Managing Response Time in a Call Routing Problem with Service Failure

Francis de Véricourt* Yong-Pin Zhou†

October 2004

Abstract

Traditional research on routing in queueing systems usually ignores service quality related factors. In this paper, we analyze the routing problem in a system where customers call back when their problems are not completely resolved by the service customer representatives (CSRs). We introduce the concept of call resolution probability, and we argue that it constitutes a good proxy for call quality. For each call, both the call resolution probability (p) and the average service time ($1/\mu$) are CSR dependent. We use an MDP formulation to obtain analytical results and insights about the optimal routing policy that minimizes the average total time of call resolution including callbacks. In particular, we provide sufficient conditions under which it is optimal to route to the CSR with the highest call resolution rate ($p\mu$) among those available. We also develop efficient heuristics that can be easily implemented in practice.

*Fuqua School of Business, Duke University, fdv1@duke.edu

†University of Washington Business School, Seattle, yongpin@u.washington.edu

1 Introduction

Customer service oriented call centers are traditionally operated as cost centers. Service accessibility and customer waiting time are the dominant performance measures. As a result, capacity planning and call routing software systems strive to minimize costs while achieving self-imposed service level constraints, such as "average wait in queue less than 15 seconds". These traditional approaches do not consider, however, the quality of answers provided by the Customer Service Representatives (CSRs). Low quality of service has a significant impact on the call center operations besides customer defection: As dissatisfied customers call back for more help for the same problem, the load on the system increases.

This operational impact of service failure is often ignored by call center capacity planning and call routing management systems. Our paper is motivated by the problems at a major European telecommunication service provider, which found that, on average, a customer needed to talk to more than three different CSRs to get his/her problem resolved. This company also observed noticeable differences among CSRs in their ability to resolve the customers' problems. In our paper, we integrate this service quality related information into call routing decisions. The goal is to minimize the average total time of call resolution, defined as the total time spent by a customer in the system to resolve one issue, including all the callbacks.

A key feature of our approach is the way we model the quality of a CSR's answer. For customer service call centers, a high-quality answer provided by the CSR should resolve the customer's issue during that call. We operationalize this concept by defining call quality as the call resolution probability, the probability that the customer is satisfied and does

not call back for the same problem. The call resolution probability is directly related to a customer's perception of call quality which depends on the CSR's understanding of the customer's needs, courtesy, and competency (Zeithaml *et al.* [34]). Furthermore, it can be quantified and measured by most of the call center information systems in use today.

Our experience suggests that a CSR's call resolution probability is often highly correlated with his/her call speed (defined as the service rate). On the one hand, the correlation could be negative. Due to very high turnover rates and long training lead-time in this industry (see Gans and Zhou [11] for example), some call centers are pressed to make the most use of their CSRs. It is common for the call center to compensate CSRs on the number of calls served over a period of time, or their call handle time, thereby encouraging them to handle calls as fast as possible. As a result, CSRs sometimes rush to end a call without making sure that the root problem is fixed and will not re-occur later (Read [28]). On the other hand, the correlation could be positive. Many times, better trained and more experienced CSRs are able to handle the calls faster and provide higher service quality at the same time. In this paper, we model the service time and the call resolution probability as exogenous variables, and we do not explicitly model the correlation between them.

Intuitively, call centers that deal with complex issues, such as technical support for corporate computer users or medical help over the phone, may have low call resolution probabilities. Nevertheless, we know from our experience that even when customer problems are simple (as for the European call center on which this study is based), the call resolution probabilities can be significantly low. We believe that this comes from the CSR compensation system mentioned previously and the high turnover rate which results in under-trained employees. In this paper, we describe routing rules that account for these call resolution

probabilities. Although we do not directly identify compensation schemes that can improve the call resolution probabilities while addressing the high turnover rate, our results provide interesting insights into this issue.

We analyze a call routing problem where there exist several classes of CSRs, each with its own average call speed μ and call resolution probability p. The goal is to minimize the average total time of call resolution. Potentially there is a tradeoff between call speed and call resolution in routing calls: If call resolution is the only concern then it would be optimal to route calls only to the CSR class with the highest p. The customers' wait, however, may become excessively long. If call speed is the only concern, then the objective would be to minimize the average waiting time of each call instance independently, without paying attention to the number of customer attempts. Hence, we feel the average total time of call resolution is the best single measure that encompasses both call speed and call resolution, and it can be construed as the average number of customer tries times the average waiting time of each try. Other objective functions, such as linear combinations of call resolution and call speed, are possible, but the weights are hard to determine and they generally lead to intractable models.

We formulate the routing problem as a Markov Decision Process (MDP), where the call center is represented by a heterogeneous, multi-server queueing system. In this framework, we provide several partial characterizations of the optimal routing policy. Our main result states that, whenever possible, a call should be routed to the CSR class with the highest call resolution rate, $p\mu$. If the highest-$p\mu$ CSRs are all busy, then the call may be routed to another available CSR or kept in the queue. Furthermore, we derive sufficient conditions under which it is optimal to route a call to the CSR with the highest resolution rate *among*

the available CSRs. We call this the $p\mu$ rule. In particular, we show that when the CSRs differ only in their call speed or call resolution probability, the $p\mu$ rule is optimal. We also fully characterize the optimal routing policy for a system with two heterogeneous CSRs. In this case, we show that the optimal policy is of a threshold type: a call will always be routed to the CSR with the higher resolution rate, whenever possible; the other CSR will be routed a call only when the number of calls waiting in queue exceeds a certain threshold.

Based on these findings, we propose simple and intuitive routing policies. Our numerical studies show that the $p\mu$ rule performs very well in most cases even when it is not optimal. Moreover, the $p\mu$-t policy, defined as the $p\mu$ rule plus a threshold, is almost optimal in all of our test cases. We also numerically demonstrate that call centers can significantly improve their performance by incorporating call resolution probability p into routing decisions.

The $p\mu$ index introduced in this paper is a simple and effective routing index that accounts for both the call speed and the call quality. It also suggests that CSRs should be evaluated and compensated on their call resolution rate, rather than their service rate alone, as is often the case.

To ascertain the robustness of our findings, we analyze the problem under more general modeling assumptions. We show that our results remain valid when callbacks are put in a separate queue and given priority. We also show numerically that the $p\mu$-based policies perform well even if there is an exponentially distributed delay before a customer calls back. When the service time depends on whether, and how many times, the customer has talked to the same CSR before, we introduce and evaluate a dedicated routing policy, which routes new calls using the $p\mu$-t policy, but always routes callbacks to the same CSR. A requirement for the implementation of this policy is the call center's ability to identify the history of a

call before serving it (e.g. a case number is required for callbacks at the phone prompt), which is not the case for the call center we study. In Section 5.4 , we will study the dedicated policy as an extension to the basic model.

The rest of the paper is organized as follows: In Section 2, we review the literature, and in Section 3, we formulate and discuss the model. Results for the optimal routing policy are presented in Section 4. In Section 5 we use extensive numerical tests to show the importance of accounting for call resolution probability in making the routing decisions. Several heuristics are proposed and compared. We also analyze the problem when some modeling assumptions are relaxed. We conclude the paper and comment on further research in Section 6.

2 Literature Review

The probability of health deterioration after treatment in the health care system (e.g. Berk and Moinzadeh [4] and De Angelis [1]), which is a strong indicator of the treatment efficiency, is similar to the probability of callback, $1-p$, in our model. To our knowledge, however, our paper is the first to apply such a measure of quality to the research of call centers or other service delivery systems.

If calls bring direct revenue to the company (e.g., catalog merchant), customer loyalty, measured by the probability of defection, better reflects the service quality provided by the call center. Hall and Porteus [14] and Gans [9] are two examples of this approach. In this paper we focus on the customer service call centers, so we assume that dissatisfied customers will call back, instead of simply defecting. Furthermore, to address customer allocation and

capacity planning problems, we use a more detailed model of the service system than those in [14] and [9].

There is a large body of literature on the retrial queues. See Falin and Templeton [8] and the references therein. More recently, Mandelbaum *et al.* (e.g. [23], [24], [22]) study the effect of customer retrial behavior patterns specifically in the context of call centers. The customer retrials they study differ from the customer callbacks in this paper in that a retrial occurs *before* the customer receives her service (when a customer calls and receives a busy signal, she "retries" by calling back sometime later), while a callback occurs *after* the customer has already received her service.

When the CSRs in a call center have different skills and speeds, skills-based routing has been shown to outperform the FIFO and first-available-CSR call routing rules in many situations. As a result, much study has been done on the skills-based call routing schemes, both in the industry and in academia (e.g. Bell and Williams [3], Harrison and Lopéz [15], Gans and Zhou [12], Atar, Mandelbaum, and Reiman [2], and some other references contained in Gans, Koole, and Mandelbaum [10]).

Research on routing in general often suggests priority-based policies: some call-CSR combinations are given priority so they will be used whenever possible; the other combinations will be used only if the system is in certain states. A good example is the traditional $c\mu$ rule (see Van Mieghem [31] for a generalized $c\mu$ rule and Mandelbaum and Stolyar [25] for its application in the call center setting). The main issue in these models is how to minimize total cost based on the different processing speeds associated with each call-CSR combination and the different call-type specific holding costs.

The stream of research most relevant to ours is the so-called slow server problem. In the

two-server slow server problem, there is one Poisson arrival stream and two heterogeneous exponential servers. The objective is to find a routing policy to minimize the average wait. Larsen [18] first formulates the problem and conjectures that a threshold policy should be optimal. Later, Lin and Kumar ([20]), Walrand ([33]), and Koole ([17]) prove this conjecture using MDP policy iteration, coupling argument, and MDP value iteration, respectively. Larsen and Agrawala [19] develop a good and computationally simple approximation to the threshold.

The general slow server problem allows for more than 2 heterogeneous servers. Due to the increase in state space dimensionality, the problem becomes very complex (e.g. see Rykov [29] and Luh and Viniotis [21]). So far, the optimal routing policy has not been fully characterized for the general case (see de Véricourt and Zhou [7]). Our model can be viewed as the general slow server model with multiple classes of servers and the additional callback loops - in particular, when the call resolution probabilities are all equal to one, our model reduces to the general slow server problem. The optimality of the $p\mu$ rule in our model implies that allocating a call to the fastest server (the μ rule) is optimal for the general slow server problem. This extends the existing literature on the general slow server problem.

Most analysis of the slow server problem is exact. Teh and Ward [30], on the other hand, studies the problem in the heavy-traffic regime. They show that, as the heavy traffic limit is approached, the system is stable and the threshold policy is optimal if and only if the threshold grows at a logarithmic rate. In other words, in the heavy traffic regime, the threshold does not disappear.

3 Formulation of the Problem

3.1 Model and Assumptions

Consider a call center with C classes of CSRs. A class is a group of CSRs with the same service time distribution and call resolution probability. We assume that there are S_i CSRs in class i, $i \in \{1, \ldots, C\}$. For a Class-i CSR, $i \in \{1, \ldots, C\}$, the service time is exponentially distributed with rate μ_i, and the call resolution probability is p_i. When a Class-i CSR completes a call, there are two possible outcomes: 1) with probability p_i, the issue is completely resolved and the customer will simply leave the system; and 2) with probability $1 - p_i$, the issue is not completely resolved, and the customer calls back right away.

Our model does not differentiate new calls from callbacks and all customers are served on a first-come-first-served basis. In practice, however, callbacks are sometimes given higher priority if they can be identified. This means that callbacks are put in a separate queue and given priority over new calls. A simple coupling argument shows nonetheless that such a priority scheme does not alter the average total waiting time of the system and our findings remain valid.

The arrival of customers with new requests follows a Poisson process with rate λ, and they wait in a queue if they are not served upon arrival. There is no limit on the waiting space. To ensure stability, we assume that $\lambda < \sum_{i=1}^{C} S_i p_i \mu_i$. See Figure 1 for details.

Due to the memoryless property of Poisson arrival and exponential service times, the state of the system at time t can be described by a $(C+1)$-dimensional vector $\mathbf{n}(t) = (n_0(t), n_1(t), \ldots, n_C(t))$, where $n_0(t) \geq 0$ is the number of calls waiting in the queue and $n_i(t) \in \{0, \ldots, S_i\}$, $i \in \{1, \ldots, C\}$ is the number of busy Class-i CSRs.

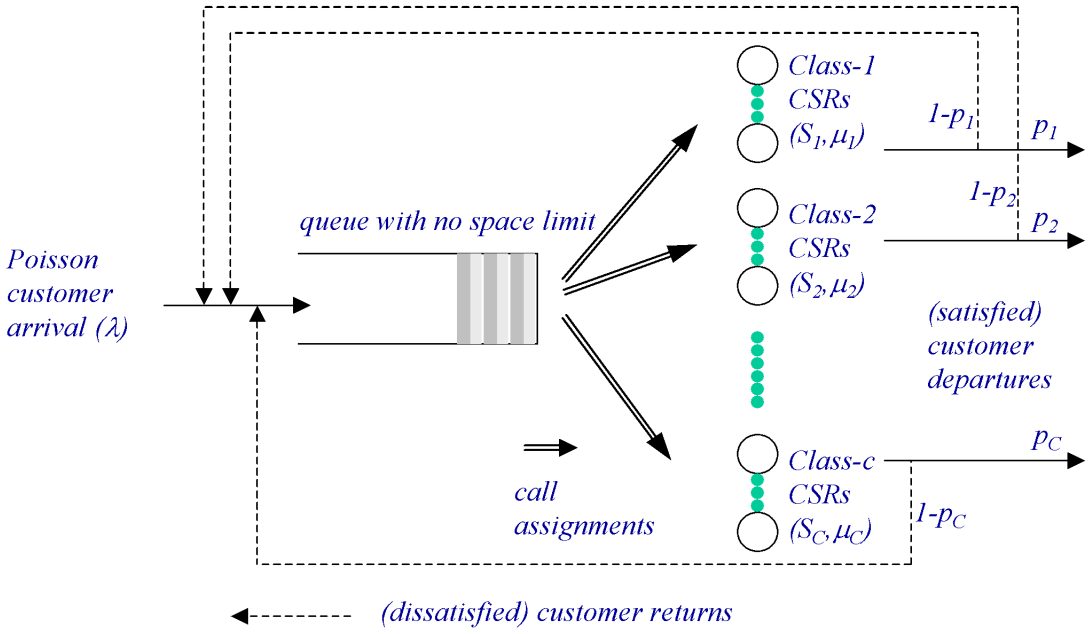

Figure 1: Model Overview

At any time, the system controller must decide 1) whether to keep a call in the queue or to route it to an available CSR, and 2) if a call is to be routed, which CSR class it should be routed to. The goal of our model is to minimize the average total time of call resolution.

In this model, we assume that both the call resolution probabilities and the service rates are independent of the number of previous calls made by the customer for the same problem. Such an assumption may not be realistic in certain situations, for instance when there is a setup time each time a customer meets a new CSR, or more generally when the service time decreases with the number of attempts. We discuss this situation in Section 5.3.

We also assume customers *immediately* return to the system when they are dissatisfied. This assumption is reasonable when a customer can quickly check the accuracy of the CSR's answer. Examples include technical support call centers that deal with computer hardware/software applications, where the delay in callback is usually small compared to the service time. In other practical situations, however, dissatisfied customers call back after

a longer delay. In Section 5.4, we present a model where an exponential amount of time elapses before dissatisfied customers call back. Numerical studies show that the routing policies developed for the immediate callback model also perform well in this case.

3.2 The Markov Decision Problem

The routing policies we study are non-anticipating and non-preemptive. Furthermore, due to the Markovian assumptions, the policies are also not history dependent. As is well known in the literature, it is optimal to take actions only at arrival and service departure epochs. Any possible action is represented by a C-dimensional vector $(a_1, \ldots, a_C)$, where $a_i, \forall i \in \{1, \ldots, C\}$, is the number of calls routed to Class-i CSRs. In particular, the zero vector represents the (non-)action of not routing any call. A routing policy π is thus a rule that determines, for every decision epoch, what action to take.

The objective is to determine the routing policy that minimizes the average total time of call resolution. By Little's Law, this is equivalent to minimizing the average number of customers in the system. As a result, we look for the Markov routing policies that minimize the average number of customers in the system:

$$g^* = \min_{\pi} \quad \limsup_{T\to\infty} \frac{1}{T} E^{\pi}_{\mathbf{n}_0} \left[\int_0^T \sum_{i=0}^{C} \mathbf{n}_i(t) dt \right], \tag{1}$$

where $E^{\pi}_{\mathbf{n}_0}$ denotes the conditional expectation given policy π and the initial state at time 0.

The main approach we use in this paper is the standard MDP value iteration (e.g. see Ha [13] or Veatch and Wein [32]). Let $v(\mathbf{n})$ be the standard MDP "cost-to-go" function in state $\mathbf{n}$, then v is a mapping from $\mathbf{N}^{C+1}$ to $\Re^+$, where $\mathbf{N}$ and $\Re^+$ are the sets of integers and non-negative real numbers respectively. In the next section, we will define the desirable

properties for the optimal MDP value function $v(\cdot)$ and show that these properties are preserved by the value iteration operators. We define below the two value iteration operators T and Γ.

Because the inter-arrival and service time are exponentially distributed, we can study an equivalent Markov process with *i.i.d* inter-event time, by adding fictitious transitions. This procedure is known as *uniformization*. (See Section 11.5 in [27] for details.) The uniformized Markov process will have a fixed total transition rate of $\lambda + \sum_{i=1}^{C} S_i\mu_i$ in every state. Without loss of generality, we can scale the time and assume that $\lambda + \sum_{i=1}^{C} S_i\mu_i = 1$. Let $\mathbf{e}_i$, $i \in \{0, \ldots, C\}$, denote a $(C+1)$-dimension vector whose $(i+1)$th component is 1 and all other components 0, $\Omega = \{f | f : \mathbf{N}^{C+1} \to \Re^+\}$, and $T : \Omega \to \Omega$.

We denote by $K(\mathbf{n})$ the set of classes with available CSRs in state $\mathbf{n}$:

$$K(\mathbf{n}) = \{i \in \{1, \ldots, C\} \mid n_i < S_i\}. \tag{2}$$

Then for any $v \in \Omega$,

$$Tv(\mathbf{n}) = \begin{cases} v(\mathbf{n}) & \text{if } n_0 = 0 \text{ or } K(\mathbf{n}) = \emptyset, \\ \min\{Tv(\mathbf{n} + \mathbf{e}_j - \mathbf{e}_0), v(\mathbf{n}) \mid j \in K(\mathbf{n})\} & \text{if } n_0 > 0 \text{ and } K(\mathbf{n}) \neq \emptyset. \end{cases} \tag{3}$$

Note that more than one calls can be routed at once. Instead of listing all these possible routings in the minimization operator, we choose to use equivalent recursive definition in (3). The recursion is well defined because n_0 decreases by one each time a call is routed. For example, take $n_0 = 2$, and apply the previous recursive definition twice. We have

$$Tv(\mathbf{n}) = \min\{v(\mathbf{n} + \mathbf{e}_j + \mathbf{e}_k - 2\mathbf{e}_0), v(\mathbf{n} + \mathbf{e}_j - \mathbf{e}_0), v(\mathbf{n}) \mid j \in K(\mathbf{n}), k \in K(\mathbf{n} + \mathbf{e}_j)\}.$$

When $Tv(\mathbf{n}) = v(\mathbf{n} + \mathbf{e}_j + \mathbf{e}_k - 2\mathbf{e}_0)$ for some $j, k \in K(\mathbf{n})$, the corresponding policy routes two calls at the same time.

The MDP optimality equations then becomes

$$\Gamma v^*(\mathbf{n}) \;=\; v^*(\mathbf{n}) + g^*, \tag{4}$$

where g^* is the optimal average number of customers defined in (1) which is independent of the initial state (see for instance Rykov (2002) and the references therein), $v^*(\mathbf{n})$ is the optimal relative value function, and $\Gamma : \Omega \to \Omega$ is the dynamic operator that satisfies:

$$\begin{aligned}\Gamma v(\mathbf{n}) \;=\; & \sum_{i=0}^{C} n_i + \lambda T v(\mathbf{n}+\mathbf{e}_0) + \sum_{i=1}^{C} n_i(1-p_i)\mu_i T v(\mathbf{n}-\mathbf{e}_i+\mathbf{e}_0) \\ & + \sum_{i=1}^{C} n_i p_i \mu_i T v(\mathbf{n}-\mathbf{e}_i) + \sum_{i=1}^{C}(S_i - n_i)\mu_i T v(\mathbf{n}).\end{aligned} \tag{5}$$

Note that the last term corresponds to a fictitious transition due to the uniformization procedure. We allow an action at these transitions, as in Koole [17].

4 Analysis of the Optimal Routing Policy

In this section, we present partial characterizations of the optimal routing policy. We first show that, whenever possible, it is optimal to route a call to the CSR class with the highest call resolution rate $p\mu$. We then derive conditions under which a generalization of this property, the $p\mu$ rule, is optimal. More precisely, we assume without loss of generality that the different classes of CSRs are indexed such that $p_1\mu_1 \geq \ldots \geq p_C\mu_C$. Then the $p\mu$ rule stipulates that, if the state is $\mathbf{n}$ when a call is routed, then the call should be routed to CSR class $m(\mathbf{n})$, where $m(\mathbf{n}) = \min\{k | k \in K(\mathbf{n})\}$.

We end this section with a full characterization of the optimal routing policy for the two-CSR case.

4.1 Partial Characterization of the Optimal Policy

For any $v \in \Omega$, define

$$\begin{aligned} \Delta_i v(\mathbf{n}) &= v(\mathbf{n}+\mathbf{e}_i) - v(\mathbf{n}) \qquad \forall\ i \in \{0,\ldots,C\}, \\ \Delta_{ij} v(\mathbf{n}) &= v(\mathbf{n}+\mathbf{e}_i) - v(\mathbf{n}+\mathbf{e}_j), \qquad \forall\ i,j \in \{0,\ldots,C\}. \end{aligned}$$

Moreover, define V to be the set of all $v \in \Omega$ that satisfy the following properties:

[P.1] $\Delta_i v(\mathbf{n}) \geq 0, \forall i \in K(\mathbf{n})$.

[P.2] $\Delta_0 v(\mathbf{n}) \geq 0$.

[P.3] $\Delta_{1i} v(\mathbf{n}) \leq 0$ if $1 \in K(\mathbf{n})$ and $i \in K(\mathbf{n})$.

[P.4] $\Delta_{10} v(\mathbf{n}) \leq 0$ if $1 \in K(\mathbf{n})$.

Properties P.1 and P.2 are fairly intuitive. They state that fewer calls in the system, either with Class-i CSRs or in queue, always result in smaller average total time in the system. Properties P.3 and P.4 together imply that, whenever possible, the policy corresponding to $v \in V$ always routes a call to a Class-1 CSR first.

The following lemma is used repeatedly in our analysis later. Its proof is straightforward and thus omitted:

Lemma 1 *Let* $\{x_1,\ldots,x_p\}$ *and* $\{y_1,\ldots,y_q\}$ *be two sets of real numbers. If for any* $i \in \{1,\ldots p\}$, *there exists a* $j(i) \in \{1,\ldots q\}$ *such that* $x_i \geq y_{j(i)}$, *then* $\min_{i\in\{1,\ldots p\}}\{x_i\} \geq \min_{j\in\{1,\ldots q\}}\{y_j\}$.

The following lemma states that Operator T preserves V.

Lemma 2 *If* $v \in V$ *then* $Tv \in V$.

Proof: In this proof, the terms "positive" and "negative" mean "non-negative" and "non-positive", respectively. Let $v \in V$. We first show that Tv satisfies P.1-P.3 by induction on n_0, the number of calls waiting in queue. We then deduce P.4.

Step 1 Consider states $\mathbf{n}$ where $n_0 = 0$. A direct computation leads to $\Delta_i Tv(\mathbf{n}) = \Delta_i v(\mathbf{n})$, $\Delta_0 Tv(\mathbf{n}) = \min\{\Delta_i v(\mathbf{n}), \Delta_0 v(\mathbf{n}) \mid i \in K(\mathbf{n})\}$, and $\Delta_{1i} Tv(\mathbf{n}) = \Delta_{1i} v(\mathbf{n})$. It follows from $v \in V$ that Tv satisfies P.1-P.3 for $\mathbf{n}$ such that $n_0 = 0$.

Step 2 Consider states $\mathbf{n}$ where $n_0 > 0$. Assume that Tv satisfies P.1-P.3 for all states where the number of calls waiting in queue is strictly less than n_0.

P.1 By definition, $Tv(\mathbf{n}+\mathbf{e}_i) = \min\{Tv(\mathbf{n}+\mathbf{e}_i+\mathbf{e}_j-\mathbf{e}_0), v(\mathbf{n}+\mathbf{e}_i) \mid j \in K(\mathbf{n}+\mathbf{e}_i)\}$. Note that if $j \in K(\mathbf{n}+\mathbf{e}_i)$ then $j \in K(\mathbf{n})$. Moreover, $Tv(\mathbf{n}+\mathbf{e}_i+\mathbf{e}_j-\mathbf{e}_0) \geq Tv(\mathbf{n}+\mathbf{e}_j-\mathbf{e}_0)$ since Tv is assumed to satisfy P.1 for states with $n_0 - 1$ waiting calls. Furthermore, $v(\mathbf{n}+\mathbf{e}_i) \geq v(\mathbf{n})$ since $v \in V$. Hence, $Tv(\mathbf{n})$ satisfies P.1 by Lemma 1.

P.2 Similarly, because $K(\mathbf{n}+\mathbf{e}_0) = K(\mathbf{n})$, v satisfies P.2, and Tv satisfies P.2 for $n_0 - 1$, we can use Lemma 1 to show that $Tv(\mathbf{n}+\mathbf{e}_0) = \min\{Tv(\mathbf{n}+\mathbf{e}_j), v(\mathbf{n}+\mathbf{e}_0) \mid j \in K(\mathbf{n}+\mathbf{e}_0)\} \geq \min\{Tv(\mathbf{n}+\mathbf{e}_j-\mathbf{e}_0), v(\mathbf{n}) \mid j \in K(\mathbf{n})\} = Tv(\mathbf{n})$.

P.3 For all $j \in K(\mathbf{n}+\mathbf{e}_i), j \neq 1$, we also have $j \in K(\mathbf{n}+\mathbf{e}_1)$. So we have $Tv(\mathbf{n}+\mathbf{e}_j-\mathbf{e}_0+\mathbf{e}_i) \geq Tv(\mathbf{n}+\mathbf{e}_j-\mathbf{e}_0+\mathbf{e}_1)$ since Tv is assumed to satisfy P.3 for n_0-1. For $j = 1 \in K(\mathbf{n}+\mathbf{e}_i)$, we can choose $i \in K(\mathbf{n}+\mathbf{e}_1)$, and we have $Tv(\mathbf{n}+\mathbf{e}_1-\mathbf{e}_0+\mathbf{e}_i) = Tv(\mathbf{n}+\mathbf{e}_i-\mathbf{e}_0+\mathbf{e}_1)$. Moreover, $v(\mathbf{n}+\mathbf{e}_i) \geq v(\mathbf{n}+\mathbf{e}_1)$ since v satisfies P.3. Therefore, by Lemma 1, $Tv(\mathbf{n})$ satisfies property P.3.

It follows that Tv satisfies P.1-P.3 for all n_0.

P.4 Finally, note that

$$\begin{aligned} Tv(\mathbf{n}+\mathbf{e}_0) &= \min\{Tv(\mathbf{n}+\mathbf{e}_j), v(\mathbf{n}+\mathbf{e}_0) \mid j \in K(\mathbf{n}+\mathbf{e}_0)\} \\ &= \min\{Tv(\mathbf{n}+\mathbf{e}_1), v(\mathbf{n}+\mathbf{e}_0)\} = Tv(\mathbf{n}+\mathbf{e}_1). \end{aligned} \tag{6}$$

So $\Delta_{10}Tv(\mathbf{n}) = 0$. The second equality in (6) holds because Tv satisfies Property P.3. The last one follows from the fact that $Tv(\mathbf{n}+\mathbf{e}_1)$ is less than or equal to $v(\mathbf{n}+\mathbf{e}_1)$ from the definition of T, which is in turn less than or equal to $v(\mathbf{n}+\mathbf{e}_0)$ from Property P.4. ■

If $v \in V$, then according to Lemma 2, Tv satisfies Properties P.3 and P.4, and (3) becomes: For any $\mathbf{n}$ where $1 \in K(\mathbf{n})$,

$$Tv(\mathbf{n}) = \begin{cases} v(\mathbf{n}) & \text{if} \quad n_0 = 0 \\ Tv(\mathbf{n}+\mathbf{e}_1-\mathbf{e}_0) & \text{if} \quad n_0 > 0 \end{cases} \tag{7}$$

In (6), we have actually shown that Tv satisfies a property stronger than P.4:

Corollary 1 *If $v \in V$ then $\Delta_{10}Tv(\mathbf{n}) = 0, \forall \mathbf{n}$ s.t. $1 \in K(\mathbf{n})$.*

The following theorem establishes P.1-P.4 for the optimal value function.

Theorem 1 *If $p_1\mu_1 \geq p_i\mu_i,\ \forall i \in \{2,\ldots,C\}$ and $v \in V$, then $\Gamma v \in V$.*

Proof: Consider $v \in V$. We first study the sign of $\Delta_i\Gamma$ for $i \geq 1$. From (5),

$$\begin{aligned} \Delta_i\Gamma v(\mathbf{n}) &= 1 + [(S_i - n_i - 1)\mu_1 + \sum_{j\neq i}(S_j - n_i)\mu_j]\Delta_i Tv(\mathbf{n}) \\ &\quad + \lambda\Delta_i Tv(\mathbf{n}+\mathbf{e}_0) + \mu_i(1-p_i)\Delta_0 Tv(\mathbf{n}) \\ &\quad + \sum_{j=1}^{C} n_j p_j \mu_j \Delta_i Tv(\mathbf{n}-\mathbf{e}_j) + \sum_{j=1}^{C} n_j(1-p_j)\mu_j \Delta_i Tv(\mathbf{n}-\mathbf{e}_j+\mathbf{e}_0), \end{aligned} \tag{8}$$

which is positive from P.1, P.2, and Lemma 2.

Similarly, based on P.3 and Lemma 2, we conclude

$$\Delta_0\Gamma v(\mathbf{n}) = 1 + \sum_{j=1}^{C}(S_j - n_j)\mu_j\Delta_0 Tv(\mathbf{n}) + \lambda\Delta_0 Tv(\mathbf{n}+\mathbf{e}_0)$$

$$+\sum_{j=1}^{C} n_j p_j \mu_j \Delta_0 Tv(\mathbf{n}-\mathbf{e}_j) + \sum_{j=1}^{C} n_j (1-p_j)\mu_j \Delta_0 Tv(\mathbf{n}-\mathbf{e}_j+\mathbf{e}_0) \geq 0. \quad (9)$$

We now turn our attention to $\Delta_{1i}\Gamma$.

$$\begin{aligned}
\Delta_{1i}\Gamma v(\mathbf{n}) &= \left[(S_i - n_i - 1)\mu_i + \sum_{j\neq i}(S_j - n_j)\mu_j\right]\Delta_{1i}Tv(\mathbf{n}) \\
&\quad +(\mu_1 - \mu_i)[Tv(\mathbf{n}+\mathbf{e}_0) - Tv(\mathbf{n}+\mathbf{e}_1)] - (p_1\mu_1 - p_i\mu_i)\Delta_0 Tv(\mathbf{n}) \\
&\quad +\lambda\Delta_{1i}Tv(\mathbf{n}+\mathbf{e}_0) + \sum_{j=1}^{C} n_j p_j \mu_j \Delta_{1i} Tv(\mathbf{n}-\mathbf{e}_j) \\
&\quad +\sum_{j=1}^{C} n_j (1-p_i)\mu_j \Delta_{1i} Tv(\mathbf{n}-\mathbf{e}_j+\mathbf{e}_0). \qquad (10)
\end{aligned}$$

We know that $Tv(\mathbf{n}+\mathbf{e}_0) = Tv(\mathbf{n}+\mathbf{e}_1)$ from Corollary 1. Moreover, since $p_1\mu_1 \geq p_i\mu_i$, $-(p_1\mu_1 - p_i\mu_i)\Delta_0 Tv(\mathbf{n}) \leq 0$ from P.2. Consequently, by P.3 and Lemma 2, $\Delta_{1i}\Gamma v(\mathbf{n}) \leq 0$.

Finally, we compute $\Delta_{10}\Gamma v$.

$$\begin{aligned}
\Delta_{10}\Gamma v(\mathbf{n}) &= \left[(S_1 - n_1 - 1)\mu_1 + \sum_{j>1}(S_j - n_j)\mu_j\right]\mu_j\Delta_{10}Tv(\mathbf{n}) + \lambda\Delta_{10}Tv(\mathbf{n}+\mathbf{e}_0) \\
&\quad -p_1\mu_1\Delta_0 Tv(\mathbf{n}) + \sum_{j=1}^{C} n_j p_j \mu_j \Delta_{10} Tv(\mathbf{n}-\mathbf{e}_j) \\
&\quad +\sum_{j=1}^{C} n_j (1-p_j)\mu_j \Delta_{10} Tv(\mathbf{n}-\mathbf{e}_j+\mathbf{e}_0), \qquad (11)
\end{aligned}$$

which is negative from P.2, P.4, and Lemma 2. ■

Theorem 1 allows us to partially characterize the optimal policy:

Corollary 2 *Assume that* $p_1\mu_1 \geq p_i\mu_i,\ \forall i \in \{2,\ldots,C\}$. *It is optimal to route a call to a Class-1 CSR whenever possible.*

Proof: From Theorem 1 and the application of MDP value iteration, the optimal value function can be shown to belong to V. Then, from P.3 and P.4, we conclude that at any

time, routing a call to a Class-1 CSR is better than either routing it to another available CSR or keeping it in the queue. ■

It is worth noting that, as long as $p_1\mu_1 \geq p_i\mu_i, \forall i$, calls will be routed to a Class-1 CSR whenever possible, even when μ_1 is smaller than some μ_i. Hence, $p\mu$ is a more useful index than μ in routing decisions. We believe that managers should focus on improving the CSRs' call resolution rates $p\mu$, instead of just their service rates μ. Moreover, CSRs should be given incentives that correspond to their call resolution rate. For instance, CSRs' compensation could be evaluated based on "calls resolved" rather than "calls handled". Shumsky and Pinker [26] have additional discussions on this topic.

4.2 The $p\mu$ Rule

Corollary 2 states that priority should be given to Class-1 CSRs, but it does not specify what to do when all Class-1 CSRs are busy and some other CSRs are available. A straightforward extension would be to give priority to the class with the highest $p\mu$ index among all those available. Recall that we name this the $p\mu$ rule.

In the case of two classes, the $p\mu$ rule is optimal, and can be viewed as an analog of the well known $c\mu$ result. However, for more than two classes of CSRs, the $p\mu$ rule may not be optimal. Consider the case where Class-2 CSRs have a higher call resolution rate, but they are much slower (i.e., $p_2\mu_2 > p_3\mu_3$ and $\mu_2 << \mu_3$). In this case, a call routed to a Class-2 CSR may still be in service when a better CSR (from Class 1) becomes available. However, if it had been routed to a Class-3 CSR instead, it might have either left the system earlier or returned and been re-routed to a Class-1 CSR earlier. So the optimal policy may prefer

Class 3 to Class 2 in some states. Specifically, for $C = 3$, $S_1 = 5$, $S_2 = S_3 = 2$, $\lambda = 7$, $\mu_1 = 4$, $p_1 = 0.6$, $\mu_2 = 3$, $p_2 = 0.4$, $\mu_3 = 9$, $p_3 = 0.1$, the optimal action in state $(1, 5, 0, 1)$ is 3. That is, when one call is in the queue, all Class-1 CSRs are busy, all Class-2 CSRs and 1 Class-3 CSR are available, it is optimal to route the call to a Class-3 CSR instead of a Class-2 CSR.

These are very rare and extreme cases, however. As our numerical tests will show, the $p\mu$ rule is optimal in most practical situations. Nevertheless, we need additional assumptions to analytically show the optimality of the $p\mu$ rule. Let the classes again be indexed such that $p_1\mu_1 \geq \ldots \geq p_C\mu_C$. We show below that the $p\mu$ rule is optimal when $\mu_2 \geq \ldots \geq \mu_C$. These conditions cover the cases in which the CSRs differ only in p (e.g. they follow the same scripts but have different problem-solving skills/training) or only in μ (e.g. the slow server problem). They also cover the cases in which p and μ are positively correlated (e.g., more experienced CSRs handle calls faster and give better answers).

Let W be the set of all real-valued functions defined on $\mathbf{N}^{C+1}$ that satisfy Properties P.1, P.2, P.4 and the following property:

[P.3'] $\Delta_{ki} w(\mathbf{n}) \leq 0$ if $i \in K(\mathbf{n})$ and $k = m(\mathbf{n})$.

Property P.3 is a special case of Property P.3' for $m(\mathbf{n}) = 1$, so W is a subset of V. In particular, under P.3' (7) remains true, and the policy corresponding to a value function belonging to W routes a call to a Class-1 CSR whenever possible.

The following lemma is analogous to Lemma 2.

Lemma 3 *If $\mu_2 \geq \ldots \geq \mu_C$ and $w \in W$ then $Tw \in W$.*

Proof: Since $W \subset V$, Tw satisfies P.1, P.2, and P.4 from Lemma 2. Now we use induction on n_0 to show that Tw satisfies P.3'.

Step 1 When $n_0 = 0$, $\Delta_{ki} Tw(\mathbf{n}) = \Delta_{ki} w(\mathbf{n})$ by definition and Tw satisfies P.3'.

Step 2 When $n_0 > 0$, we assume that Tv satisfies P.3' with $n_0 - 1$ calls waiting in the queue. This implies that $Tw(\mathbf{n} + \mathbf{e}_j - \mathbf{e}_0 + \mathbf{e}_i) \geq Tw(\mathbf{n} + \mathbf{e}_j - \mathbf{e}_0 + \mathbf{e}_k)$, for all $j \in K(\mathbf{n} + \mathbf{e}_i), j \neq k$. For $j = k \in K(\mathbf{n} + \mathbf{e}_i)$, we can choose $i \in K(\mathbf{n} + \mathbf{e}_k)$, and we have $Tv(\mathbf{n} + \mathbf{e}_k - \mathbf{e}_0 + \mathbf{e}_i) = Tv(\mathbf{n} + \mathbf{e}_i - \mathbf{e}_0 + \mathbf{e}_k)$. Furthermore, $w(\mathbf{n} + \mathbf{e}_i) \geq w(\mathbf{n} + \mathbf{e}_k)$ since $w \in V$ and the result follows from Lemma 1. ■

We are now ready to provide sufficient conditions under which the $p\mu$ rule is optimal.

Theorem 2 *If $p_1\mu_1 \geq \ldots \geq p_C\mu_C$, $\mu_2 \geq \ldots \geq \mu_C$ and $w \in W$, then $\Gamma w \in W$.*

Proof: Since w satisfies P.3', it also satisfies P.3. Following the same approach as in Theorem 1, we can show that Γw satisfies P.1, P.2, and P.4.

For Property P.3', a direct computation leads to, for $k \leq i$,

$$\begin{aligned}
\Delta_{ki}\Gamma w(\mathbf{n}) \;=\; & \left[(S_i - n_i - 1)\mu_i + \sum_{j \neq i}(S_j - n_j)\mu_j\right] \Delta_{ki} Tw(\mathbf{n}) \\
& + (\mu_k - \mu_i)[Tw(\mathbf{n} + \mathbf{e}_0) - Tw(\mathbf{n} + \mathbf{e}_k)] - (p_k\mu_k - p_i\mu_i)\Delta_0 Tw(\mathbf{n}) \\
& + \lambda \Delta_{ki} Tw(\mathbf{n} + \mathbf{e}_0) + \sum_{j=1}^{C} n_j p_j \mu_j \Delta_{ki} Tw(\mathbf{n} - \mathbf{e}_j) \\
& + \sum_{j=1}^{C} n_j (1 - p_j)\mu_j \Delta_{ki} Tw(\mathbf{n} - \mathbf{e}_j + \mathbf{e}_0). \qquad (12)
\end{aligned}$$

By the definition of T, $Tw(\mathbf{n} + \mathbf{e}_0) \leq Tw(\mathbf{n} + \mathbf{e}_k)$. Moreover, since μ_k is assumed to be larger than μ_i, $(\mu_k - \mu_i)[Tw(\mathbf{n} + \mathbf{e}_0) - w(\mathbf{n} + \mathbf{e}_k)] \leq 0$. The other terms of $\Delta_{ki}\Gamma$ are also negative from P.2 and P.3'. ■

The following straightforward corollary presents this result for the optimal control policy.

Corollary 3 *If $p_1\mu_1 \geq \ldots \geq p_C\mu_C$ and $\mu_2 \geq \ldots \geq \mu_C$ then*

- *The optimal policy routes a call to a Class-1 CSR whenever possible.*

- *If it is optimal to route a call in state* $\mathbf{n}$*, then this call is always routed to a Class-*$m(\mathbf{n})$ *CSR.*

That is, the $p\mu$ *rule is optimal.*

By including callback loops, Corollary 3 provides non-trivial generalizations of the slow server problem. Specifically, if we let $p_i = 1$, $i \in \{1, \ldots, C\}$, then Corollary 3 extends the optimality of the μ rule in Lin and Kumar [20], Walrand [33], and Koole [17] to more than two classes.

4.3 Threshold Policies

Results in Sections 4.1 and 4.2 partially characterize the optimal policy. In particular, Corollaries 2 and 3 specify *where* to route a call when it is optimal to do so. They do not specify *when* to route a call. In most cases, a threshold policy seems to provide an efficient and simple way to make this type of decision. Optimality of the threshold policy has been proved for the two-server slow server problem ($C = 2, S_1 = S_2 = 1, p_1 = p_2 = 1$) (see Lin and Kumar [20], Walrand [33], and Koole [17]). The theorem below extends their result to include the callback loops (p_1 and p_2 less than 1). Its proof can be found in Appendix A.

Theorem 3 *Suppose* $C = 2$*,* $S_1 = S_2 = 1$*, and* $p_1\mu_1 \geq p_2\mu_2$*. The optimal routing policy is characterized by a threshold* t^* *such that:*

- *If the Class-1 CSR is available, the policy routes a waiting call to the Class-1 CSR;*

- *If the Class-1 CSR is busy, the policy routes a waiting call to the available Class-2 CSR if and only if the queue length is larger than t^*.*

Theorem 3 is the first optimality result for threshold policy in a queueing system with callback loops. It suggests that threshold-based policies are indeed suitable heuristics for systems with the callback loops.

When there are more than one CSR per class, and/or more than two classes, the situation is much more complex. Our extensive numerical tests (heavy, medium, and light traffic and different combinations of arrival and service rates, and call-resolution probabilities) suggest that the optimal policy is always a state-dependent threshold policy: in each state, it is optimal to route a call to a certain idle CSR (not necessarily following the $p\mu$ rule) if and only if queue length exceeds a threshold. These thresholds depend on the number of busy CSRs in *each* class, so potentially there could be as many as $\left[\prod_{i=2}^{C}(1+S_i)\right]-1$ thresholds.

5 Numerical Analysis and Extensions

Since optimal state-dependent threshold policies are hard to compute and incorporate, in Section 5.1 we propose heuristics that perform well and are simple to apply in practice. More fundamentally, we evaluate the importance of incorporating p into the routing decisions in Section 5.2. We also investigate various extensions of our modeling assumptions: In Section 5.3 we explore situations where the service rate depends on the number of times a CSR has talked to the same customer before. In Section 5.4 we consider cases in which customers call back not immediately but after an exponentially distributed time. To conclude our numerical analysis, we propose a lower bound for the optimal average time of call resolution

in Section 5.5.

5.1 $p\mu$-based policies

Lemmas 2 and 3 show that $p\mu$ is a very important routing index. In this section we study two policies based on the $p\mu$ rule:

- Theorem 3 shows the optimality of threshold policies in simple settings that include callbacks. This inspires us to use a threshold-based policy for more complex settings. Consider the *$p\mu$ policy with a fixed threshold t*, or simply the *$p\mu$-t policy*. With two CSR classes, this policy uses the $p\mu$ rule and routes a call to a Class-2 CSR if the queue length exceeds t, regardless of how many (as long as not all) Class-2 CSRs are busy. The threshold t will be optimally selected among all possible fixed thresholds. This policy simplifies the state-dependent threshold policy by using a single fixed (i.e., state-independent) threshold, and is optimal for the case of two heterogeneous CSRs ($C = 2, S_1 = S_2 = 1$).

- A *$p\mu$ policy* further simplifies the *$p\mu$-t policy* by routing a call to an available Class-2 CSR as soon as possible. That is, it sets $t = 0$.

For comparison purposes, we also study the following policy, which does not use $p\mu$ as a factor in the routing decisions:

- A *random assignment policy* routes a call randomly to *any* available CSR. This is the policy often used by call centers that do not incorporate any $p\mu$ information into routing decisions.

Our numerical analysis includes 54 cases, which cover light (Cases 1-18), medium (Cases 19-36), and heavy (Cases 37-54) traffic situations. Of the 18 cases for each situation, we analyze when $p_1\mu_1$ and $p_2\mu_2$ are close (the first 9 cases) and far apart (the next 9 cases). Then for each fixed $p_i\mu_i, i = 1, 2$, we let p_i and μ_i take on three sets of values so that there are 9 combinations. The purpose is to test "normal" cases as well as "extreme" cases, which will give us a sense of the "bound" on the differences. Detailed parameter values are given in Table 4 in Appendix B. For each case, we compare the random assignment policy, the $p\mu$ policy, and the $p\mu$-t policy with the optimal state-dependent threshold policy determined numerically by a value iteration algorithm.

Results in Table 4 show that the benefit of allowing the threshold to vary state-by-state (i.e., optimal vs $p\mu$-t) is minimal. This is intuitive: Although the thresholds used by the (optimal) state-dependent threshold policy vary significantly between $n_2 = 0$ and $n_2 = S_2-1$, only a few of these thresholds really matter since most of the (S_1, n_2) states are visited very infrequently (if at all) in the steady state. Therefore the $p\mu$-t policy, which uses the best t for all states, performs well. This also simplifies the search for optimal control parameters.

Furthermore, we observe that the benefit of withholding some calls (i.e. $p\mu$-t vs $p\mu$), similar to the benefit gained in the slow server problem, is far less than the benefit of recognizing and utilizing the $p\mu$ rule in call routing (i.e. $p\mu$ vs random). Since the $p\mu$ policy does not require any computation except for the ranking of $p\mu$ index, this means that in most cases the $p\mu$ policy is a better policy for implementation. Actually, the performance of the $p\mu$ policy is dramatically worse than that of the $p\mu$-t policy only for Cases 28, 31, and 34. These cases correspond to 1) medium traffic situations (the utilization rate is 50%), 2) a wide difference between $p_1\mu_1$ and $p_2\mu_2$, and 3) $p_2 = 1$. To understand 1), we note that when

traffic is high, Class-2 CSRs are heavily used and the optimal threshold is low. When traffic is low, Class-2 CSRs are hardly necessary. Both of these situations lead to small difference between $p\mu$ and $p\mu$-t policies. For 2), when the CSR heterogeneity is higher, the optimal threshold should be higher, leading to a greater difference between $p\mu$ and $p\mu$-t policies. To see 3) we note that when $p_2 < 1$, an unresolved call by a Class-2 CSR can be re-routed to a Class-1 CSR. When $p_2 = 1$, however, once a call is routed to a Class-2 CSR, it remains there. So the use of a threshold to withhold calls becomes more important when $p_2 = 1$, leading to a greater difference between $p\mu$ and $p\mu$-t policies. In practical situations, the traffic is usually high, and $p_2 < 1$. Therefore, the difference between the $p\mu$ and $p\mu$-t policies diminishes.

We conclude this section by analyzing the performances of the $p\mu$-t and $p\mu$ policies as the size of the call center increases. Tested cases and results are presented in Table 1. For all cases we let $\mu_1 = \mu_2 = 2$, $p_1 = 1$, $p_2 = 0.5$, and increase λ and $S_1 = S_2$ by a scale factor varying from 1 to 20 such that $\rho = 2/3$.

As shown by Table 1, the $p\mu$-t policy always performs very well with an error less than or equal to 0.3%. Maybe more interesting is the efficiency of the $p\mu$ policy, which has an error less than 1%. These results suggest that our findings for small systems remain true for larger ones. We also observe that as the system size grows (with traffic intensity at a fixed value), the threshold also increases but at a lower rate and remains small relative to the total number of CSRs.

We note that there could be another way of testing the size effect. Instead of fixing the system utilization as we increase the size of the call center, we could also fix a certain service level (e.g. 5% delay probability). As the arrival rate increases, the size of the call center would increas in a way that follows the square-root staffing rule (e.g., see Borst,

Scale Factor	λ	Optimal threshold for $p\mu$-t policy	Cost Increase Over Optimal Policy	
			$p\mu$-t policy	$p\mu$ policy
1	16	2	0.00%	0.62%
2	32	2	0.00%	0.89%
4	64	3	0.01%	0.88%
6	96	3	0.04%	0.83%
8	128	4	0.02%	0.76%
10	160	4	0.02%	0.71%
12	192	5	0.30%	0.93%
14	224	5	0.00%	0.01%
16	256	6	0.04%	0.33%
18	288	6	0.02%	0.60%
20	320	6	0.03%	0.58%

Table 1: Impact of the Size of the Call Center

Mandelbaum, and Reiman [6]). One difficulty is that these rules are not derived for the heterogeneous servers, callback loops, and priority rules that are essential in our model. It would be an interesting area for future research to see how the square-root staffing rule can be adapted to our model.

5.2 Importance of Call Resolution Probability p

We want to stress in this paper the importance of incorporating the call resolution probability p into the call routing priority index. In this section, we set $\mu_1 = \mu_2$ and fix p_1. Then we systematically decreased p_2, starting from $p_2 = p_1$. If the manager of a call center only measures the speed of its CSRs, then it will assume that all CSRs are the same. Therefore a random assignment policy will be used. On the other hand, if the call center measures the call resolution probability p of each CSR, then it should route calls according to p (i.e., use the $p\mu$ and $p\mu$-t policies).

The parameters used in the tests are as follows: $\lambda = 1, S_1 = S_2 = 8, \mu_1 = \mu_2 = 0.18, p_1 = 0.7$; p_2 varies. Results are summarized in Figure 2.

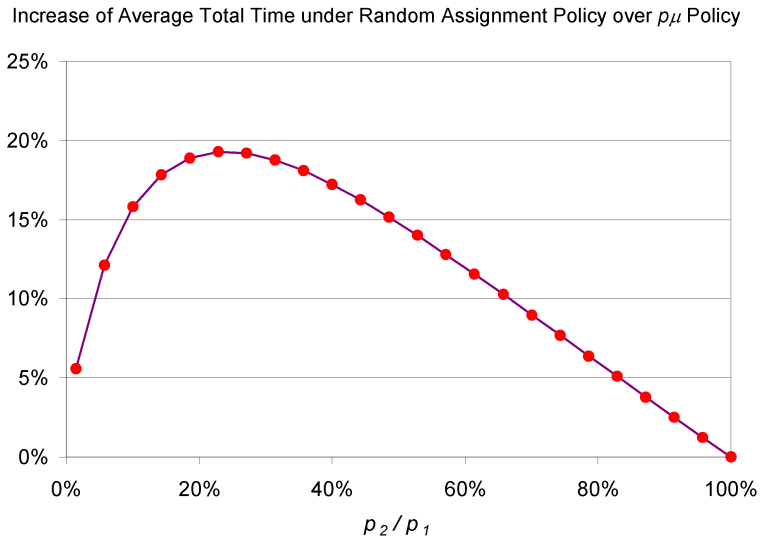

Figure 2: Comparing the $p\mu$ and random assignment policies: $\mu_1 = \mu_2$.

We observe that the random policy performs very poorly against the $p\mu$ policy in most cases. In general, the smaller the ratio p_2/p_1, the bigger the difference. This is intuitive because the benefit of recognizing the difference between p_1 and p_2 and utilizing it in routing is greater when the difference is bigger. However, in the extreme, as p_2 approaches 0, the system traffic intensity approaches 0.99. This is very heavy traffic and all the policies tend to use the Class-2 CSRs whenever possible. That explains why the difference narrows as $p_2 \to 0$.

Note that when $\mu_1 = \mu_2$, the $p\mu$ policy simply gives calls to the CSR class with the higher p. When the ranking of the CSRs according to their call resolution probabilities is common knowledge (or can be measured), such a policy is very easy to implement, and also gives significant benefits.

5.3 Dedicated Policy

So far, we have assumed that the service rates do not depend on the number of previous attempts to solve the customer's problem. In practical situations, the service time may decrease if the customer talks to the same CSR (e.g. call centers dealing with complex issues such as medical and legal help). In such cases, it may indeed be better to route a callback to the CSR who answered this call the first time (the *original CSR*).

In this section we numerically evaluate the performance of a *dedicated policy*. This policy allocates new calls according to the $p\mu$-t routing policy, but always routes callbacks immediately to the original CSR. The dedicated policy applies to situations where it is the CSR who reaches the conclusion that the problem has not been properly addressed. Instead of handing the call off to another CSR, which would result in another setup time, the CSR may want to keep the call and give it another try. The dedicated policy also applies to call centers requiring callbacks to enter a case number at the phone prompt that corresponds to the particular customer issue. For call centers that cannot identify the reason of a call as it enters the system, the implementation of the dedicated policy is difficult.

Let us assume that each time a callback is routed to the original CSR, the average service rate increases by a given percentage δ. In other words, the average service rate for the kth attempt is equal to $(1+\delta)^{k-1}\mu_i$ for a Class-i CSR. Therefore, the total average service time (taking the callbacks into account), $\tilde{\mu}_i$, is equal to $\mu_i(p_i+\delta)/(1+\delta)$. To simplify the analysis we assume that the total service time is exponentially distributed, and the system becomes a slow server problem with rates $\tilde{\mu}_i$.

We compare the dedicated policy with the $p\mu$-t policy. We assume that the $p\mu$-t policy

does not utilize customer callback information, so that it is unlikely for a callback to be reassigned to the original CSR. Therefore we assume that under the $p\mu$-t policy the service rates do not depend on the number of previous attempts. At the end of this section, we discuss how to use callback information in the $p\mu$-t policy.

When $\delta = 0$, the total service time by a Class-i CSR is a geometric sum of exponential random times with the same rate μ_i, and the system is equivalent to a slow server system with service rates of $\{p_i\mu_i\}$, and no callbacks. As δ increases (i.e., the time saving becomes larger), the gap between the dedicated policy and the $p\mu$-t should narrow. Eventually there should exist a δ^* such that the dedicated policy outperforms the $p\mu$-t policy if $\delta \geq \delta^*$.

We let $S_1 = S_2 = 8$, $p_2\mu_2 = 1$, and let $p_1\mu_1 = \mu_2$ vary from 1.1 to 2. Figure 3 depicts δ^* as $\Delta p\mu := (p_1\mu_1 - p_2\mu_2)/p_2\mu_2$ increases. Although δ^* is increasing in $\Delta p\mu$, for δ^* to be significant $\Delta p\mu$ needs to be large. For instance, when $p_1\mu_1$ is 50% larger than $p_2\mu_2$, the dedicated routing policy should be used as soon as CSRs can improve the service rates at each attempt by 4%.

This suggests that the dedicated routing policy should work well when $\Delta p\mu$ is not particularly large. It also suggests that when the original CSR of a callback can be identified, the call routing policy should use this information. Here we propose a modified $p\mu$-t policy that takes advantage of the benefits of the dedicated policy: The $p\mu$-t policy still determines when (i.e. when the queue length exceeds the threshold) and where (i.e. $p\mu$ rule) to route a call as before. In addition, if the call is a callback, the original CSR is idle and is in the class identified by the $p\mu$ rule, then the modified $p\mu$-t policy should route the call to the original CSR. Note that if the queue discpline is FCFS, the original CSR may not always be available. Even when callbacks are given priority in routing, they may still wait in the

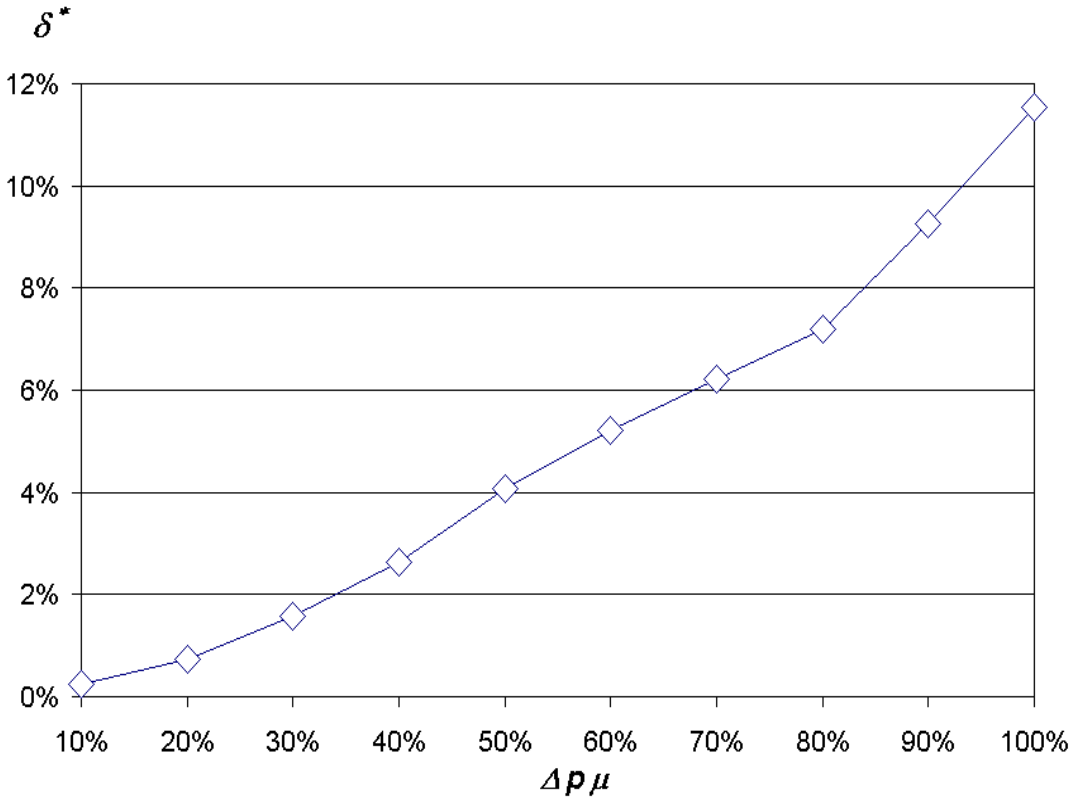

Figure 3: δ^* as a Function of $\Delta p\mu$

queue if, upon their return, all Class-1 CSRs are busy and the queue length is below threshold. By the time the callbacks are routed, their original CSRs may not be available either. In summary, the modified policy is based on the $p\mu$-t policy but it routes the callbacks to their original CSR whenever possible. Other modifications of the $p\mu$-t are also possible. We believe that when δ is significant, these modified $p\mu$-t policies constitute good alternatives to the dedicated routing policy.

5.4 Delay in call back

So far, we have assumed that when a call is not resolved successfully, the call returns immediately to the system. In many instances, however, the resolution of a call may not be

immediate. Therefore, the customer leaves the system after being served, and calls back (if needed) only after a certain amount of time. From a modeling perspective, this system can be viewed as having a callback "orbit". Unresolved calls stay in the orbit for an exponentially distributed time with rate ν before coming back to the system. Since the number of calls in the orbit is usually unknown to the call center, this model is a Partially Observed Markov Decision Process for which general results and algorithms are limited (e.g., see Puterman [27]).

Let us call the immediate-callback model in Section 3 the IC model, and the delayed-callback model the DC model. Note that the IC model corresponds to the DC model in which $\nu = \infty$. In this section, we test how well the $p\mu$ heuristics (developed for the IC model) perform in the DC model. More precisely, we identify the best threshold of the $p\mu$-t policy for the IC model and apply the same policy to the corresponding DC model. We also test the $p\mu$ policy.

Since the DC model cannot be evaluated, we actually look at another system in which the orbit size is limited and the full state information, including the size of the callback orbit, is known to the decision maker. Both assumptions reduce the system cost, resulting in a lower bound of the system. It is against this lower bound that we numerically test the performance of the $p\mu$-t and $p\mu$ policies (see Appendix C for more details).

Tests for 24 cases are summarized in Table 2. In cases 1-8, 9-16, and 17-24, we have $\mu_1 = \mu_2$, $\mu_1 > \mu_2$, and $\mu_1 < \mu_2$ respectively. We also start with the IC model ($\nu = \infty$). Then we gradually decrease ν to 1/64. If an average call lasts 5 minutes, then $\nu = 1/64$ corresponds to an average delay of more than 5 hours before callback. Values for the other parameter values and the results are given in Table 2.

			Cost Increase Over Lower Bound	
Case	μ_2	ν	$p\mu$-t policy	$p\mu$ policy
1	1	∞	0.0%	0.0%
2	1	1	0.0%	0.0%
3	1	1/2	0.0%	0.0%
4	1	1/4	0.0%	0.0%
5	1	1/8	0.0%	0.0%
6	1	1/16	0.0%	0.0%
7	1	1/32	0.0%	0.0%
8	1	1/64	0.0%	0.0%
9	0.8	∞	0.1%	0.1%
10	0.8	1	0.2%	0.3%
11	0.8	1/2	0.1%	0.4%
12	0.8	1/4	0.1%	0.4%
13	0.8	1/8	0.0%	0.5%
14	0.8	1/16	0.0%	0.5%
15	0.8	1/32	0.0%	0.5%
16	0.8	1/64	0.0%	0.5%
17	1.2	∞	0.0%	0.0%
18	1.2	1	0.0%	0.0%
19	1.2	1/2	0.0%	0.0%
20	1.2	1/4	0.0%	0.0%
21	1.2	1/8	0.0%	0.0%
22	1.2	1/16	0.0%	0.0%
23	1.2	1/32	0.0%	0.0%
24	1.2	1/64	0.0%	0.0%

Table 2: Effect of ν for different μ_2, with $\lambda = 4$, $p_1 = 1$, $\mu_1 = 1$, $p_2 = 0.6$ and $S_1 = S_2 = 4$.

Results in Table 2 suggest that further decreasing ν will not have a significant impact. Moreover,

- Both the IC $p\mu$-t and $p\mu$ policies, when applied to the DC model, have costs that are extremely close to the DC lower bound (all errors are less than 1%). This suggests that in practice, the information about the orbit size (i.e., the number of unresolved calls that will eventually come back) is not necessary.

- As the average delay of callback varies from "immediate" to "more than 5 hours", no significant changes are noted, providing another justification for the immediate callback assumption: The $p\mu$-t and $p\mu$ policies generated by the IC model work very well in the DC model, where there is a significant delay in the callback.

An explanation for the insensitivity of the results to the delay is that, in the MDP formulation we study the steady-state behavior of a stationary queueing system. With or without delay, unresolved calls will eventually come back. So the delay orbit simply changes the timing of the callbacks, but not the rate of the callbacks. Consequently, the total rate of callbacks to the system is similar for both the IC and the DC models. In the IC model, there is a strong correlation between service completion and callback arrivals. In the DC model, due to the (esp. exponential) delay between the two events, the correlation becomes weaker. Numerical results in Table 2 suggests that the similarity of overall callback rate between the IC and DC models is much stronger than the difference between the two models caused by the completion-callback correlation.

5.5 Lower bound policy

The $p\mu$, $p\mu$-t, and dedicated policies (see Table 4, Column 11 for the performance of the dedicated policy when $\delta = 0$) perform well in general, but they all provide an upper bound on the performance of the optimal policy studied in Section 3. To complete the analysis we provide in this section the closed-form solution to a policy that gives a lower bound on the optimal system performance.

The lower bound policy we study is the *preemptive policy*: at any time, even if a call is already being served by a CSR, we allow it to be handed over to another CSR during the service. Since the service times are exponentially distributed, we can assume that the call starts over after the hand-off. The preemptive assumption is very restrictive, making the policy applicable only to call centers where customers tolerate such hand-offs. Nevertheless, it provides a good lower bound which is also easy to evaluate.

Because preemption is allowed at any time, it makes sense to not hold calls in the queue when there are idle CSRs (one can always re-route the calls later). Moreover, a call should always be routed or re-routed to the highest-$p\mu$ available CSR. These intuitions are formalized by the following theorem.

Theorem 4 *The optimal preemptive policy always routes (or re-routes, if the call is already in service) a call to the available CSR with the highest $p\mu$ index.*

Proof: To simplify the proof, we introduce a new class of calls, the *dummy calls*, which are not generated by real customers but rather by the system controller. At any time instant the controller can add a dummy call to the queue or to an available server. Once in the system, dummy calls are routed, re-routed, and served just like the real calls but they do not count

towards the system cost. As a result, the whole system evolves exactly as if the dummy calls were real, but the system cost is lower than with the real calls. The role of a dummy call is essentially that of a "placeholder": The use of a dummy call is equivalent to idling the exact same sequence of CSRs that would have been used by a real call, if the real call were introduced into the system precisely when the dummy call is added. This technique was also used in Walrand [33] for the 2-server slow server problem.

The proof follows two steps. First, we show that whenever possible, it is better to route a call to an available CSR than to hold it in the queue. Next we show that when routing, it is best to route to the highest-$p\mu$ available CSR. We use coupling arguments to show contradictions if these are not true.

Step 1 Assume that π is an optimal preemptive policy and, without loss of generality, assume routing under π is FCFS. Let there be a recurrent state s (transient states are of no consequence), in which π holds a call in the queue instead of routing it to an available CSR, i. Tag this call and name it a. Now let π' be a policy that duplicates all of policy π's actions, until the system gets into state s, when π' routes a to an available CSR i. For the purpose of this proof, we will call the system under π System 1 and the system under π' System 2.

Because π is FCFS, it will route a next. Assume that π routes a after T time units to CSR j (with possibly $j = i$). Between 0 and T, there may be some job completions and arrivals, but under π, there will be no routing actions. Under π', we let the same job completions and arrivals to occur, and take no routing action either. The only difference between Systems 1 and 2 is that under π' there may be an additional event, namely CSR i completing call a. There are three possibilities:

1. If under π', CSR i completes a before T and the customer is satisfied, then we let a leave System 2, and at the same time, we insert a dummy call into the queue. At this moment, Systems 1 and 2 are completely coupled in terms of state dynamics, but the cost is lower for System 2 because it has a dummy call instead of a real one.

2. If under π', CSR i completes a before T and the customer is not satisfied. Then the customer calls back right away, and we keep her in the queue. At this moment, the two systems are completely coupled.

3. If under π', CSR i does not complete a before T, then at time T, we let π' preempt a from CSR i and route it to CSR j . At this moment, the two systems are completely coupled.

Following these steps, System 1 and System 2 are coupled. Moreover, System 2 does no worse than System 1 in all three cases, and it is strictly better (has one fewer real call) for a strictly positive time period in the first case. Since s is recurrent, π' is strictly better than π, which leads to a contradiction.

Step 2 We now prove the optimality of the $p\mu$ rule also by contradiction. Without loss of generality, we uniformize the event rates to $\lambda + \sum_{i=1}^{C} S_i\mu_i = 1$, and study the uniformized Markov chain.

Assume that in a recurrent state s, instead of routing a call (again, name it a) to CSR i, an optimal policy π routes a to CSR j, where $p_j\mu_j < p_i\mu_i$. In System 1, after one uniformize transition, two type of events are possible:

(i) a is satisfied and leaves the system, with probability $p_j\mu_j$; or

(ii) any other event occurs, with probability $1 - p_j\mu_j$, and a is re-routed to CSR k (with possibly $k = j$).

The events in (ii) include a service completion of a that is not satisfactory, a service completion of another job, an arrival, or a fictitious event due to uniformization.

Let π' be a policy that duplicates π's actions until the system gets into state s, when it routes a to CSR i instead of j. After one uniformized transition,

1. if (i) occurs in System 1, we let a be satisfied and leave System 2;

2. if (ii) occurs in System 1,

   - with probability $\frac{p_i\mu_i - p_j\mu_j}{1-p_j\mu_j}$, we let a be satisfied and leave System 2; at the same time, we assign a dummy to CSR k;
   - with probability $\frac{1-p_i\mu_i}{1-p_j\mu_j}$, a remains in the system and is re-routed to CSR k.

It is straightforward to verify that in System 2, a is satisfied and leaves the system with probability $p_i\mu_i$ (as it should be). Moreover, at the end of one uniformized transition, System 2 is completely coupled with System 1 with the help of the dummy, and, thanks to the dummy, it has lower system cost for a positive period of time with positive probability. Therefore, Policy π' is strictly better than π, which leads to a contradiction. ■

As a result of Theorem 4, when a service is completed and the customer is dissatisfied, the call would be routed to the same CSR (otherwise, this call would have been re-routed earlier). Therefore, without loss of generality, we can set $p_1 = p_2 = \ldots = p_C = 1$, and μ_i to be the original $p_i\mu_i$ in the following analysis.

Due to preemption, calls in the system will always be handled by the fastest CSRs. For example, if there are i calls in the system, where $S_1 < i \leq S_1 + S_2$, then S_1 of the calls will be handled by Class-1 CSRs and $i - S_1$ of them will be handled by Class-2 CSRs. As a result, the only variable we need to keep track of is the total number of calls in the system, i. So if μ_i denotes the total rate of service completion in state i, then

$$\mu_i = \begin{cases} i\mu_1 & \text{for } 0 \leq i \leq S_1 \\ S_1\mu_1 + (i - S_1)\mu_2 & \text{for } S_1 < i \leq S_1 + S_2 \\ \cdots & \\ \sum_{k=1}^{C-1} S_k\mu_k + (i - \sum_{k=1}^{C-1} S_k)\mu_C & \text{for } \sum_{k=1}^{C-1} S_k < i \leq \sum_{k=1}^{C} S_k \\ \sum_{k=1}^{C} S_k\mu_k & \text{for } \sum_{k=1}^{C} S_k < i. \end{cases} \tag{13}$$

If we let q_i denote the steady-state probability of the system being in state i, then the state-transition balance equations are: $\lambda q_i = \mu_{i+1} q_{i+1}, \forall\ i$.

Therefore, if we let $S = \sum_{k=1}^{C} S_k$ and $\mu_S = \sum_{k=1}^{C} S_k\mu_k$, we must have

$$q_i = \begin{cases} q_S \cdot \prod_{j=i+1}^{S} \left(\frac{\mu_j}{\lambda}\right) & \forall 0 \leq i \leq S-1 \\ q_S \cdot \left(\frac{\lambda}{\mu_S}\right)^{i-S} & \forall i \geq S \end{cases}. \tag{14}$$

Solving the probability uniformization equation $\sum_{i=0}^{\infty} p_i = 1$, we obtain

$$q_S = \frac{1}{\sum_{i=0}^{S-1} \left[\prod_{j=i+1}^{S} \left(\frac{\mu_j}{\lambda}\right)\right] + \frac{1}{1-(\lambda/\mu_S)}}. \tag{15}$$

This, along with (14), uniquely determine all of the steady-state probabilities. We can use these steady-state probabilities to calculate the average number in the system, which will give us a lower bound on the performance of our system:

$$L_s = q_S \left[\sum_{i=1}^{S-1} \left(i \prod_{j=i+1}^{S} \frac{\mu_j}{\lambda}\right) + \frac{\lambda\mu_S}{(\mu_S - \lambda)^2} + \frac{S\mu_S}{\mu_S - \lambda}\right]. \tag{16}$$

Using the closed-form expressions given in (14)-(16), we can quickly compute the performance of this preemptive system. To see how tight the lower bound is, we test it using the 54 cases in §5.1. Results are included in Table 4. For most cases the lower bound is within

2.5% of the optimal policy. The cases with bad performance (46, 47, 49, 50, 52, and 53) are extreme cases in which $\mu_1 >> \mu_2$. They are very unlikely to occur in practice.

5.6 Workforce Scheduling

When callback probability $1 - p$ is high, there are two ways in which the $p\mu$ based policies can benefit a call center:

1. With the same number of CSRs, a better service level can be achieved (less wait for the customers); or

2. Fewer CSRs are needed to achieve the same performance.

In this section, we will study the latter, headcount reduction, in the context of a simple 2-class workforce scheduling problem.

We assume that currently the call center uses random assignment policy to achieve the following service level for every 30-minute time interval: "Average wait in the system should be less than 3.25 minutes". We use a scaled-down version of call volumes at a financial services company call center as the arrival rates in our example. For simplicity, we will assume that the mix of two classes in a call center is about 50%-50%. Class-1 and Class-2 CSRs can both serve 20 customers in an hour, but they differ in the service quality: Class-1 CSRs have 90% call resolution probability while Class-2 CSRs have only 50%.

We assume workshifts of 8 consecutive hours. Without loss of much generality, we ignore the coffee breaks and lunch breaks to simplify the analysis (in a more thorough analysis, these can be incorporated into the Linear Program we use here). Each shift can start every half an hour. The tool we use to make daily workforce scheduling is a standard scheduling

Linear Program (LP). The objective function in LP is headcount minimization, and the constraints are the standard one that the service level be achieved for all time intervals.

Data and results of the numerical analysis are summarized in the Table 3. Call volumes by every half-hour are listed in the second column. Because of the random assignment routing policy and the 50%-50% CSR mix assumption, it is equivalent to view all the CSRs as the same, with a 70% call resolution probability. In the third column, we then use the standard Erlang-C formula to calculate the number of CSRs needed under the current random assignment policy to satisfy the service level.

Next, given the necessary numbers in the third column, we run the scheduling LP to figure out the minimum total number of CSRs necessary to meet those needs. The result is that 19 is the minimum. So the call center following the random assignment policy will schedule 19 people to work, and the work schedule is listed in the fourth column. Clearly, due to the 8-consecutive-hour workshift restriction, in many time intervals we will have more CSRs working than necessary. And the service level achieved will be better than the stated one.

Now we will see how the call center can benefit by using the optimal routing policy. First, of the 19 people currently employed, we assume that 10 of them are Class-1 and 9 of them are Class-2. We run a scheduling LP for the 10 Class-1 CSRs first. The result is listed in the fifth column. Next, for each half-hour, given the number of Class-1 CSRs, we use our model to numerically calculate the minimum number of Class-2 CSRs needed for that time interval, so as to achieve the same (or better) service level as achieved by the random assignment policy (which may be better than the stated one). This is done for each half-hour, and the

Time Interval	Arrival	# Needed	# Scheduled		Class-1 Schedule	Class-2 Needed	Class-2 Scheduled
12:00-12:30	4.15	2	2		1	0	0
12:30-1:00	2.9	2	4		1	0	0
1:00-1:30	2.6	2	3		1	0	0
1:30-2:00	1.8	2	3		1	0	0
2:00-2:30	1.05	1	2		1	0	0
2:30-3:00	1.1	2	2		1	0	0
3:00-3:30	1.1	2	2		1	0	0
3:30-4:00	2.4	2	2		1	0	0
4:00-4:30	0.5	1	4		1	0	0
4:30-5:00	0.95	1	4		1	0	0
5:00-5:30	1.65	2	4		1	0	0
5:30-6:00	3.25	2	4		1	0	0
6:00-6:30	3.2	2	5		1	0	0
6:30-7:00	6.05	2	5		2	0	0
7:00-7:30	10.55	3	5		3	0	0
7:30-8:00	15.8	3	5		3	0	0
8:00-8:30	30.05	5	5		3	0	3
8:30-9:00	46.4	6	6		3	2	3
9:00-9:30	86.25	9	9		5	3	3
9:30-10:00	99	10	10		6	2	3
10:00-10:30	101.85	11	11		6	3	3
10:30-11:00	104	11	11		6	3	3
11:00-11:30	96.45	10	12		6	2	4
11:30-12:00	105.55	11	12		6	4	4
12:00-12:30	99.3	10	10		5	4	4
12:30-1:00	92	10	11		7	1	4
1:00-1:30	93.5	10	11		7	0	4
1:30-2:00	100.15	11	11		7	0	5
2:00-2:30	97.05	10	10		7	0	5
2:30-3:00	93.1	10	10		6	2	6
3:00-3:30	92.8	10	10		5	4	6
3:30-4:00	90.9	10	10		5	4	6
4:00-4:30	85.45	9	12		6	1	3
4:30-5:00	71.25	8	9		6	0	3
5:00-5:30	58.35	7	7		4	2	3
5:30-6:00	49.1	6	6		3	3	3
6:00-6:30	44.4	6	6		3	2	3
6:30-7:00	40.7	6	6		3	2	3
7:00-7:30	35.05	5	5		3	1	2
7:30-8:00	32.4	5	5		3	0	2
8:00-8:30	27.95	5	5		4	0	2
8:30-9:00	26.4	4	4		2	2	2
9:00-9:30	23.2	4	4		2	1	2
9:30-10:00	20.05	4	4		2	1	1
10:00-10:30	18.8	4	4		2	1	1
10:30-11:00	17.15	4	4		2	0	0
11:00-11:30	10.65	3	4		2	0	0
11:30-12:00	9.9	3	4		2	0	0

Table 3: Workforce scheduling example

results are listed in the sixth column. Finally, we run the scheduling LP again to find the minimum Class-1 CSR needed. The result is 6, and the schedule is listed in the last column.

This means, using the current random assignment policy, 19 CSRs are needed to satisfy the service level; while using our optimal $p\mu$ policy, we need only 10 Class-1 and 6 Class-2 CSRs to achieve the same (or better) service levels as those under current policy. The implication is that by incorporating the $p\mu$ index in routing decisions and scheduling accordingly, we can achieve a 15.8% headcount saving.

It is well known, however, that the call volumes vary significantly from day to day and from week to week. The headcount reduction would be hard to achieve consistently if on some days the possible reduction is small while on other days it is big. A better way to interpret this numerical result is that by incorporating the $p\mu$ optimal policy in routing, the call center could have 3 Class-2 CSRs in training for that entire day, without sacrificing the service level. The 3 CSRs do not have to be the same ones for the whole day: a rotating training scheduling is possible. That way, the call center can hope to achieve better service in the long run (by training Class-2 CSRs) without having to sacrificing the short-run system performance. For companies in the process of migrating from the traditional cost-based metrics (e.g. average wait time) to the profit-based metrics (e.g. first-call resolution) (see [28] for details), our procedure helps them to maintain the service measured by current metrics in the short run, while increasing their service as measured by new metrics in the long run.

5.7 More Than Two Classes of CSR

So far we have focused on $C = 2$, but we would like to extend the results to $C \geq 3$. There are two immediate problems when we extend the $p\mu$-t policy to $C \geq 3$: 1) the $p\mu$ rule may not be optimal; 2) even when the $p\mu$ rule is optimal, we still have to find $C - 1$ fixed thresholds - one for each Class-i, $2 \leq i \leq C$. It would be helpful if instead, we could find a simpler and intuitive heuristic that performs well.

In reality, each CSR has a different p and μ value. The categorization of CSRs into classes is an approximation done for convenience and tractability. We argue that in most applications, there is a decreasing incremental benefit for having more classes: the benefit of recognizing the difference in CSRs and having two CSR classes is great; but the benefit of going from two classes to three is less; and so on. This is just our intuition, which is hard to show analytically. In this section we carry out numerical experiments to test this idea.

To do that, we assumed that the CSRs could be grouped into three classes, with parameters S_1, p_1, μ_1, S_2, p_2, μ_2, and S_3, p_3, μ_3. On the one hand, we could use these parameters to numerically calculate the optimal routing policy. On the other hand, we could deliberately choose to have only two classes by merging two of the three classes and using some aggregate average parameters for the merged class. Then we would use the heuristics we developed in the previous section to find a routing policy based on these two classes. To see how well this approach works, we then evaluated it in the three-class situation.

If the three classes are numbered in descending order of their call resolution rate $p\mu$ as usual, then there are three ways to merge the groups: merge Classes 1 and 2; merge Classes 2 and 3; and merge Classes 1 and 3. In this section we will test to see which merge works

best, and when.

Without loss of generality, let's assume for now that Classes 2 and 3 are merged. Call this Class 2'. Class 2' will have $S_{2'} = S_2 + S_3$ CSRs. Moreover, we let (as a heuristic)

$$S_2\mu_2 + S_3\mu_3 = S_{2'}\mu_{2'} \qquad \text{and} \qquad S_2\mu_2 p_2 + S_3\mu_3 p_3 = S_{2'}\mu_{2'}p_{2'},$$

then

$$\mu_{2'} = \frac{S_2\mu_2 + S_3\mu_3}{S_2 + S_3} \qquad \text{and} \qquad p_{2'} = \frac{S_2\mu_2 p_2 + S_3\mu_3 p_3}{S_2\mu_2 + S_3\mu_3}.$$

The idea is that if the call center manager collects data only on the processing rate, then s/he should observe approximately the same average speed from all these CSRs, whether they are considered to be two classes or one class. Similarly, we want the merged class to have approximately the same effective service rate as the separate two classes.

Clearly these aggregated parameters are approximations because they should be different if different routing policies are used.

Once we had $S_{2'}$, $\mu_{2'}$ and $p_{2'}$, we then used them, along with S_1, μ_1 and p_1, to calculate the best $p\mu$-t policy. To evaluate the resulting policy in a three-CSR-class situation, we let any calls routed to Class 2' be randomly distributed among the CSRs who originally were Class 2 or Class 3.

The above procedure was repeated for each merging possibility (1&2, 2&3, 1&3), and we chose the one with the lowest system cost. This numerical procedure was carried out for 162 tests, which include the following scenarios:

- We let $S_1 = S_2 = S_3 = 4$.
- We examined heavy and medium traffic intensity ($\lambda = 4.5$ and $\lambda = 3$). These are the regions in which call centers are most likely to operate.

- For each traffic intensity, we fixed $p_1\mu_1 = 0.8, p_3\mu_3 = 0.2$ and studied $p_2\mu_2 = 0.7$, $p_2\mu_2 = 0.5$, or $p_2\mu_2 = 0.2$ (representing the cases in which Classes 1 and 2 are close, all classes are evenly spaced, and Classes 2 and 3 are close).

- For each fixed $p\mu$ value, we studied three cases: low p, medium p, and high p.

  - For $p_1\mu_1 = 0.8$, we could have: $(p_1, \mu_1) = (0.1, 8), (0.8, 1)$, or $(1, 0.8)$;

  - For $p_2\mu_2 = 0.7$, we could have: $(p_2, \mu_2) = (0.1, 7), (0.7, 1)$, or $(1, 0.7)$;

  - For $p_2\mu_2 = 0.5$, we could have: $(p_2, \mu_2) = (0.1, 5), (0.5, 1)$, or $(1, 0.5)$;

  - For $p_2\mu_2 = 0.3$, we could have: $(p_2, \mu_2) = (0.05, 5), (0.3, 1)$, or $(1, 0.3)$;

  - For $p_3\mu_3 = 0.2$, we could have: $(p_3, \mu_3) = (0.05, 4), (0.2, 1)$, or $(1, 0.2)$.

  Because there were three classes, there were 27 combinations for each fixed $p_1\mu_1$, $p_2\mu_2$, and $p_3\mu_3$.

We observe from the numerical results that the following "$p\mu$ first, μ second" rule-of-thumb seems to work well:

- $p\mu$ is the most important index to use when considering which two classes to merge. In most of the tests, when Classes 1 and 2 are close, merging them is best; when Classes 2 and 3 are close, merging them is best. It should be intuitive that one should never merge Classes 1 and 3 because that means treating *very* different CSRs as if they were the same. Numerical results confirm this.

- There are some exceptions to the above rule. For example, when $p_1\mu_1$ and $p_2\mu_2$ are close but μ_1 and μ_2 are very different, merging Classes 2 and 3 could be better. Even

then, merging Classes 1 and 2 is only slightly worse, so using the above rule-of-thumb still works well. In reality, these extreme cases rarely occur.

- When the $p\mu$ values are evenly spaced (0.8, 0.5, 0.2), we find that μ serves as a very good secondary index: it's better to merge the two classes whose μ values are closer. There are also a few exceptions to the rule, but merging according to this rule is once again only slightly worse.

As to the efficiency of the proposed "merging + $p\mu$-t" heuristic procedure, we have the following observations:

- Overall, the heuristic performs quite well. The heuristic performs worst in the cases where either the $p\mu$'s or the μ's are widely different. Again, such extreme cases rarely occur in practice.

- We also tested the random assignment policy for all these cases. In general, the random assignment policy performs much worse than our heuristic.

Of course, there may be other, better heuristics. But it should be noted that this proposed heuristic is (relatively) easy to use. The straightforward ($p\mu$ then μ) rule-of-thumb helps in deciding how to merge classes. Compared with what exists in the literature and in practice, this is a practical way of dealing with a complex situation that still yields good results.

It should also be noted that it is hardly desirable to have many classes in practice. While having many classes may pave the way for differentiated pay and a clear career path for the CSRs, in most cases it creates complexity both for routing/scheduling and for management. Whenever it is desirable to have only 2 classes of CSR, no merging is necessary, and the $p\mu$-t

policy performs very well.

6 Conclusion

Traditional research on routing decisions focuses on speed and waiting cost. Service quality related metrics are rarely taken into account for such operational decisions, though they play a crucial role in the short-term traffic reduction and long-term customer loyalty of a firm. We see our research as a promising step in showing that service quality can be - and should be - incorporated into operational decisions.

In this paper, we consider both service speed and quality in routing decisions for a telephone call center. We argue that call resolution probability p is a good measure of call quality. An MDP model is used to characterize the optimal routing policy. Our main contribution is to identify call resolution rate $p\mu$ as a simple priority index in routing calls: First we show that the use of the $p\mu$ rule is optimal in a broad set of cases. Then we show that the $p\mu$-threshold policies are optimal in certain cases. Finally we show numerically that simple $p\mu$-based policies work well as heuristics. These numerical tests highlight the benefits that can be achieved by considering p, in addition to the traditional measure of μ, when making routing decisions.

Even though results in this paper focus primarily on the short-term benefit of traffic reduction, incorporating quality related metrics into routing decisions could also provide significant long-term benefits. For instance, the workforce scheduling example shows that, to achieve the same service level on customer waiting time, fewer low-$p\mu$ CSRs are needed under a $p\mu$-based policy than under a μ-based policy. The freed-up low-$p\mu$ CSRs can then be

scheduled to receive training. Over the long run, the call center could improve its CSRs' service speed and/or quality, all without adding extra personnel or sacrificing service level. For companies in the process of migrating from the traditional cost-based metrics (e.g. average wait time) to the profit-based metrics (e.g. call resolution probability), our procedure helps them to maintain the service measured by current metrics in the short run, while increasing their service as measured by new metrics in the long run.

In our future research, we will examine other long-term benefits such as customer loyalty. For example, when a customer is dissatisfied with a service, s/he may simply defect and never call back. So if call quality is not carefully considered in routing decisions, a company could lose many customers in the long run due to poor service quality. For call centers that outsource, this also has an impact on how both speed and resolution probability should be specified in contracts.

Acknowledgments

The authors thank Paul Zipkin for his insightful comments on an earlier version of this paper. We also thank the seminar participants at Duke University, the University of North Carolina, and the University of Alberta for helpful suggestions. The many helpful comments and suggestions by two anonymous reviewers and the Associate Editor have significantly improved the paper, and they are much appreciated.

References

[1] V. De Angelis. Planning Home Assistance for AIDS Patients in the City of Rome, Italy. Interfaces, 28(3), May-June 1998

[2] R. Atar, A. Mandelbaum and M. Reiman. Scheduling a Multi-Class Queue with Many Exponential Servers: Asymptotic Optimality in Heavy-Traffic. The Annals of Applied Probability, 14(3):1084–1134, 2004.

[3] 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

[4] E. Berk, and K. Moinzadeh. The Impact of Discharge Decisions on Health Care Quality. Management Science, 44(3), March 1998

[5] D. Bertsekas. Dynamic Programming: Deterministic and Stochastic Models. Prentice-Hall, NJ, 1987.

[6] S. Borst, A. Mandelbaum and M. Reiman. Dimensioning Large Call Centers. Operations Research, 52(1):17–34, 2004.

[7] F. de Véricourt and Y-P. Zhou A Note On the Incomplete Results for the Multiple-Server Slow-Server Problem. Technical Report, Duke University, The Fuqua School of Business, October 2004.

[8] G.I. Falin and J.G.C. Templeton. *Retrial Queues.* Chapman & Hall, 1997.

[9] N. Gans. Customer Loyalty and Supplier Quality Competition. Management Science, 48(2), Feb 2002

[10] N. Gans, G. Koole and A. Mandelbaum. Telephone Call Centers: Tutorial, Review, and Research Prospects. Manufacturing & Service Operations Management, 5:79-141, 2003.

[11] N. Gans and Y-P. Zhou. Managing Learning and Turnover in Employee Staffing. Operations Research, 50(6), Nov-Dec 2002

[12] N. Gans and Y-P. Zhou. A Call-Routing Problem with Service-Level Constraints. Operations Research, 51(2), Mar-Apr, 2003.

[13] A. Ha. Stock-Rationing Policy for a Make-to-Stock Production System with Two Priority Classes and Backordering. Naval Research Logistics, 44:457-472, 1997

[14] J. Hall and E. Porteus. Customer Service Competition in Capacitated Systems. Manufacturing & Service Operations Mgmt, 2(2), Spring 2000

[15] J. M. Harrison and M. J. Lopéz. Heavy Traffic Resource Pooling in Parallel-Server Systems. *Queueing Systems*, 33, 1999.

[16] J. Heskett and E. Sasser. Putting the service-profit chain to work. Harvard Business Review, March-April 1994

[17] G. Koole. A Simple Proof of the Optimality of a Threshold Policy in a Two-Server Queueing System. *Systems & Control Letters*, 26, 1995

[18] R.L. Larsen. Control of Multiple Exponential Servers with Application to Computer Systems. Ph.D. Dissertation, Department of Computer Science, University of Maryland, College Park, 1981.

[19] R. Larsen and A. K. Agrawala. Control of a Heterogeneous Two-Server Exponential Queueing System. *IEEE Transactions on Software Engineering*, 9(4), 1983

[20] W. Lin and P.R. Kumar. Optimal Control of a Queueing System with Two Heterogeneous Servers. *IEEE Transactions on Automatic Control*, AC-29(8), 1984

[21] H. Luh and I. Viniotis. Threshold Control Policies for Heterogeneous Server Systems. *Mathematical Methods of Operations Research*, 55, 2002

[22] A. Mandelbaum, W.A. Massey, M. Reiman, B. Rider, and A. Stolyar. Queue Lengths and Waiting Times for Multiserver Queues with Abandonment and Retrials. *Selected Proceedings of the Fifth INFORMS Telecommunications Conference*, 2000.

[23] A. Mandelbaum, W.A. Massey, M. Reiman, and B. Rider. Time Varying Multiserver Queues with Abandonment and Retrials. In P.Key and D.Smith, *Teletraffic Engineering in a Competitive World*, ITC-16, 355–364, Elsevier, 1999.

[24] A. Mandelbaum, W.A. Massey, M. Reiman, and A. Stolyar. Waiting Time Asymptotics for Time Varying Multiserver Queues with Abandonment and Retrials. *Allerton Conference Proceedings*, 1999.

[25] A. Mandelbaum and A. Stolyar. $Gc\mu$ Scheduling of Flexible Servers: Asymptotic Optimality in Heavy Traffic. Technical Report, September 2002

[26] R. Shumsky and E. Pinker. Gatekeepers and Referrals in Services. *Management Science*, 49(7), July 2003.

[27] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: John Wiley & Sons, 1994.

[28] B. Read. The Struggling Revolution. *Call Center Magazine*, Dec 4, 2002 Downloadable at http://www.callcentermagazine.com/article/CCM20021202S0005.

[29] V.V. Rykov. Monotone Control of Queueing Systems with Heterogeneous Servers. Queueing Systems, 37(4):391-403, 2001

[30] Y.-C. Teh and A.R. Ward. Critical Thresholds for Dynamic Routing in Queueing Networks. Queueing Systems,42:297–316, 2002.

[31] J.A. van Mieghem. Dynamic scheduling with convex delay costs: the generalized $c\mu$ rule. Annals of Applied Probability, 5:809-833, 1995

[32] M. Veatch and L. Wein Scheduling a Make-To-Stock Queue: Index Policies and Hedging Points. *Operations Research*, 44(4):634-647, 1996

[33] J. Walrand. A Note on "Optimal Control of a Queueing System with Two Heterogeneous Servers". *Systems & Control Letters*, 4, 1984

[34] V. Zeithaml, A. Parasuraman and L. Berry. The nature and determinants of customer expectations of service. Acad. Marketing Sci., 21(1), 1993

A Proof of Theorem 3

Corollary 2 allows us to simplify the state space to $\mathbf{x} = (x_1, x_2)$ where $x_1 = n_1 + n_0$ and $x_2 = n_2$.

If we allow actions to be taken at fictitious transitions caused by uniformization, then g^* and $v^*(\mathbf{x})$ will satisfy the following optimality equation:

$$\tilde{\Gamma} v^*(\mathbf{x}) + g^* \quad = \quad v^*(\mathbf{x}), \tag{17}$$

where

$$\begin{aligned} \tilde{\Gamma} v(\mathbf{x}) \quad = \quad & (x_1 + x_2) + \lambda T v(x_1 + 1, x_2) + (1 - p_1)\mu_1 T v(x_1, x_2) + p_1 \mu_1 T v((x_1 - 1)^+, x_2) \\ & + \left[(1 - p_2)\mu_2 T v(x_1 + 1, 0) + p_2 \mu_2 T v(x_1, 0)\right] I_{\{x_2 = 1\}} + \mu_2 T v(x_1, 0) I_{\{x_2 = 0\}}, \end{aligned} \tag{18}$$

where I is the indicator function, and

$$T v(\mathbf{x}) = \begin{cases} \min[v(x_1, 0), v(x_1 - 1, 1)] & \text{if } x_1 \geq 2 \text{ and } x_2 = 0, \\ v(\mathbf{x}) & \text{otherwise.} \end{cases} \tag{19}$$

We now introduce the set U of all real-valued functions defined on $\mathbf{N}^2$ that satisfy the following properties:

[P.5] $\Delta_{12} u(\mathbf{x} + \mathbf{e}_1) \geq \Delta_{12} u(\mathbf{x})$ if $x_2 = 0$.

[P.6] $\Delta_2 u(\mathbf{x} + \mathbf{e}_1) \geq \Delta_2 u(\mathbf{x})$ if $x_2 = 0$.

[P.7] $\Delta_i u(\mathbf{x} + \mathbf{e}_i) \geq \Delta_i u(\mathbf{x})$ if $x_i = 0$.

[P.8] $\Delta_i u(\mathbf{x}) \geq 0$ if $x_i = 0$.

Property P.5 implies that if $u \in U$, the corresponding routing policy is of the threshold type. Property P.6 is better known as supermodularity. Together, Properties P.5 and P.6

imply that u is convex with respect to x_i, which is exactly Property P.7. Property P.8 is equivalent to Properties P.1 and P.2.

The following Lemma plays a crucial role in the proof:

Lemma 4 *If $u \in U$, then $\tilde{\Gamma}u \in U$.*

Proof: Consider $u \in U$. As stated previously, Property P.7 is implied by P.5 and P.6. Furthermore Koole [17] shows that $Tu(\mathbf{x})$ and $Tu((x_1-1)^+, x_2)$ belong to U. Let us show that $\tilde{\Gamma}u$ verifies P.5. A direct computation yields:

$$\begin{aligned}\Delta_{12}\tilde{\Gamma}v(\mathbf{x}) &= \lambda\Delta_{12}Tv(\mathbf{x}+\mathbf{e}_1) + (1-p_1)\mu_1\Delta_{12}Tv(\mathbf{x}) + p_1\mu_1\Delta_{12}Tv\left((x_1-1)^+, x_2\right) \\ &\quad +p_2\mu_2\Delta_1 Tv(x_1, 0), \end{aligned} \tag{20}$$

which is increasing in x_1 from Properties P.5 and P.7.

Similarly for P.6,

$$\begin{aligned}\Delta_2\tilde{\Gamma}v(\mathbf{x}) &= 1 + \lambda\Delta_2Tv(\mathbf{x}+\mathbf{e}_1) + (1-p_1)\mu_1\Delta_2Tv(\mathbf{x}) + p_1\mu_1\Delta_2Tv\left((x_1-1)^+, x_2\right) \\ &\quad +(1-p_2)\mu_2\Delta_1 Tv(x_1, 0), \end{aligned} \tag{21}$$

which is also increasing in x_1 for the same reasons.

Note also that $\Delta_2\tilde{\Gamma}$ is positive since T verifies Property P.8. $\Delta_1\tilde{\Gamma}$ can also be easily checked to be positive which shows P.8. ■

From Lemma 4 and the application of the value iteration, the optimal value function v^* belongs to U. The result directly follows if we define t^* as the smallest x_1 such that $\Delta_{12}v^*(x_1, 0)$ is positive or null. The value of t^* is well defined from Property P.5.

B Test cases in Sections 5.1

The parameters and results are detailed in Table 4.

C Formulation of the delayed-callback model

In the delayed-callback model, dissatisfied customers go into an orbit, and stay there for an exponentially distributed time before calling back. The average time in orbit is $1/\nu$. We represent the states by $(\mathbf{n}(t), u(t))$, where $u(t)$ is the number of dissatisfied customers in the callback orbit at time t. Then the total callback rate at time t is equal to $\nu u(t)$. Because $u(t)$ is unbounded, to apply the uniformization procedure, we need to impose an upper bound M on the number of calls in orbit and assume that $\lambda + \sum_{i=1}^{C} S_i\mu_i + M\nu = 1$.

The state of the orbit is not directly observable, but we will use this information to solve the problem. Clearly, the solution will be a lower bound on the optimal solution of the POMDP. Since M limits the callback, a finite M imposes a further lower bound. It is this lower bound that we compare with the IC model $p\mu$-t and $p\mu$ policies. Note that these two policies are based only on the observable part (number of calls in queue, number of busy CSRs, etc.) of the system.

We can find the lower bound by numerically solving the following optimality equation using value iteration:

$$\begin{aligned} v^*(\mathbf{n}) + g^* \;=\; & \sum_{i=0}^{C} n_i + \lambda Tv(\mathbf{n}+\mathbf{e}_0, u) + \nu u Tv(\mathbf{n}+\mathbf{e}_0, u-1) + \sum_{i=1}^{C} n_i(1-p_i)\mu_i Tv(\mathbf{n}-\mathbf{e}_i, u+1) \\ & + \sum_{i=1}^{C} n_i p_i \mu_i Tv(\mathbf{n}-\mathbf{e}_i) + \left[\sum_{i=1}^{C}(S_i - n_i)\mu_i + (M-u)\nu\right] v(\mathbf{n}). \end{aligned} \tag{22}$$

where g^* is the optimal cost and

$$Tv(\mathbf{n}) = \begin{cases} \min\{v(\mathbf{n}+\mathbf{e}_i-\mathbf{e}_0, \min(u,M)), v(\mathbf{n}, \min(u,M)) \mid i \in K(\mathbf{n})\} & \text{if} \quad n_0 > 0 \\ v(\mathbf{n}, \min(u,M)) & \text{otherwise} \end{cases}. \tag{23}$$

The numerical tests in Table 2 are restricted to medium utilization rate, a few CSRs, small ν, and large ρ. This is due to the curse of dimensionality (common with most MDP approaches) and the fact that we need to choose M and the truncated queue sufficiently large to obtain accurate results. We believe that the results hold for general cases as well.

						Cost Increase Over Optimal Policy				
Case	p_1	μ_1	p_2	μ_2	ρ	rand. assign.	$p\mu$	$p\mu$-t	preemptive	dedicated
1	0.65	1	1	0.6	0.2	4.74 %	0 %	0 %	-0.06 %	0.07 %
2	0.65	1	0.6	1	0.2	3.95 %	0 %	0 %	-0.04 %	0.09 %
3	0.65	1	0.06	10	0.2	1.02 %	0 %	0 %	0 %	0.12 %
4	1	0.65	1	0.6	0.2	4.05 %	0 %	0 %	-0.06 %	0.07 %
5	1	0.65	0.6	1	0.2	3.26 %	0 %	0 %	-0.04 %	0.09 %
6	1	0.65	0.06	10	0.2	0.73 %	0 %	0 %	0 %	0.12 %
7	0.1	6.5	1	0.6	0.2	7.09 %	0 %	0 %	-0.06 %	0.07 %
8	0.1	6.5	0.6	1	0.2	6.63 %	0 %	0 %	-0.04 %	0.09 %
9	0.1	6.5	0.06	10	0.2	3.29 %	0 %	0 %	0 %	0.12 %
10	0.95	1	1	0.3	0.2	94.07 %	0.23 %	0 %	-0.01 %	0 %
11	0.95	1	0.3	1	0.2	52.45 %	0.06 %	0 %	-0.01 %	0 %
12	0.95	1	0.03	10	0.2	8.95 %	0 %	0 %	-0.01 %	0 %
13	0.5	1.9	1	0.3	0.2	116.68 %	0.23 %	0 %	-0.01 %	0 %
14	0.5	1.9	0.3	1	0.2	73.8 %	0.06 %	0 %	-0.01 %	0 %
15	0.5	1.9	0.03	10	0.2	15.59 %	0 %	0 %	-0.01 %	0 %
16	0.1	9.5	1	0.3	0.2	161.45 %	0.23 %	0 %	-0.01 %	0 %
17	0.1	9.5	0.3	1	0.2	130.51 %	0.06 %	0 %	-0.01 %	0 %
18	0.1	9.5	0.03	10	0.2	50.45 %	0 %	0 %	-0.01 %	0 %
19	0.54	0.5	1	0.23	0.5	5.76 %	0 %	0 %	-1.62 %	1.31 %
20	0.54	0.5	0.46	0.5	0.5	5.07 %	0 %	0 %	-0.97 %	1.97 %
21	0.54	0.5	0.1	2.3	0.5	3.02 %	0 %	0 %	-0.3 %	2.67 %
22	1	0.27	1	0.23	0.5	4.68 %	0 %	0 %	-1.62 %	1.31 %
23	1	0.27	0.46	0.5	0.5	3.96 %	0 %	0 %	-0.97 %	1.97 %
24	1	0.27	0.1	2.3	0.5	2.18 %	0 %	0 %	-0.3 %	2.67 %
25	0.1	2.7	1	0.23	0.5	8.31 %	0 %	0 %	-1.62 %	1.31 %
26	0.1	2.7	0.46	0.5	0.5	8.05 %	0 %	0 %	-0.97 %	1.97 %
27	0.1	2.7	0.1	2.3	0.5	6.09 %	0 %	0 %	-0.3 %	2.67 %
28	0.8	0.5	1	0.1	0.5	82.41 %	14.49 %	0 %	-2.36 %	0.01 %