (both queues empty) earlier (stochastically) than the $t_1 < \infty$ system. Thus, when server 2 is idle the number of jobs completed by the $t_1 = \infty$ system is stochastically larger. Thus, the number of jobs completed (either by server 1 or by server 2) under the T1(∞) policy is stochastically larger than that completed under the T1 policy with $t_1 < \infty$. ■



(a) $c_1\mu_1 = \frac{1}{4}$ (b) $c_1\mu_1 = 1$ (c) $c_1\mu_1 = 4$

Figure 4: *The mean response time under the T1 policy as a function of* t_1*. Here,* $c_1 = c_2 = 1$*,* $c_1\mu_{12} = 1$*,* $c_2\mu_2 = \frac{1}{16}$*, and* $\rho_2 = 0.6$ *are fixed.*

Since $t_1 = \infty$ achieves the largest stability region (Corollary 1), if $c_1 = c_2$, $t_1 = \infty$ is the optimal choice with respect to both mean response time and the stability region. Note that the T1(∞) policy is the policy of following the $c\mu$ rule, as server 2 "prefers" to run its own jobs in a $c\mu$ sense when $c_1\mu_{12} \le c_2\mu_2$. Therefore, below we limit our attention to the case of $c_1\mu_{12} > c_2\mu_2$, where server 2 "prefers" to run type 1 jobs in a $c\mu$ sense. Note that condition $c_1\mu_{12} > c_2\mu_2$ is achieved when type 1 jobs are small and type 2 jobs are large, when type 1 jobs are more important than type 2 jobs, and/or in the pathological case when type 1 jobs have good affinity with server 2. (These, in addition, may motivate use of the Beneficiary-Donor model, giving smaller or more important jobs better service.) We will see that the optimal t_1 threshold is typically finite when $c_1\mu_{12} > c_2\mu_2$, in contrast to the $c\mu$ rule.

Figure 4 shows the mean response time under the T1 policy as a function of t_1; we see that optimal t_1 is finite and depends on environmental conditions such as load ($\hat\rho_1$) and job sizes (μ_1). Here, different columns correspond to different μ_1's. In each column, the mean response time is evaluated at three loads, $\hat\rho_1 = 0.8, 0.9, 0.95$, by changing λ_1 (Note that $\hat\rho_1 = 0.8, 0.9, 0.95$ corresponds to $\rho_1 = 2.08, 2.34, 2.47$ when $\mu_1 = 1/4$ in column 1, $\rho_1 = 1.12, 1.26, 1.33$ when $\mu_1 = 1$ in column 2, and $\rho_1 = 0.88, 0.99, 1.05$ when $\mu_1 = 4$ in column 3.) By Theorem 1, a larger value of t_1 leads to a larger stability region, and hence there is a tradeoff between good performance at the estimated load, $(\hat\rho_1, \rho_2)$, which is achieved at smaller t_1, and stability at higher $\hat\rho_1$ and/or ρ_2, which is achieved at larger t_1. Note also that the curves have sharper "V shapes" in general

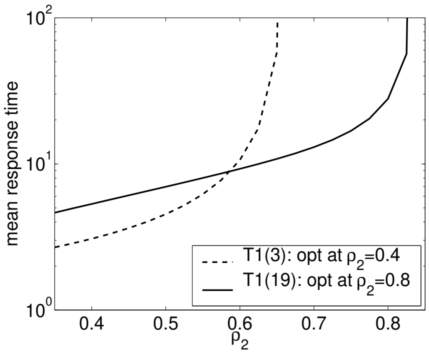


Figure 5: *The mean response time under the T1(3) policy and the T1(19) policy as a function of* ρ_2*, where* $c_1 = c_2 = 1$*,* $c_1\mu_1 = c_1\mu_{12} = 1$*,* $c_2\mu_2 = \frac{1}{16}$*, and* $\rho_1 = 1.15$ *are fixed.*

at higher $\hat{\rho}_1$, which complicates the choice of t_1, since the mean response time quickly diverges to infinity as t_1 becomes smaller.

In addition, analyses (Osogami, 2005) show that the value of the $c\mu$ product primarily determines the behavior of the T1 policy, and individual values of c and μ have smaller effect. Also, when ρ_2 is lower (and thus ρ_1 is higher for a fixed $\hat{\rho}_1$), the optimal t_1 tends to become smaller, and hence the tradeoff between the performance at the estimated load and stability at higher loads is more significant. This makes intuitive sense, since at lower ρ_2, server 2 can help more.

Figure 5 highlights the static robustness of the T1 policy, plotting the mean response time as a function of ρ_2 (only λ_2 is changed) for T1(3) and T1(19). When $\rho_2 = 0.4$, $t_1 = 3$ is the optimal threshold (but T1(19) still provides finite mean response time). However, if it turns out that $\rho_2 = 0.8$ is the actual load, then the T1(3) policy leads to instability (infinite mean response time), while the T1(19) policy minimizes mean response time. Thus, choosing a higher t_1 (=19) guarantees stability against misestimation of the load, but results in worse performance at the estimated load. This experiment and others like it (Osogami, 2005) lead us to conclude that the T1 policy is poor with respect to static robustness.

2.2 T2 policy

In this section, we investigate the mean response time and static robustness of the T2 policy, comparing it to the T1 policy. The T2 policy is formally defined as follows:

Definition 2 *The T2 policy with parameter* t_2*, the T2(*t_2*) policy, is characterized by the following*

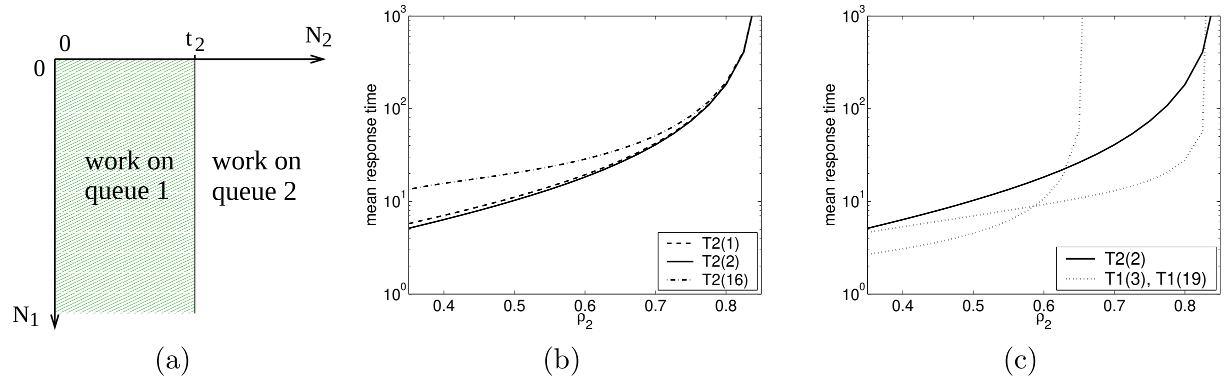


Figure 6: *Part (a) shows whether server 2 works on jobs from queue 1 or queue 2 as a function of N_1 and N_2, under the T2 policy with parameter t_2. Part (b) shows the mean response time under the T2 policy with various t_2 threshold values as a function of ρ_2. Part (c) compares the mean response time under the "optimized" T2 policy and two T1 policies. In (b) and (c), $c_1 = c_2 = 1$, $c_1\mu_1 = c_1\mu_{12} = 1$, $c_2\mu_2 = \frac{1}{16}$, and $\rho_1 = 1.15$ are fixed.*

set of rules, all of which are enforced preemptively (preemptive-resume):

- *Server 1 serves only its own jobs.*
- *Server 2 serves jobs from queue 1 if $N_2 < t_2$. Otherwise server 2 serves jobs from queue 2.*

When $N_1 = 1$ and $N_2 = 0$, we allow the same exception as in the T1 policy.

Figure 6(a) shows the jobs processed by server 2 as a function of N_1 and N_2 under the T2 policy. Recall that the T1 policy guarantees stability whenever $\hat{\rho}_1 < 1$ and $\rho_2 < 1$ *provided* that t_1 is chosen appropriately. By contrast, the T2 policy guarantees stability whenever $\hat{\rho}_1 < 1$ and $\rho_2 < 1$ for *any* finite t_2. It clearly dominates the T1 policy in this respect. More formally, the following theorem holds, which can be proved in a similar way as Theorem 1.

Theorem 3 *Under the T2 policy with $t_2 < \infty$, queue 1 is stable if and only if $\hat{\rho}_1 < 1$, and queue 2 is stable if and only if $\rho_2 < 1$.*

2.2.1 Assessing the performance of the T2 policy

It is not at all obvious how the T2 policy's performance compares with that of the T1 policy, when each is run with its optimal threshold. In this section, we investigate this question. The T2 policy can be analyzed via DR as in Section 2.1.2, approximating the 2D-infinite Markov chain by a 1D-infinite Markov chain tracking the exact number of *type 1* jobs (cf. the 1D-infinite Markov chain for the T1 policy tracks the exact number of type 2 jobs). With respect to the number of type 2 jobs, the chain differentiates only between 0, 1, ..., $t_2 - 1$, or t_2^+ jobs.

Figure 6(b) illustrates that the mean response time under the T2 policy is minimized at a small t_2 (in this case $t_2 = 2$) for a range of load. This figure is representative of a wide range of parameter values that we studied. Since choosing a small t_2 minimizes the mean response time and still provides the maximum stability region, there is no tradeoff between minimizing the mean response time and maximizing the stability region with the T2 policy.

However, Figure 6(c) (and many other experiments like it) imply that the mean response time under the T2 policy with the optimal t_2 is typically higher than that under the T1 policy with the optimal t_1. We conclude that, although the T2 policy has more static robustness than the T1 policy, it performs worse with respect to mean response time.

3 Analysis of multi-threshold allocation policies

The tradeoff between the low mean response time of the T1 policy and good static robustness of the T2 policy motivates us to introduce a class of multi-threshold allocation policies: ADT policies. We will study how the mean response time and static robustness of these multi-threshold allocation policies compare to that of the single threshold allocation policies.

3.1 The adaptive dual threshold (ADT) policy

The key idea in the design of the ADT policy is that we want the ADT policy to operate as a T1 policy to ensure low mean response time, but we will allow the value of t_1 to adapt, depending on the length of queue 2, to provide static robustness. Specifically, the ADT policy behaves like the T1 policy with parameter $t_1^{(1)}$ if the length of queue 2 is less than t_2 and otherwise like the T1 policy with parameter $t_1^{(2)}$, where $t_1^{(2)} > t_1^{(1)}$. We will see that, indeed, the ADT policy is far superior to the T1 policy with respect to static robustness. In addition, one might also expect that the mean response time of the optimized ADT policy will significantly improve upon that of the optimized T1 policy, since the ADT policy generalizes the T1 policy (the ADT policy is reduced to the T1 policy by setting $t_1^{(1)} = t_1^{(2)}$). However, this turns out to be largely false, as we

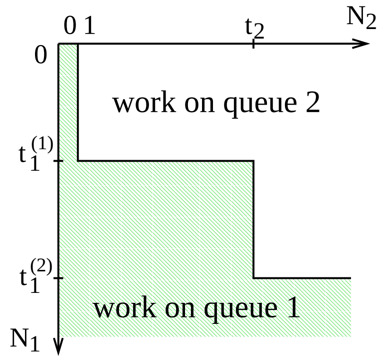


Figure 7: *Figure shows whether server 2 works on jobs from queue 1 or queue 2 as a function of N_1 and N_2 under the ADT policy with parameters $t_1^{(1)}$, $t_1^{(2)}$, and t_2.*

see below. Formally, the ADT policy is characterized by the following rule.

Definition 3 *The ADT policy with parameters $t_1^{(1)}$, $t_1^{(2)}$, and t_2, the ADT($t_1^{(1)}, t_1^{(2)}, t_2$) policy, operates as the T1($t_1^{(1)}$) policy if $N_2 \le t_2$; otherwise, it operates as the T1($t_1^{(2)}$) policy.*

Figure 7 shows the jobs processed by server 2 under the ADT policy as a function of N_1 and N_2. (A separate class of multi-threshold allocation policies that place only *one* threshold on queue 1 and one on queue 2, the T1T2 policy, is introduced in Osogami et al., 2004. Its mean response time and static robustness are only marginally improved over T1 and T2 policies. Thus, T1T2 is in general inferior to ADT.)

At high enough $\hat{\rho}_1$ and ρ_2, N_2 usually exceeds t_2, and the policy behaves similar to the T1 policy with parameter $t_1^{(2)}$. Thus, the stability condition for ADT is the same as that for T1 with parameter $t_1^{(2)}$. The following theorem can be proved in a similar way as Theorem 1.

Theorem 4 *The stability condition for the ADT policy with parameters $t_1^{(1)}$, $t_1^{(2)}$, and t_2 is given by the stability condition for the T1 policy with parameter $t_1^{(2)}$ (Theorem 1).*

The ADT policy can likewise be analyzed via DR as in Section 2.1.2, by approximating the 2D-infinite Markov chain by a 1D-infinite Markov chain (see Figure 8). For the ADT policy, the 1D-infinite Markov chain tracks the exact number of type 2 jobs, but tracks the number of type 1 jobs only up to the point where there are $t_1^{(2)} - 1$ jobs. A type 1 arrival at this point starts a "busy period," which ends when there are once again $t_1^{(2)} - 1$ jobs of type 1. We approximate the duration of this busy period with a two-phase PH distribution with parameters $(\beta_1, \beta_2, \beta_{12})$,

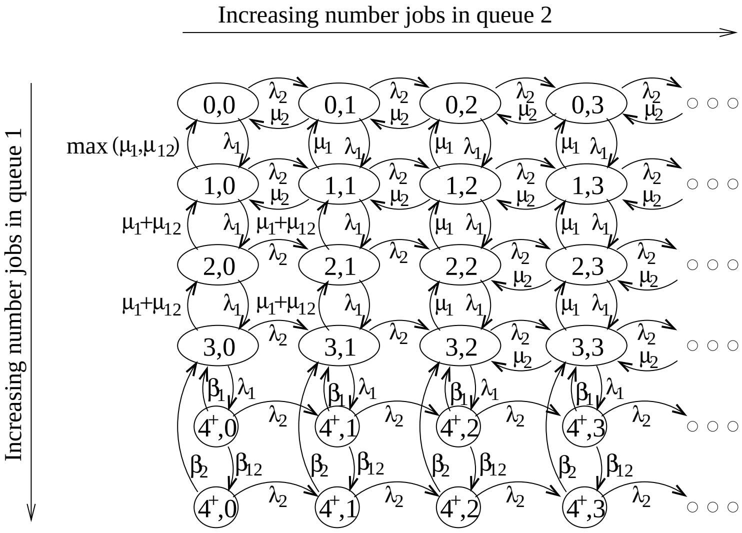


Figure 8: *The 1D-infinite Markov chain that models the behavior under the ADT(2,4,2) policy.*

matching the first three moments as before. State $(t_1^{(2)^+}, j)$ denotes that there are *at least* $t_1^{(2)}$ jobs of type 1 and there are j jobs of type 2 for $j \geq 0$. The mean response time is again obtained via matrix analytic methods. In Appendix A.1, we analyze the ADT policy more formally.

3.2 Results: Static robustness of the ADT policy

Figure 9 illustrates static robustness of the ADT policy, showing the mean response time under the ADT policy as a function of ρ_2; the ADT policy achieves at least as low mean response time as the *better* of the T1 policies with the two different t_1 values throughout the range of ρ_2. Though not shown, the ADT policy is also (statically) robust against misestimation of $\hat{\rho}_1$ (Osogami, 2005).

The robustness of the ADT policy can be attributed to the following. The dual thresholds on queue 1 make the ADT policy adaptive to misestimation of load, in that the ADT policy with parameters $t_1^{(1)}$, $t_1^{(2)}$, and t_2 operates like the T1 policy with parameter $t_1^{(1)}$ at the estimated load and like the T1 policy with parameter $t_1^{(2)}$ at a higher load, where $t_1^{(2)} > t_1^{(1)}$. Thus, server 2 can help queue 1 less when there are more type 2 jobs, preventing server 2 from becoming overloaded. This leads to the increased stability region and improved performance.

In specifying the three thresholds, $t_1^{(1)}$, $t_1^{(2)}$, and t_2, for the ADT policy in Figure 9, we have used the following sequential heuristic:

1. Set $t_1^{(1)}$ as the optimal t_1 value for the T1 policy at the estimated (given) load.

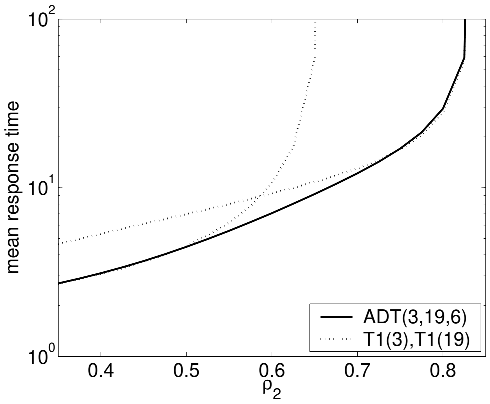


Figure 9: *The mean response time under the ADT policy as a function of ρ_2. Here, $c_1 = c_2 = 1$, $c_1\mu_1 = c_1\mu_{12} = 1$, $c_2\mu_2 = \frac{1}{16}$, and $\rho_1 = 1.15$ are fixed.*

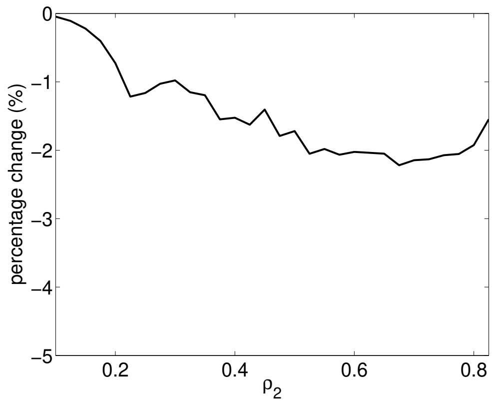
2. Choose $t_1^{(2)}$ so that it achieves stability in a desired range of load. We find that the mean response time at the estimated load is relatively insensitive to $t_1^{(2)}$, and hence we can choose a high $t_1^{(2)}$ to guarantee a large stability region.

3. Find t_2 such that the policy provides both low mean response time at the estimated load and good static robustness. This is a nontrivial task. If t_2 is set too low, the ADT policy behaves like the T1 policy with parameter $t_1^{(2)}$, degrading the mean response time at the estimated load, since $t_1^{(2)}$ is larger than the optimal t_1 in the T1 policy. If t_2 is set too high, the ADT policy behaves like the T1 policy with parameter $t_1^{(1)}$. This worsens the mean response time at loads higher than the estimated load. In plotting Figure 9, we found "good" t_2 values manually by trying a few different values, which took only a few minutes.

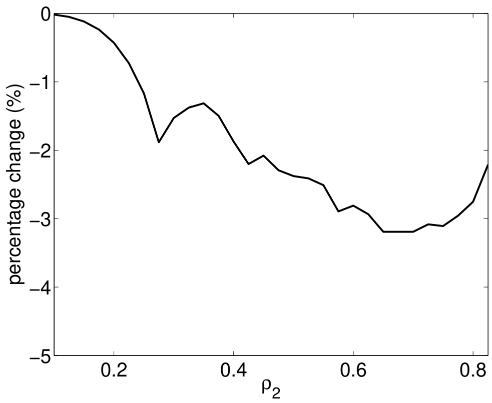
Observe that since the stability region is insensitive to $t_1^{(1)}$ and t_2, we can choose these values so that the mean response time at the estimated load is optimized.

3.3 Results: Mean response time of the ADT policy

We have already seen the benefits of the ADT policy when the load is not exactly known (static robustness). One might also expect that, even when the load is known exactly, the ADT policy might significantly improve upon the T1 policy with respect to mean response time. Earlier work of Meyn (2001) provides some support for this expectation; Meyn shows via numerical examples that, in the case of finite buffers for both queues, the policy that minimizes mean response time



(a) $c_2\mu_2 = \frac{1}{4}$



(b) $c_2\mu_2 = \frac{1}{16}$

Figure 10: *The percentage change (%) in the mean response time of the (locally) optimized ADT policy over the optimized T1 policy at each given load, as a function of ρ_2. A negative percentage indicates the* improvement *of ADT over T1. Here, $c_1 = c_2 = 1$, $c_1\mu_1 = c_1\mu_{12} = 1$, and $\rho_1 = 1.15$ are fixed.*

is a "flexible" T1 policy which allows a continuum of T1 thresholds, $\{t_1^{(i)}\}$, where threshold $t_1^{(i)}$ is used when the length of queue 2 is i. The ADT policy can be seen as an approximation of a "flexible" T1 policy, using only two t_1 thresholds.

To evaluate the benefit of the ADT policy, we compare it over a range of ρ_2 against the T1 policy optimized for the given ρ_2. Since the search space of the threshold values for the ADT policy is large, we find *locally* optimal threshold values, which are found to be optimal within a search space of ± 5 for each threshold. We measure the percentage change in the mean response time of ADT versus T1:

$$\frac{E[R_{ADT}] - E[R_{T1}]}{E[R_{T1}]} \times 100 \quad (\%), \tag{2}$$

where $E[R_X]$ denotes the mean response time in policy $X \in \{\text{ADT,T1}\}$.

Figure 10 shows the percentage reduction in the mean response time of the locally optimized ADT policy over the T1 policy optimized at each ρ_2, as a function of ρ_2. Figure 10 shows that, surprisingly, the benefit of the ADT policy is quite small with respect to mean response time under fixed Poisson arrivals; the improvement of the ADT policy is larger at moderately high ρ_2 and at smaller $c_2\mu_2$ value, but overall the improvement is typically within 3%. We conjecture that adding more thresholds (approaching the flexible T1 policy) will not improve mean response time appreciably, given the small improvement from one to two thresholds. Thus, whereas the

ADT policy has significant benefits over the simpler T1 policy with respect to static robustness, the two policies are comparable with respect to mean response time.

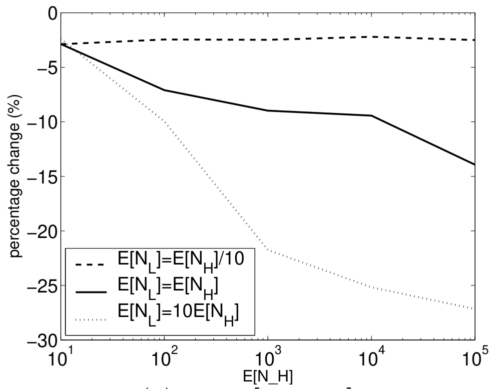
4 Dynamic robustness of threshold-based policies

We have seen, in Section 3.1, that the mean response time of the optimized ADT policy is similar to that of the optimized T1 policy, although the ADT policy has greater static robustness. Note that this observation is based on the assumption of Poisson arrivals. Consider, to start, an alternate scenario, in which the load at queue 2 fluctuates. For example, a long high load period (e.g., $\rho_1 = 1.15$ and $\rho_2 = 0.8$) is followed by a long low load period (e.g., $\rho_1 = 1.15$ and $\rho_2 = 0.4$), and the high and low load periods alternate. The T1 policy with a fixed threshold value must have a high mean response time either during the high load period or during the low load period (recall Figure 5). On the other hand, the ADT policy may provide low mean response time during both high and low load periods, since the t_1 threshold value is self-adapted to the load (recall Figure 9). In this section, we study the mean response time of the ADT policy when the load fluctuates, or the *dynamic robustness* of the ADT policy.

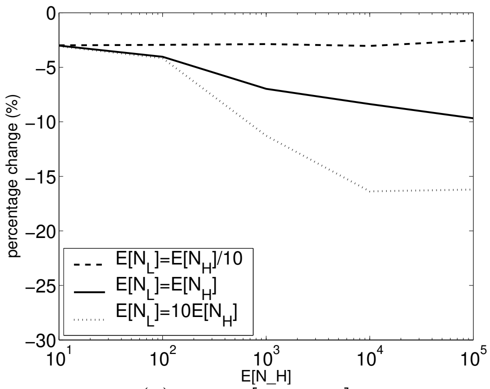
We use a Markov modulated Poisson process of order two (MMPP(2)) as an arrival process at queue 2. An MMPP(2) has two phases, which we denote as the high load phase and the low load phase. The duration of each phase has an exponential distribution, which can differ in each phase. During the high (respectively, low) load phase, the arrival process follows a Poisson process with rate λ_H (respectively, λ_L), where $\lambda_H > \lambda_L$. We postpone describing the techniques used to analyze the ADT policy under the MMPP to Appendix A.2, and first study the results.

Results: Dynamic robustness of the ADT policy vs. the T1 policy

Figure 11 shows the percentage change in the mean response time of the (locally) optimized ADT policy over the optimized T1 policy, when arrivals at queue 1 follow a Poisson process and arrivals at queue 2 follow an MMPP(2). The arrival rates in the MMPP(2) are chosen such that the load during the high load period is $\rho_2 = 0.8$ and the load during the low load period is $\rho_2 = 0.2$, 0.4, or 0.6, while $\rho_1 = 1.15$ is fixed throughout. We choose the horizontal axis to be the expected number of type 2 arrivals during a high load period, and the three lines (solid, dashed, and dotted) to



(a) $\rho_2 = [0.2, 0.8]$



(c) $\rho_2 = [0.6, 0.8]$

Figure 11: *The percentage change (%) in the mean response time of the (locally) optimized ADT policy over the optimized T1 policy for each given MMPP(2), shown as a function of the expected number of arrivals during a high load period, $E[N_H]$, and during a low load period, $E[N_L]$. A negative percentage indicates the* improvement *of ADT over T1. Here, $c_1 = c_2 = 1$, $c_1\mu_1 = c_1\mu_{12} = 1$, $c_2\mu_2 = \frac{1}{16}$, and $\rho_1 = 1.15$ are fixed.*

be different expected number of type 2 arrivals during a low load period. Note that since there are less frequent arrivals during a low load period, having the same number of arrivals during the high and low load periods implies that the low load period is longer. Thus, the number of arrivals during each period is an indicator of how frequently the load changes, which we find to be an important parameter in studying dynamic robustness. The threshold values of the optimized T1 policy and the (locally) optimized ADT policy are chosen such that the overall weighted mean response time is minimized for each given arrival process.

The first thing to notice in Figure 11 is that the improvement of the ADT policy over the T1 policy is smaller when the duration of the high and low load periods is shorter, or equivalently when there are less arrivals in each period. This makes intuitive sense, since the MMPP(2) reduces to a Poisson process when the high and low load periods alternate infinitely quickly, and under the Poisson process, the optimized T1 policy and the optimized ADT policy provide similar mean response time; see Section 3.1.

However, even when the durations are longer, the performance improvement of the ADT policy over the T1 policy is comparatively small (3 to 25%). This is mainly because the mean response time of the jobs arriving during the high load period tends to dominate the *overall* mean response time for two reasons: (i) the response time of jobs arriving during the high load period is much higher than that of jobs arriving during the low load period, partially due to the fact that any reasonable allocation policy (such as the optimized T1 policy and the optimized ADT policy) can provide low mean response time at low load, and (ii) assuming that a low load period and a high load period have the same *duration*, there are more arrivals during a high load period. Since the T1 policy with a fixed t_1 threshold can provide low mean response time for the jobs arriving during the high load period, it can provide low *overall* mean response time. (Of course, if there were *many* more arrivals during the low load period than the high load period, the t_1 threshold would be adjusted.)

It is only when the jobs arriving during the high load period and the jobs arriving during the low load period have roughly equal contribution to the *overall* mean response time that the ADT policy can have appreciable improvement over the T1 policy. This happens when $\sum_{i=1}^{2} c_i p_i^L E[R_i^L] \sim \sum_{i=1}^{2} c_i p_i^H E[R_i^H]$, where p_i^L (respectively, p_i^H) is the fraction of jobs that are type i and arriving during the low (respectively, high) load period, and $E[R_i^L]$ (respectively, $E[R_i^H]$) is the mean response time of type i jobs arriving during the low (respectively, high) load period, for $i = 1, 2$. For example, Figure 11 suggests that the ADT policy can provide a mean response time that is 20-30% lower than that of the T1 policy, when the number of arrivals during a low load period is ($\sim$ 10 times) larger than that during a high load period.

In addition (not shown), we find that when arrivals at queue 1 follow an MMPP(2) or when arrivals at both queues follow MMPP(2)'s, the improvement of the ADT policy over the T1 policy tends to be smaller than when only arrivals at queue 2 follow an MMPP(2). Overall, we conclude that ADT has appreciable improvement over T1 only when there are more arrivals during the low load period than during the high load period, giving them comparable importance.

5 Application to call center scheduling

In this section we apply lessons of previous sections to designing allocation policies in a telephone call center simulated using traces at a call center of an anonymous bank in Israel in 1999 provided by Guedj and Mandelbaum (2000). This call center uses a service architecture that is similar to the Beneficiary-Donor model, based on different classes of callers. Our goals are to assess what improvement may be possible for the call center through the implementation of threshold-based policies, and more generally to evaluate some of the high level principles of prior sections. For this purpose, we will first study some relevant characteristics of the trace in Section 5.1 (see Brown et al., 2005, for a complementary study of the trace). In particular, we will see that the arrival rate at this call center has great fluctuation as in Section 4. Based on the lessons learned in previous sections, we expect that the T1 policy may perform well in this call center. We will evaluate this expectation via trace driven simulation in Section 5.2.

5.1 Settings

5.1.1 Trace characteristics

The data spans twelve months of 1999, and were collected at the level of individual calls, at a small call center of an anonymous bank in Israel. An arriving call is first connected to an interactive voice response unit (VRU), where the customer receives recorded information and possibly performs self-service transactions. In total, roughly 1,200,000 calls arrived at the VRU during the year of 1999; out of those, about 420,000 calls indicated a desire to speak to an agent. Below, we limit our focus on the 420,000 calls that requested connection to an agent.

The calls requesting connection to an agent can be divided into two types: Internet assistance (IN) calls and Regular calls. IN calls generally ask for technical support for online transactions via web sites. All the other calls are classified as Regular calls. Prior to August 1999, both the IN calls and the Regular calls joined a single shared queue, and were served by the same pool of agents. Post August 1999, the call center split the IN and Regular calls into two separate queues to be served by separate pools of agents (see Figure 12). In addition to distinguishing between two types of calls, the call center also differentiates between high and low priority customers, and looks for ways to give high priority customers shorter waiting times.

Table 1 summarizes the total number of calls of each type and of each priority class at the call center during 1999. Out of the approximately 420,000 total calls, about 400,000 calls are Regular, and about 20,000 calls are IN. Out of the 400,000 Regular calls, about 140,000 calls have high priority, and about 260,000 calls have low priority. By contrast, almost all IN calls have low priority. Table 2 shows the percentage of calls that are served by agents. About 85% of the calls are served, and 15% of the calls are abandoned before receiving service by agents (as there are

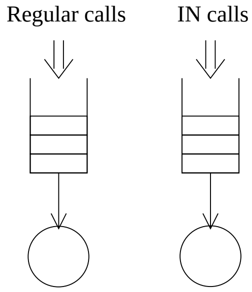


Figure 12: *Post-August architectural model of a call center.*

only two IN calls with high priority, the entry is kept blank in the table).

	both types	Regular	IN
both prio	419,857	400,765	19,092
high prio	137,317	137,315	2
low prio	282,540	263,450	19,090

Table 1: *Total number of calls during the year.*

	both types	Regular	IN
both prio	84.9%	85.2%	78.9%
high prio	85.8%	85.8%	-
low prio	84.4%	84.8%	78.9%

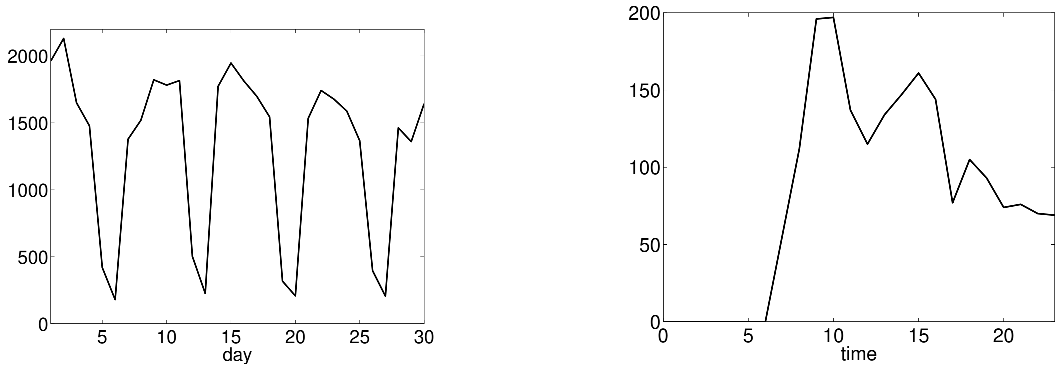
Table 2: *Percentage of calls served.*

Figure 13 details the total number of calls, showing (a) the daily number of calls during a month (November) and (b) the hourly number of calls during a day (November 1). As Figure 13(a) suggests, the number of calls per day drops on weekends (Fridays and Saturdays in Israel). As Figure 13(b) suggests, the call center opens at 7am on weekdays, and the number of calls per hour peaks before lunch time ($\sim$ 200 calls per hour). After lunch time, there is another peak, and then calls decline through the evening (to roughly 70 calls per hour). The arrival pattern does not differ much day to day during weekdays.

Table 3 summarizes the mean service demand (in seconds) and its squared coefficient of variation for those calls that are served. As there are only two IN calls with high priority, the entry is kept blank in the table. Note that the IN calls have noticeably longer service demand with higher variability, and this might be a reason for the call center to serve the IN calls by a separate pool of agents, so that the Regular calls are not blocked by long (low priority) IN calls.

5.1.2 Architectural models for experiment

In our experiment, we consider two possible service models for the call center, as shown in Figure 14. In both models, we assume that each queue has a single agent, approximating the fact that each queue is served by a pool of several agents at the call center (in fact, up to 13 agents



(a) Daily arrivals (b) Hourly arrivals

Figure 13: *A typical arrival pattern of all 420,000 calls. The figures show the number of (a) daily arrivals in November and (b) hourly arrivals on November 1.*

	both types	Regular	IN
both prio	190.1	180.6	406.3
high prio	208.7	208.7	-
low prio	180.9	165.7	406.3

(a) mean

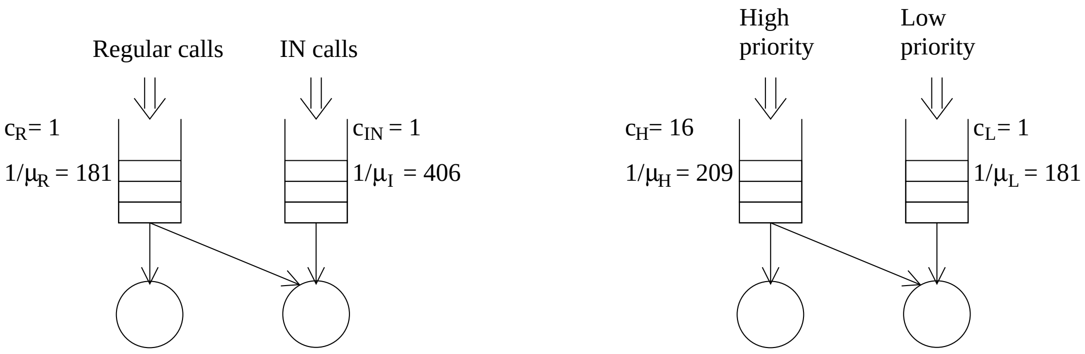
	both types	Regular	IN
both prio	2.217	1.836	2.974
high prio	1.871	1.871	-
low prio	2.420	1.741	2.974

(b) C^2

Table 3: *Statistics of the duration of a service: (a) mean (in seconds) and (b) squared coefficient variation.*

serve the call center). This approximation becomes more accurate as the load becomes higher, where the study of performance becomes important. In the Regular-IN model Figure 14(a), we separate the IN calls from the Regular calls as in the original architectural model of a call center (Figure 12), but allow the IN agent to sometimes serve the Regular calls. Since almost all IN calls have low priority while 34% of the Regular calls have high priority, we place more weight (importance) on the Regular calls (specifically, the Regular calls have weight $c_R = 4$, and the IN calls have weight $c_{IN} = 1$). As the IN calls have longer service demand, more variability, *and* less importance, we do not want the Regular call agent to serve the IN calls. In this section, we assume a nonpreemptive service discipline, following call center convention. (In previous sections we have chosen preemptive service disciplines for clarity, as the analysis of the nonpreemptive case is complex, though possible, as described in Osogami, 2005.)

In the Priority model Figure 14(b), we separate high priority calls from low priority calls, and place more weight on high priority jobs (specifically, high priority calls have weight $c_H = 16$, and



(a) Regular-IN model (b) Priority model

Figure 14: *Two architectural models of a call center.*

low priority calls have weight $c_L = 1$). As high priority calls and low priority calls have roughly the same service demand, high priority calls have higher $c\mu$ value; thus, we allow the agent for low priority calls to sometimes serve high priority calls.

5.1.3 Preprocessing of trace

We consider only those arrivals during weekdays, removing holidays (which have smaller numbers of calls). In our trace driven simulation, we feed the trace of the Regular or high priority calls into queue 1 and the trace of the IN or low priority calls into queue 2. In order to use our trace of limited length multiple times (specifically, 30 times in each run) with different arrival sequences, we follow a common approach in simulation (Uhlig and Mudge, 1997), whereby each call is independently removed from the trace with some probability, as specified below. Another reason for sampling the arrival sequence in this way is to create different loads. Hereby, we define the load at queue i, ρ_i, as follows for $i = 1, 2$:

$$\rho_i = \frac{(1 - q_i) \times (\text{total number of calls at queue } i) \times (\text{average duration of a service for queue } i)}{(\text{total operation hours})},$$

where q_i is the fraction of the calls at queue i removed from the trace.

We pick the service demand of a call from the lognormal distribution whose parameters are estimated from the trace. Picking service demands from a distribution allows us to use the same trace multiple times with different service demand sequences.

5.2 Results

Our trace characterization (Section 5.1.1) shows that the load at the call center has large fluctuation. We will see that the T1 policy (with a fixed t_1 threshold) provides a low mean response time even under this fluctuating load. In addition, we will also see how much improvement the call center can expect with respect to static and dynamic robustness by employing allocation policies such as the T1 and ADT policies, which allow resource sharing. For this purpose, we evaluate the following three allocation policies:

The Dedicated policy: Each agent serves only its own queue, as in the original call center model (Figure 12).

The T1 policy: The rules are specified in Definition 1, but they are enforced *non*preemptively.

The ADT policy: The rules are specified in Definition 3, but they are enforced *non*preemptively.

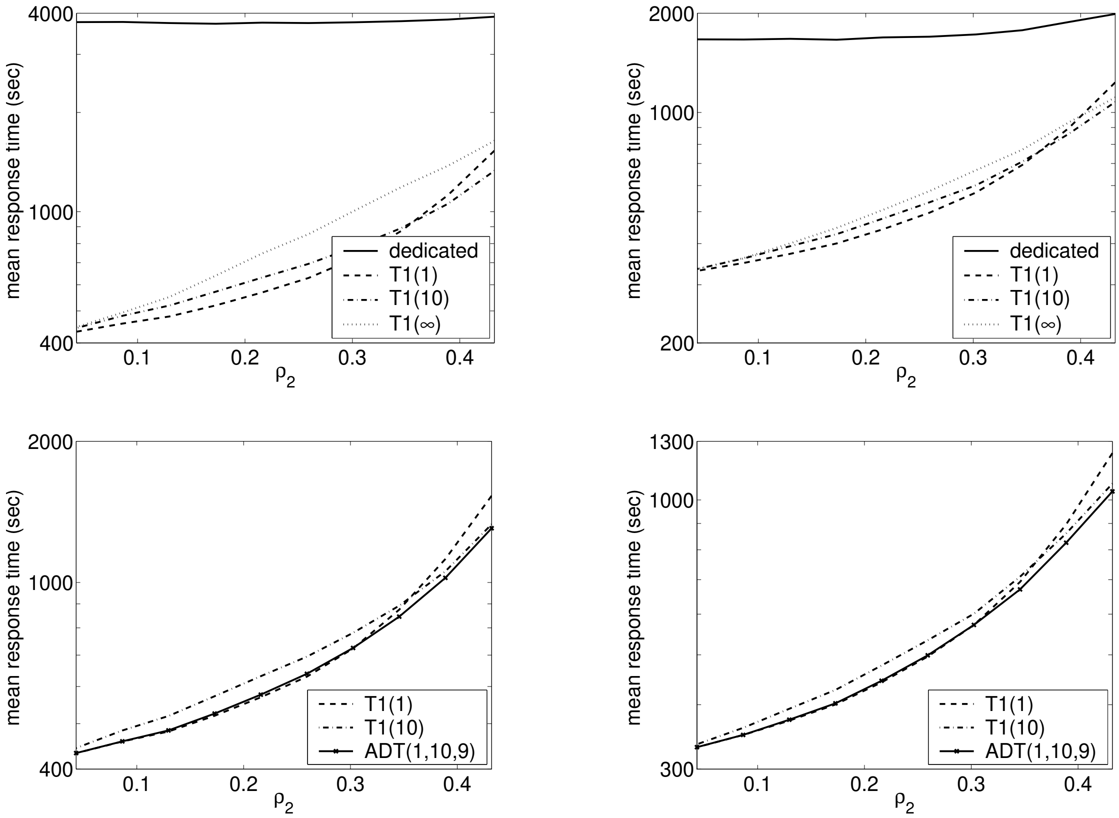
We study static robustness of these policies in Section 5.2.1, and dynamic robustness in Section 5.2.2.

5.2.1 Static robustness

Figures 15-16 illustrate static robustness of the Dedicated, T1, and ADT policies, plotting the mean response times as a function of ρ_2, under the Regular-IN model (Figure 15) and under the Priority model (Figure 16). In the Regular-IN model, ρ_2 ranges only between 0 and 0.43; no calls are removed from the trace at $\rho_2 = 0.43$. On the other hand, the Priority model allows us to change ρ_2 in a much wider range, and we show only a portion of the full range of ρ_2. In both the Regular-IN and Priority models, ρ_1 is chosen such that the mean response time of calls at queue 1 under Dedicated is about 60 minutes in column (a), i.e. $\rho_1 = 0.7$ in Regular-IN and $\rho_1 = 0.42$ in Priority, and about 30 minutes in column (b), i.e. $\rho_1 = 0.53$ in Regular-IN and $\rho_1 = 0.32$ in Priority. In both the Regular-IN and Priority models, the top row shows the mean response time under Dedicated, T1(1), T1(10), and T1(∞), and the bottom row shows the mean response time under T(1), T(10), and ADT with a different scale on the vertical axis.

The top rows of Figures 15-16 show that all of the T1 policies can significantly improve upon the Dedicated policy for a range of ρ_2 for both high and low ρ_1. In the case of the Regular-IN

Regular-IN model

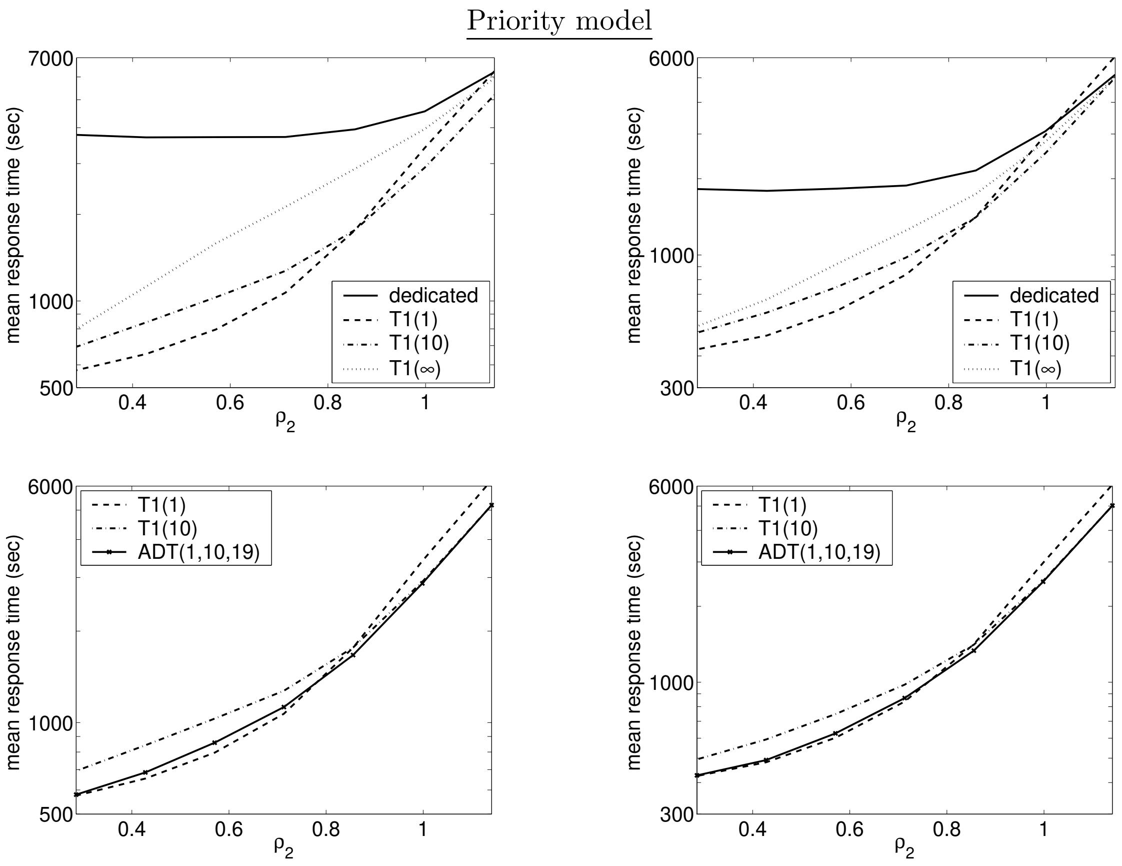


(a) $\rho_1 = 0.70$ (b) $\rho_1 = 0.53$

Figure 15: *Static robustness of Dedicated, T1, and ADT under the Regular-IN model.*

model, this improvement implies that resource sharing (T1) has a significant benefit over the original call center architecture (Dedicated) with respect to mean response time. The figures also show that the improvement of the T1 policies over Dedicated becomes smaller at higher ρ_2 and lower ρ_1. This makes intuitive sense, since the agent at queue 2 can help queue 1 less at higher ρ_2, and is needed less at lower ρ_1.

T1(∞) is the policy where the agent at queue 2 helps queue 1 only when there are no calls waiting at queue 2, and is equivalent to the T2(1) policy. Figures 15-16 (top rows) suggest that T1(∞) can significantly improve upon Dedicated, but its mean response time can be much higher than the T1 policy with the optimized t_1 threshold value. This is in agreement with what we have observed in Section 2.2: the mean response time under the T2 policy, including T2(1), is typically higher than that under the optimized T1 policy when queue 1 has higher $c\mu$ value ($c_1\mu_{12} > c_2\mu_2$).



(a) $\rho_1 = 0.42$ (b) $\rho_1 = 0.32$

Figure 16: *Static robustness of Dedicated, T1, and ADT under the Priority model.*

Taking a closer look, we see in Figure 16 (top row) that at very high ρ_2, the mean response time under T1(1) becomes higher than that under Dedicated. This is due to the smaller stability region of the T1 policy with small T1 threshold value, which is discussed in Section 2.1. Observe that in our parameter settings, T1(1) is equivalent to the $c\mu$ rule, as the $c\mu$ value is higher at queue 1 for both models. The loss of stability of the T1 policies is less clear in Figures 15-16 than, for example, in Figure 5, due to the fact that there are no calls after midnight at the call center, and thus all the calls are served eventually in our simulation settings.

Overall, Figures 15-16 (top rows) show that the T1 policy lacks static robustness. T1(1) provides low mean response time at lower ρ_2, but its mean response time becomes high (sometimes even higher than that under Dedicated) at higher ρ_2. On the other hand, T1(10) has higher mean response time than T1(1) at lower ρ_2, but it can provide lower mean response time at higher ρ_2.

Specifically, in Figures 15(a), the mean response time under T1(10) can be 10% worse than that under T1(1) at lower ρ_2, and the mean response time under T1(1) can be 10% worse than that under T1(10) at higher ρ_2. Likewise, in Figures 16(a), the mean response time under T1(10) can be 30% worse than that under T1(1) at lower ρ_2, and the mean response time under T1(1) can be 20% worse than that under T1(10) at higher ρ_2.

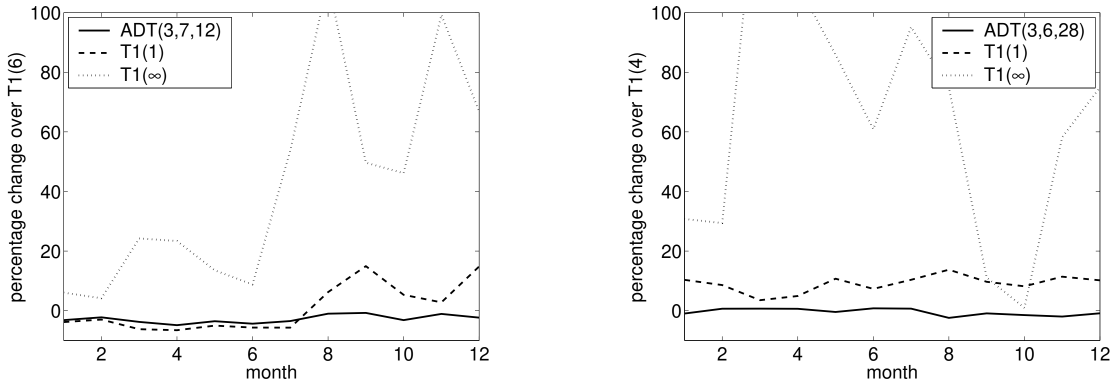
The bottom rows of Figures 15-16 illustrate static robustness of the ADT policy, where thresholds $t_1^{(1)} = 1$ and $t_1^{(2)} = 10$ are fixed and t_2 is chosen via our heuristic introduced in Section 3.2. The figures show that the ADT policy provides roughly at least as good mean response time as the better of the two T1 policies for the full range of ρ_2 and for both high and low ρ_1; i.e. the ADT policy excels in static robustness. This reinforces our findings in Section 3.2

The conclusion of our experiments is that resource sharing (T1) can significantly improve the mean response time at the call center. The ADT policy is an even better choice if the call center wants static robustness against changes in the number of calls in each day, for example, due to changes in service at the bank, increased patronage, or due to increased popularity of online transactions (which in turn leads to increased IN calls).

5.2.2 Dynamic robustness

In the previous section, we studied the effect of misestimation of the average load, by considering the mean response time of ADT and T1 at different average loads. To isolate the effect of dynamic robustness, we now hold the average load fixed, and study the effect of the load fluctuation inherent in our trace.

Figure 17 illustrates dynamic robustness of the T1 and ADT policies, plotting the percentage change in the mean response time each month against the optimized T1 policy. Recall that the trace has large fluctuations in the arrival rate within each day (Figure 13). As the arrival pattern in the trace is slightly different in each month, evaluating the mean response time each month allows us to evaluate dynamic robustness over twelve different arrival patterns. The threshold value, t_1, of the optimized T1 policy is chosen such that the overall mean response time during the year is minimized, and is fixed throughout the year. In the Regular-IN model Figure 17(a), T1(1), T1(∞), and ADT(3,7,12) are evaluated against the optimized T1 policy, T1(6). In the Priority model Figure 17(b), T1(1), T1(∞), and ADT(3,6,28) are evaluated against the optimized T1



(a) Regular-IN: $\rho_1 = 0.70$ and $\rho_2 = 0.37$ (b) Priority: $\rho_1 = 0.42$ and $\rho_2 = 0.82$

Figure 17: *Dynamic robustness of T1 and ADT under (a) the Regular-IN model and (b) the Priority model. Figures show the percentage change (%) in the mean response time of T1(1), T1(∞), and the (locally) optimized ADT policy over the T1 policy with the optimal t_1 threshold for the year: (a) $t_1 = 6$ and (b) $t_1 = 4$. A negative percentage indicates the* improvement *over the optimized T1 policy.*

policy, T1(4). Here, ADT(3,7,12) and ADT(3,6,28) are (locally) optimized ADT policies whose threshold values are chosen to minimize the mean response time during the year, as in Section 4. The loads, ρ_1 and ρ_2, are chosen such that the mean response time under Dedicated is roughly 60 minutes for both queue 1 and queue 2, but as Figures 15-16 suggest, mean response time is much lower under T1 and ADT policies.

Figure 17 shows that the mean response time under T1(∞) can be twice as high as the mean response time under the optimized T1 policy. Overall, in T1(∞) the agent at queue 2 is too conservative, and could help queue 1 more without penalizing calls at queue 2 too much.

Figure 17 also shows that the mean response time under T1(1) can be higher than the optimized T1 policy by 10-20% for some months. In the Regular-IN model, T1(1) is superior to the optimized T1 policy during the first seven months, but its mean response time becomes higher during the rest of the year. This is due to the fact that the number of IN calls per month increases throughout the year, and the T1(1) policy causes starvation at queue 2 during the peak hours. On the other hand, in the Priority model the T1(1) policy is consistently ($\sim$ 10%) worse than the optimized T1 policy. Overall, the mean response time under the T1(1) policy tends to be high, as it is likely to cause starvation at queue 2 at peak hours.

Finally, Figure 17 shows that the mean response time under the (locally) optimized ADT policy is slightly better than the optimized T1 policy, but parallel to the observation in Section 4, the performance advantage of ADT over T1 is small under load fluctuation. Specifically, the improvement of the optimized ADT policy over the optimized T1 policy is never more than 5% in the Regular-IN model and is never more than 2.5% in the Priority model.

The conclusion of our experiments is that the ADT policy yields only a small improvement over the T1 policy with respect to dynamic robustness. The small improvement suggests that using more thresholds may not improve the mean response time appreciably. Thus, with respect to minimizing the mean response time at the call center, the T1 policy suffices, even though the load at the call center has large fluctuations within each day. Note that these observations agree with our findings in Section 4.

6 Conclusion

This paper presents the first analytical study of the performance of a wide range of threshold based (resource) allocation policies in a multiserver system. The speed and accuracy of our analysis allow an extensive evaluation of these allocation policies, and we find surprising conclusions.

We first consider single threshold policies, T1 and T2, and find that the T1 policy is superior with respect to (overall weighted) mean response time. That is, the threshold for resource allocation is better determined by the beneficiary queue length (queue 1) than by the donor queue length (queue 2), in all cases studied.

We then compare single threshold policies to a multiple threshold policy, the adaptive dual threshold (ADT) policy, with respect to mean response time, assuming that the load is fixed and known. We find that when the threshold value is chosen appropriately, the mean response time of the T1 policy is at worst very close to the best mean response time achieved by the ADT policy. This is surprising, since the optimal policy appears to have infinitely many thresholds, but evidently the improvement these thresholds generate is marginal.

We next study static robustness, where the load is constant, but may have been misestimated. We find that the ADT policy not only provides low mean response time but also excels in static robustness, whereas the T1 policy does not. The increased flexibility of the ADT policy enables it to provide low mean response time under a range of loads. Hence, when the load is not exactly known, the ADT policy is a much better choice than the T1 policy.

Finally, and surprisingly, our analysis shows that this improvement in static robustness does not necessarily carry over to dynamic robustness, i.e. robustness against load fluctuation. We observe that when the load is fluctuating, the T1 policy, which lacks static robustness, can often provide low mean response time, comparable to ADT. This can occur for example because the mean response time of jobs arriving during the high load period may dominate the overall mean response time.

Complementing our analytical work, we evaluate the performance of various allocation policies by using a trace from a call center. The arrival pattern at the call center exhibits quite large fluctuation; our trace driven simulation reinforces our conclusions, implying that this call center could significantly improve mean response time using the T1 policy, and the ADT policy has only a small improvement over the T1 policy with respect to dynamic robustness. However, the ADT policy is a better choice if the call center wants static robustness.

Acknowledgement

This work is supported by NSF Career Grant CCR-0133077, NSF Theory CCR-0311383, NSF ITR CCR-0313148, and IBM Corporation via Pittsburgh Digital Greenhouse Grant 2003. The authors also thank Li Zhang, who contributed to an earlier version of this paper (Osogami et al., 2004).

References

H. S. Ahn, I. Duenyas, and R. Q. Zhang. Optimal control of a flexible server. *Advances in Applied Probability*, 36:139-170, 2004.

S. L. Bell and R. J. Williams. Dynamic scheduling of a system with two parallel servers in heavy traffic with complete resource pooling: Asymptotic optimality of a continuous review threshold policy. *Annals of Applied Probability*, 11:608-649, 2001.

L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, and L. Zhao. Statistical analysis of a telephone call center: A queueing-science perspective. *Journal of the American Statistical Association*, 100(469):36-50, 2005.

D. R. Cox and W. L. Smith. *Queues.* Kluwer Academic Publishers, 1971.

L. Green. A queueing system with general use and limited use servers. *Operations Research*, 33(1):168-182, 1985.

I. Guedj and A. Mandelbaum. “Anonymous bank” call-center data, February 2000. http://ie.technion.ac.il/~serveng/.

J. M. Harrison. Heavy traffic analysis of a system with parallel servers: Asymptotic optimality of discrete review policies. *Annals of Applied Probability*, 8(3):822-848, 1998.

G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modeling.* ASA-SIAM, Philadelphia, 1999.

A. Mandelbaum and A. Stolyar. Scheduling flexible servers with convex delay costs: Heavy traffic optimality of the generalized $c\mu$-rule. *Operations Research*, 52(6):836-855, 2004.

S. Meyn. Sequencing and routing in multiclass queueing networks part i: Feedback regulation. *SIAM Journal on Control Optimization*, 40(3):741-776, 2001.

T. Osogami. Analysis of multi-server systems via dimensionality reduction of Markov chains. PhD thesis, School of Computer Science, Carnegie Mellon University, 2005. http://www.cs.cmu.edu/~osogami/thesis/

T. Osogami, M. Harchol-Balter, A. Scheller-Wolf, and L. Zhang. Exploring threshold-based policies for load sharing. In *Proceedings of the 42nd Annual Allerton Conference on Communication, Control, and Computing*, pages 1012-1021, September 2004.

R. Shumsky. Approximation and analysis of a call center with specialized and flexible servers. *OR Spektrum*, 26(3):307-330, 2004.

M. S. Squillante, C. H. Xia, D. D. Yao, and L. Zhang. Threshold-based priority policies for parallel-server systems with affinity scheduling. In *Proceedings of the IEEE American Control Conference*, pages 2992-2999, June 2001.

M. S. Squillante, C. H. Xia, and L. Zhang. Optimal scheduling in queuing network models of high-volume commercial web sites. *Performance Evaluation*, 47(4):223-242, 2002.

D. A. Stanford and W. K. Grassmann. The bilingual server system: A queueing model featuring fully and partially qualified servers. *INFOR*, 31(4):261-277, 1993.

D. A. Stanford and W. K. Grassmann. Bilingual server call centers. In D.R. McDonald and S.R.E. Turner, editors, *Analysis of Communication Networks: Call Centers, Traffic and Performance*. American Mathematical Society, 2000.

R. A. Uhlig and T. N. Mudge. Trace-driven memory simulation: A survey. *ACM Computing Surveys*, 29(2):128-170, 1997.

J. A. Van Mieghem. Dynamic scheduling with convex delay costs: The generalized $c\mu$ rule. *Annals of Applied Probability*, 5(3):809-833, 1995.

R. J. Williams. On dynamic scheduling of a parallel server system with complete resource pooling. In D. R. McDonald and S. R. E. Turner, editors, *Analysis of Communication Networks: Call Centers, Traffic and Performance*. American Mathematical Society, 2000.

A Analysis of the ADT policy

A.1 Analysis of the ADT policy under Poisson arrivals

Below we provide a formal description of the analysis of the ADT policy; this will be useful when we generalize the arrival process. The Markov chain in Figure 8 is a (nonhomogeneous) QBD process, where level j of the process denotes the j-th column, namely all states of the form (i, j) for each j. The generator matrix, $\mathbf{Q}$, of this process can be expressed as a block diagonal matrix:

$$\mathbf{Q} = \begin{pmatrix} \mathbf{L}^{(0)} & \mathbf{F}^{(0)} & & & \\ \mathbf{B}^{(1)} & \mathbf{L}^{(1)} & \mathbf{F}^{(1)} & & \\ & \mathbf{B}^{(2)} & \mathbf{L}^{(2)} & \mathbf{F}^{(2)} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}$$

where submatrix $\mathbf{F}^{(i)}$ encodes transitions from level (column) i to level $i+1$ for $i \geq 0$, submatrix $\mathbf{B}^{(i)}$ encodes transitions from level i to level $i-1$ for $i \geq 1$, and submatrix $\mathbf{L}^{(i)}$ encodes transitions within level i for $i \geq 0$. Note that our QBD process repeats after level t_2+1, i.e., $\mathbf{F}^{(i)} = \mathbf{F}^{(t_2+1)}$, $\mathbf{L}^{(i)} = \mathbf{L}^{(t_2+1)}$, and $\mathbf{B}^{(i)} = \mathbf{B}^{(t_2+1)}$ for all $i > t_2+1$.

Before dimensionality reduction, matrices $\mathbf{F}^{(i)}$, $\mathbf{L}^{(i)}$, and $\mathbf{B}^{(i)}$, had an infinite number of columns and rows, since the number of states in each level was infinite (recall Figure 3(a)). DR reduces the number of states in each level to a finite number; as a result, matrices $\mathbf{F}^{(i)}$, $\mathbf{L}^{(i)}$, and $\mathbf{B}^{(i)}$, have a finite number of columns and rows. Again, in Figure 8, the bottom two rows correspond to the "busy period", during which there are $\geq t_1^{(2)}$ type 2 jobs, approximated by a two phase PH distribution with parameters $(\beta_1, \beta_2, \beta_{12})$.

Assuming $1 < t_1^{(1)} < t_1^{(2)}$, and that the "busy period" is approximated by a *two* phase PH distribution, matrices $\mathbf{F}^{(i)}$, $\mathbf{L}^{(i)}$, and $\mathbf{B}^{(i)}$ have size $(t_1^{(2)}+2) \times (t_1^{(2)}+2)$ for all i. Specifically,

$$\mathbf{F}^{(i)} = \lambda_2 \mathbf{I},$$

for $i \geq 0$, where $\mathbf{I}$ is an identity matrix of an appropriate size;

$$\mathbf{B}^{(i)} = \mu_2 \begin{pmatrix} \mathbf{I} & \\ & \mathbf{0} \end{pmatrix}$$

where $\mathbf{I}$ is an identity matrix of size $t_1^{(1)}$ for $1 \leq i \leq t_2$ and of size $t_1^{(2)}$ for $i > t_2$, and $\mathbf{0}$ is a zero matrix of an appropriate size;

$$\mathbf{L}^{(i)} = \begin{pmatrix} (*) & \lambda_1 & & & & & & \\ \mu_1 & (*) & \lambda_1 & & & & & \\ & \mu_1+\mu_{12} & (*) & \ddots & & & & \\ & & \ddots & \ddots & \lambda_1 & & & \\ & & & \mu_1+\mu_{12} & (*) & \lambda_1 & 0 \\ & & & & \beta_1 & -(\beta_1+\beta_{12}) & \beta_{12} \\ & & & & \beta_2 & 0 & -\beta_2 \end{pmatrix}$$

for all $1 \leq i \leq t_2$, where the diagonal elements, $(*)$, are determined so that the sum of each row in the generator matrix $\mathbf{Q}$ becomes zero. Matrix $\mathbf{L}^{(0)}$ is obtained from $\mathbf{L}^{(1)}$ by replacing the (2,1) element by $\max\{\mu_1, \mu_{12}\}$. For $i > t_2$, $\mathbf{L}^{(i)}$ is obtained from $\mathbf{L}^{(1)}$ by replacing the $(k+1, k)$ element (i.e., $\mu_1 + \mu_{12}$) by μ_1 for $3 \leq k \leq t_2$.

Using matrix analytic methods, the stationary probability of being in level i, $\overrightarrow{\pi_i}$, is then given recursively by

$$\overrightarrow{\pi_i} = \overrightarrow{\pi_{i-1}} \mathbf{R}^{(i)},$$

where $\mathbf{R}^{(i)}$ is given recursively by:

$$\mathbf{F}^{(i-1)} + \mathbf{R}^{(i)}\mathbf{L}^{(i)} + \mathbf{R}^{(i)}\mathbf{R}^{(i+1)}\mathbf{B}^{(i+1)} = \mathbf{0},$$

for $i = t_2, ..., 1$, where $\mathbf{0}$ is a zero matrix of an appropriate size. Since our QBD process repeats after level $t_2 + 1$, $\mathbf{R}^{(i)} = \mathbf{R}$ for all $i \geq t_2 + 1$, where $\mathbf{R}$ is given by the minimal solution to the following matrix quadratic equation:

$$\mathbf{F}^{(t_2+1)} + \mathbf{R}\mathbf{L}^{(t_2+1)} + (\mathbf{R})^2\mathbf{B}^{(t_2+1)} = \mathbf{0}.$$

A row vector $\vec{\pi_0}$ is given by a positive solution of

$$\vec{\pi_0}\left(\mathbf{L}^{(0)} + \mathbf{R}^{(1)}\mathbf{B}^{(1)}\right) = \vec{0},$$

normalized by

$$\vec{\pi_0}\sum_{i=0}^{\infty}\prod_{k=1}^{i}\mathbf{R}^{(k)}\vec{1} = 1, \tag{3}$$

where $\vec{0}$ and $\vec{1}$ are vectors with an appropriate number of elements of 0 and 1, respectively. Note that the infinite sum in (3) may be rewritten by a closed form expression using $(\mathbf{R})^{-1}$, since $\mathbf{R}^{(i)} = \mathbf{R}$ for all $i \geq t_2 + 1$. The mean response time can be computed using the stationary distributions ($\vec{\pi_i}$'s) given above, via Little's law, as before.

A.2 Analysis of the ADT policy under MMPP

In this section, we describe the analysis of the ADT policy when arrivals at queue 2 follow an MMPP and arrivals at queue 1 follow a Poisson process. An extension to the case where arrivals at queue 1 also follow an MMPP is possible, but requires an additional technique (see e.g. Osogami, 2005). Following standard notation, we denote the parameters of an MMPP by a pair of matrices $(\mathbf{D_0}, \mathbf{D_1})$. In the case of MMPP(2),

$$\mathbf{D_0} = \begin{pmatrix} -(\alpha_H + \lambda_H) & \alpha_H \\ \alpha_L & -(\alpha_L + \lambda_L) \end{pmatrix} \quad \text{and} \quad \mathbf{D_1} = \begin{pmatrix} \lambda_H & 0 \\ 0 & \lambda_L \end{pmatrix},$$

where the duration of the high (low, respectively) load period has an exponential distribution with rate α_H (α_L, respectively). Let λ denote the fundamental rate (the overall average arrival rate) of the MMPP.

The generator matrix, $\hat{\mathbf{Q}}$, when the arrivals at queue 2 follow an MMPP with parameter $(\mathbf{D_0}, \mathbf{D_1})$ is obtained by modifying the generator matrix, $\mathbf{Q}$, for the Poisson arrivals, which we introduced in Section A.1. We denote the submatrices of $\hat{\mathbf{Q}}$ by $\hat{\mathbf{F}}^{(i)}$, $\hat{\mathbf{L}}^{(i)}$, and $\hat{\mathbf{B}}^{(i)}$, i.e.,

$$\hat{\mathbf{Q}} = \begin{pmatrix} \hat{\mathbf{L}}^{(0)} & \hat{\mathbf{F}}^{(0)} & & & \\ \hat{\mathbf{B}}^{(1)} & \hat{\mathbf{L}}^{(1)} & \hat{\mathbf{F}}^{(1)} & & \\ & \hat{\mathbf{B}}^{(2)} & \hat{\mathbf{L}}^{(2)} & \hat{\mathbf{F}}^{(2)} & \\ & & \ddots & \ddots & \ddots \end{pmatrix}.$$

Then, $\hat{\mathbf{F}}^{(i)}$, $\hat{\mathbf{L}}^{(i)}$, and $\hat{\mathbf{B}}^{(i)}$ are obtained from the submatrices, $\mathbf{F}^{(i)}$, $\mathbf{L}^{(i)}$, and $\mathbf{B}^{(i)}$, of $\mathbf{Q}$ by

$$\begin{aligned} \hat{\mathbf{F}}^{(i)} &= \mathbf{D_1} \otimes \mathbf{F}^{(i)}/\lambda \\ \hat{\mathbf{B}}^{(i)} &= \mathbf{I_0} \otimes \mathbf{B}^{(i)} \\ \hat{\mathbf{L}}^{(i)} &= \mathbf{D_0} \otimes \mathbf{I_i} + \mathbf{I_0} \otimes \mathbf{L}^{(i)} \end{aligned}$$

for all i. Here, $\mathbf{I_i}$ denotes an identity matrix of the same size as $\mathbf{L}^{(i)}$, $\mathbf{I_0}$ denotes an identity matrix of the same size as $\mathbf{D_0}$, and $\otimes$ denotes the Kronecker product.

Note that, before DR, $\mathbf{F^{(i)}}$ $\mathbf{L^{(i)}}$, and $\mathbf{B}^{(i)}$ have an infinite size, and so do $\hat{\mathbf{F}}^{(i)}$ $\hat{\mathbf{L}}^{(i)}$, and $\hat{\mathbf{B}}^{(i)}$. Dimensionality reduction reduces the size of $\mathbf{F}^{(i)}$, $\mathbf{L}^{(i)}$, an $\mathbf{B}^{(i)}$ to finite, and thus the size of $\hat{\mathbf{F}}^{(i)}$, $\hat{\mathbf{L}}^{(i)}$, an $\hat{\mathbf{B}}^{(i)}$ also becomes finite. Now, the mean response time is obtained in the same way as in Section A.1.