STAFFING AN INBOUND CALL CENTER

by

O. Z. AKSIN*
and
P. T. HARKER**

97/15/TM

* Assistant Professor of Operations Management at INSEAD, Boulevard de Constance, Fontainebleau 77305 Cedex, France.

** Department of Operations and Information Management, University of Pennsylvania, The Wharton School, Philadelphia, PA 19104-6366, USA.

Staffing an Inbound Call Center

O. Zeynep Akşin * Patrick T. Harker †

December, 1996

Abstract

This paper studies a staffing problem for inbound call centers with multiple call types and service agents that specialize in these different call types. The staffing problem seeks to allocate servers in a call center where performance is determined by its server allocation as well as its telecommunication and information technology resources; profit maximization is the goal of this problem. For the pure loss case, which does not allow for reneged calls, it is shown that a greedy allocation procedure yields the optimal server allocation. Similarly, for the more general case with only one type of call class, a simple search procedure is shown to result in the optimal number of servers. Heuristics are proposed for the general multi-class form of the staffing problem, whose performances are evaluated through numerical examples at the end of the paper.

1 Introduction

Consolidation is sweeping across the financial services industry. As new mergers are being announced, the industry's managers are struggling to cope with the need to realign their processes, technology and people needs. The effects of consolidation are also being felt in one of the major service delivery channels of financial service firms, namely their call centers. Given the changing needs of a bigger organization, some banks are establishing new call centers, while others are consolidating or resizing existing centers.

*Technology Management Area, INSEAD, Boulevard de Constance, 77305 Fontainebleau Cedex, France

†University of Pennsylvania, Department of Operations and Information Management, The Wharton School, Philadelphia,PA 19104-6366

Change has also been taking place within call centers of the financial services industry, in particular, in the consumer/retail segment of the industry. As the call center has grown and matured as a service delivery channel, the industry has seen a shift in its focus from being solely a service provider to becoming a revenue generator for the firm. This shift brings with it a need to manage resources in a way that balances the traditional efficiency and quality needs with the new revenue aspirations. In an earlier paper Akşin and Harker (1996), the authors have demonstrated that increased sales activity in a call center can dramatically change resource utilization within the center, creating the need to restructure, restaff, or resize.

The current study has in part been motivated by these trends in retail banking and is an effort to develop a methodology that will help banks restructure their call centers. The results in this paper are clearly a first step in this direction. In particular, this study will look at a static design problem. Given the infrastructure at a call center, defined as the telecommunication and information technology in place, and demand faced by this center, the economically optimal, or profit maximizing staffing levels will be determined. The work in this paper looks at staffing at a tactical level, rather than daily staffing and scheduling decisions. In this regard, it only addresses one dimension of the restructuring needs of call centers in financial services. It should be clear, however, that methodologically, this paper establishes a core building block for phone center design and management, consisting of managing interacting technology and human resources.

In order to establish the relationship between resource allocation and profits at a call center, the study will make use of results in an earlier paper by the authors Akşin and Harker (1996a). These results will be summarized in the third section, following a brief literature review in Section 2. The staffing problem is formulated in Section 3, subsequent to the review of the performance model. The paper addresses two different forms of the model, stemming from different assumptions regarding call center configuration and customer behaviour. The more basic one of these, the loss system, is analyzed in the fourth section. It is shown that a greedy allocation procedure results in an optimal solution to the staffing problem for this form of the model. The analysis next proceeds to the system with reneges, which constitutes the form that is more relevant in the retail banking call center context. Section 5 first shows that for a center with single call types, the optimal staffing level can be determined via a simple search procedure. This result is then used to suggest a decomposition scheme for centers with multiple call types, resulting in a staffing heuristic for the system with reneges. Section 6.3 evaluates the performance of this heuristic

numerically. The paper concludes with a discussion of future research directions.

2 Literature Review

A large number of studies in the operations literature have focused on design issues in stochastic systems. These strive to determine the value of various controllable variables in order to optimize performance measures of interest. Resource allocation problems constitute a significant proportion of this line of study. Allocation of transport vehicles and emergency units in urban emergency services Vitt (1979), Larson (1975), Chaiken and Larson(1972), Green and Kolesar (1984), allocation of servers, machines or workloads in multiple center manufacturing systems Shantikumar and Yao (1987), Shantikumar and Yao (1988), Dallery and Stecke (1990), Lee et al. (1991), Boxma et al. (1990), Frenk et al. (1994), Frostig (1993), and allocation of memory or processor capacity in computer networks Foshini and Gopinath (1983), Yamazaki and Sakasegawa (1987), De Waal and Van Dijk (1991), De Waal (1993) provide some examples of resource allocation applications. Performance measures that are typically optimized in these kinds of applications are throughput, customer waiting times, or resource utilization.

Staffing problems constitute an important kind of resource allocation problem. Staffing decisions pertaining to the size of the work force and their scheduling have attracted much attention in the context of post office staff scheduling Krajewski and Henderson (1979), nurse scheduling Maier-Rothe and Wolfe (1973), Rising and Kaminsky (1971), and bank teller staffing McMahon and Long (1991), Van der Velde (1988), Mabert (1977). Mabert (1986) argues that much of this research fails to incorporate the impact of equipment decisions on staffing decisions. His simulation study, based on operating data from two banks, demonstrates that increased staffing flexibility reduces required equipment investments, thereby lowering costs. A different strand of research looks at staffing issues that arise in dual resource constrained job shops Treleven (1989), where again the interaction between equipment and labor is considered. By construction, the performance model in this study enables a similar analysis, characterizing the impact of telecommunication and information processing resources on staffing decisions in inbound call centers. Issues related to detailed shift scheduling, which are dealt with in some of the earlier cited references, will not be a concern of the current study.

As phone centers have grown in number and in size, more firms have tried to improve

their management by focusing on resource utilization and service levels. This has led to a series of studies dealing with the problem of staffing phone centers, many of which have made use of queueing models to capture the impact of congestion. These models have gained acceptance through successful applications in a variety of settings like airlines Gaballa and Pearce (1979), telephone operators Sze (1984), IRS's Telephone Taxpayer Information System Harris et al. (1987), laboratories Callahan and Khan (1993), and hospitals Agnihothri and Taylor (1991). While Callahan and Khan (1983) attempt to determine revenue maximizing capacity, for all the other examples, the objective is to determine staffing levels that minimize costs, subject to a service level constraint imposed by management. Service levels for different allocations of staffing are obtained through a queueing model.

Recently, in a study of operations at L.L.Bean's phone center, Quinn et al. (1991) and Andrews and Parsons (1993) take a different approach to telecommunications resource allocation and telephone-agent staffing. Instead of allocating resources subject to a fixed service level requirement, they determine the allocation that maximizes profits. Their analyses leads to economically optimal staffing, with service levels fluctuating as a function of changing demand. It is shown that this approach leads to higher profits than the earlier ones in which service levels were pre-determined. In the ensuing study, a similar approach is taken to determine staffing levels for an inbound call center. The difference of this analysis comes from the underlying stochastic model that is used to characterize phone center operations. In particular, the study will consider staffing for multiple access channels that interact through a shared resource.

3 Model Overview

This section starts out with a review of a performance model for the operations at a phone center, first proposed in Akşin and Harker (1996a). The model is used to evaluate performance at a center, given call statistics, staffing, and installed technology base. This performance model is then embedded in an optimization model that seeks to determine optimal staffing levels at a center. This latter model is formulated subsequently, at the end of the section.

3.1 The Performance Model

The operations of a phone center are modeled earlier in Akşin and Harker (1996a), providing a relationship between capacity choice and system performance measures, which can then be used to determine the relationship with system revenues. This model takes into account the uncertainty in demand, hence establishes a measure of capacity which explicitly deals with congestion. Capacity is a stochastic entity, which is a function of demand and resource allocation within a center. Resources that jointly determine capacity are human resources in the form of service agents, telecommunication resources as phone lines and VRUs (voice response units), and information technology resources. A customer call will require the availability of a phone line, through which the call can gain access to a service representative or a VRU. At the same time, the representative will need access to certain applications or databases in order to provide the requested services.

In the sequel, this multi-channel queueing system with processor sharing is used as the performance model for a phone center. The specific assumptions underlying the proposed performance model are summarized below. The reader is referred to Akşin and Harker (1996a) for a detailed exposition and analysis of this model. Two different ways of measuring performance are dealt with herein, which are based on different assumptions regarding customer behavior and system configuration. In the basic case, which is called the *loss system*, it is assumed that customers are extremely impatient. Hence, any customer who cannot initiate service immediately will leave. It is assumed that all customers who leave are lost demand and will not retry until their next transaction. In this configuration of the system, the number of trunks or phone lines are equal to the number of service representatives. Next, consider a system which may have phone lines in excess of the number of service representatives. Upon arrival of a call, if all trunks are taken, the customer receives a busy signal and leaves the system. On the other hand, if a trunk is available but all agents are busy, the customer is put on hold. While some customers wait until an agent becomes available, some customers may exhibit impatience and leave the system while on hold before service initiation. This loss of customers is labeled as reneges and the system is called the *system with reneges*. For many inbound call centers, this will constitute the most realistic model.

Consider a phone center with K access channels. Each access channel consists of T_k, $k = 1, \ldots, K$ phone trunks and S_k, $k = 1, \ldots, K$ service agents specializing in product line k with $T_k \geq S_k$. Customers arrive at the various access channels with an arrival

rate of λ_k, where arrivals in each channel are independent of each other and the arrival process is assumed to be Poisson. Upon service initiation, the service representative will need access to the information system. This joint pool of information technology is capable of processing all transactions from different customers simultaneously. Notice that during times of high congestion, such central information systems respond with longer processing times. In other words, service times in the system are a function of the total number of customers being served in all channels. This characteristic is modeled as a processor sharing service discipline.

Let the information system be considered as a single server that processes at a constant rate of I service units per unit time. Assume that each customer in class k with $k = 1, \ldots, K$ has a service requirement that is exponentially distributed with an average service requirement of $1/\mu_k$. Then, letting $\mathbf{n} = (n_1, n_2, \ldots, n_K)$ denote the state vector, with n_k being the number of customers of class k in the system, the state dependent service rate for class k customers in the processor sharing system takes the form

$$\mu_k(\mathbf{n}) = \frac{I\, n_k\, \mu_k}{(n_1 + \ldots + n_K)}$$

for the loss system, and the form

$$\mu_k(\mathbf{n}) = \frac{I\, min(n_k, S_k)\, \mu_k}{(min(n_1, S_1) + \ldots + min(n_K, S_K))}$$

for the system with reneges. To simplify the notation and the analysis in the ensuing part of this paper, assume without loss of generality that $I = 1$. Define $\pi(\mathbf{n})$ as the equilibrium probability of being in state $\mathbf{n}$ (i.e., of having n_k customers of class k in the system). Define the sets $\mathcal{A} = \{\mathbf{n} \in \mathcal{Z}_+^K : n_k \leq T_k\}$ and $\mathcal{A}_k = \{\mathbf{n} \in \mathcal{A} : n_k < T_k\}$, where $\mathcal{Z}_+$ denotes the nonnegative integers. Finally, $\mathbf{e}_k$ is a K-dimensional vector of zeros with a one in its kth position and $\mathbf{0}$ is a K-dimensional vector of zeros.

In order to characterize the performance of the phone center, one must establish the behavior of the system in steady state. To this end, one must first determine the equilibrium distributions, $\pi(\mathbf{n})$, for the two systems being considered. In general, this distribution takes the form

$$\pi(\mathbf{n}) = \frac{\psi(\mathbf{n})}{\sum_{\mathbf{n} \in \mathcal{A}} \psi(\mathbf{n})} \tag{1}$$

where $G = \sum_{\mathbf{n} \in \mathcal{A}} \psi(\mathbf{n})$ is known as the normalization constant. For the derivation of equilibrium distributions in the loss and renege systems, the interested reader is referred to Akşin and Harker (1996a). Given the equilibrium distributions, the next step is to determine

revenue losses that result from congestion in the system. To this end, certain performance measures need to be established. Specifically, one would be interested in determining the probability of a customer being blocked upon arrival, as well as the loss that occurs due to reneging. In general, blocking probability in channel k is given by

$$B_k = 1 - \frac{\sum_{\mathbf{n}\in\mathcal{A}_k} \pi(\mathbf{n})}{\sum_{\mathbf{n}\in\mathcal{A}} \pi(\mathbf{n})}. \tag{2}$$

Reneges are the second source of customer loss, made up by the portion of customers that are lost after entering the system. In particular, the time a customer waits in queue k is assumed an exponential random variable with rate α_k. This implies a renege rate of $r_k(n_k) = \alpha_k(n_k - S_k)1(S_k < n_k \leq T_k)$ for $k = 1, \ldots, K$. Denote the long-run probability of renege for a customer of type k by R_k. Then,

$$R_k = \sum_{\mathbf{n}\in\mathcal{A}} \pi(\mathbf{n}) \frac{r_k(n_k)}{\sum_{k=1}^{K}(\mu_k(\mathbf{n}) + r_k(n_k) + \lambda_k)}. \tag{3}$$

3.2 The Staffing Model

The performance model proposed in the previous subsection provides a metric that can relate staffing decisions to overall profitability of the center. Making use of this metric, a model is developed that ensures optimal deployment of human resources within an existing phone center. In particular, it is assumed that the infrastructure of the center, i.e. telecommunication equipment and information technology, are already installed. The question that is addressed herein is what is the best way to staff this phone center.

Clearly, there are many ways of defining the optimality of a staffing plan. Frequently, cost minimization is taken as the objective. This approach fails to reflect the fact that revenues of a call center are a direct function of staffing decisions. One could attempt to characterize a customer service quality maximizing plan; however, this introduces the difficulty of measuring service quality. The objective herein is to determine economically optimal staffing levels. Economically optimal staffing levels are defined as those levels that maximize total profits of the phone center. In other words, they incorporate both the revenue and the cost dimension of the problem. In the ensuing discussion, it will be shown that economically optimal staffing levels can be determined for a loss system. For a system that allows for reneging customers, a heuristic that yields near optimal staffing will be proposed.

The static design problem ignores demand dynamics and considers a single time period. Given the demand rates for that particular period, the objective is to determine the number of service agents to be employed for each access channel such that systemwide profits are maximized. More specifically, the model can be stated as:

$$\max_{\mathbf{S}} \sum_{k=1}^{K} [v_k \lambda_k (1 - B_k(\mathbf{S}, \mathbf{T}, I))(1 - R_k(\mathbf{S}, \mathbf{T}, I)) - C^s(S_k)]$$

$$\text{s.t.} \quad S_k \leq T_k \quad k = 1, \ldots, K \tag{4}$$

$$S_k \quad \text{integer} \quad k = 1, \ldots, K, \tag{5}$$

where

$\mathbf{S} = (S_1, \ldots, S_K) \equiv$ server allocation vector;

$\mathbf{T} = (T_1, \ldots, T_K) \equiv$ trunk allocation vector;

$I \equiv$ processing rate for information system

$v_k \equiv$ revenue generated from type k customers;

$C^s(S_k) \equiv$ cost associated with keeping S_k agents of specialization k for a single time period.

The process model provides blocking and renege probabilities, B_k and R_k, for each access channel. The objective function denotes the profits of the center, accounting for the revenue of each call that is not lost as a function of blocking or reneging, and accounting for the cost of service representatives. For the case of a loss system, the objective function is modified as follows, since there is no customer loss from reneging in this instance:

$$\max_{\mathbf{S}} \sum_{k=1}^{K} [v_k \lambda_k (1 - B_k(\mathbf{S}, \mathbf{T}, I)) - C^s(S_k)]. \tag{6}$$

It is assumed that average revenue generated per call in each product line is given. On the cost side of the profit equation, the model accounts for human resource costs in the form of salaries. For the current analysis, it is assumed that the cost function C^s is linear in S_k. It would be straightforward to incorporate maintenance costs for the phone lines and technological resources. Since the current design problem assumes a given technology base, these costs will be fixed and are ignored herein.

The first set of constraints in the model ensure that the given number of phone lines in each department are not exceeded. This constraint set is imposed, based on the assumption

that every call class has its own designated set of phone lines. These can be replaced by

$$S_1 + \ldots + S_K \leq T,$$

if phone lines can be assigned to service agents flexibly, i.e. shared across departments, without having to designate a given number of phone lines to a certain department. This type of resource sharing is encountered in the call center studied in an earlier paper by the authors (Akşin and Harker 1996a), where it is shown that the earlier performance analysis still holds with a slightly modified state space.

This formulation constitutes the generic form of the static design problem. Application specific constraints and characteristics can be built-in as the model is operationalized. Even in its simple form, this problem is not easy to solve. In particular, one notes that the blocking and renege probabilities that appear in the revenue expression of the objective function are nonlinear in the number of servers. Part of this study will attempt to analyze the structure of this server allocation problem in further detail, and determine a solution algorithm to obtain the desired economically optimal staffing levels.

4 Properties of the Loss Case

The loss system constitutes the basic form of the performance model described above. While the system with reneges is the appropriate model for most retail banking phone center performance evaluations, which motivated the research in this paper, the loss system is of interest for other applications. Particularly for applications in telecommunication systems, data networks, and possibly as a model of phone center operations in discount brokerage with extremely impatient customers, the loss model constitutes an important case to be studied. Hence, this section focuses on the server allocation problem for the loss system.

As noted before, the analysis of the staffing problem is complicated by the presence of the blocking probability expression in the objective function. To deal with this difficulty, this section will start out with a brief review of structural properties of total throughput in such systems, identified by the authors in an earlier paper Akşin and Harker (1996a). The results are stated without proofs herein.

For simpler notation, let $w_i = v_i\lambda_i$ for $i = 1, \ldots, K$ and let $\rho = \lambda/\mu$. Then,

Proposition 1 *(Server Allocation) Assume that the total number of servers* $\sum_{k=1}^{K} S_k = S$ *is fixed. For any class* i *and* j *with* $w_i/\rho_i > w_j/\rho_j$ *the server allocation that maximizes the sum in (6) will have* $S_i \geq S_j$.

In order to state Proposition 2, the following definition is useful.

Definition 1 *(Definition 6.B.1 in Shaked and Shantikumar, 1994) Consider a family* $\{X(\theta),\ \theta \in \Theta\}$ *of random variables. Let* $\theta_i \in \Theta, i = 1,2,3,4$, *be any four values such that* $\theta_1 \leq \theta_2 \leq \theta_3 \leq \theta_4$ *and* $\theta_1 + \theta_4 = \theta_2 + \theta_3$. *If there exists four random variables* $\hat{X}_i$, $i = 1,2,3,4$, *defined on a common probability space, such that* $\hat{X}_i =_{st} X(\theta_i)$, $i = 1,2,3,4$, *and (i)* $\hat{X}_1 \leq min[\hat{X}_2, \hat{X}_3]$ *almost surely and (ii)* $\hat{X}_1 + \hat{X}_4 \leq \hat{X}_2 + \hat{X}_3$ *almost surely, then* $\{X(\theta), \theta \in \Theta\}$ *is said to be stochastically increasing and concave in the sample path sense.*

Making use of this definition, one can state Proposition 2 as follows.

Proposition 2 *(Monotonicity and Concavity) The weighted sum of throughputs in (6) is stochastically increasing and directionally concave in the sample path sense as a function of the server allocation vector* $\mathbf{S}$.

For a two class version of the loss model herein, De Waal and Van Dijk (1991) show that the throughput for a class of calls is monotonic in the number of servers allocated to that class. In other words, the throughput in Class 1 increases as one increases S_1 and holds S_2 constant, and decreases as one increases S_2 and holds S_1 constant. In Proposition 2, one is interested in the monotonicity of the sum of throughputs in all classes. Given the result for the throughput of a particular class, it is not obvious whether the increasing effect in one class will dominate the decreasing effects in all other classes. Proposition 2 states that this is the case.

In Federgruen and Groenevelt (1986), the authors identify structural properties of the objective function and the constraint set of a resource allocation problem, which ensure the optimality of a greedy resource allocation algorithm. Their results are reviewed next. Making use of the structural properties of throughput in the loss system, and the characteristics of the constraint set of the staffing problem herein, it is then shown that the server allocation problem for the loss system can be solved optimally using a greedy allocation algorithm.

First introduce some notation and preliminaries from Federgruen and Groenevelt (1986). Let $\mathbf{N}$ denote the set of nonnegative integers and let e^i for $i \in E$ be the ith unit basis vector of $\mathbf{N}^E$. For $x, y \in \mathbf{N}^E$, $x < y$ if $x \leq y$ and $x \neq y$. Let $\geq_R$ denote a complete order on $\mathbf{N}^E$. We write $x >_R y$ if $x \geq_R y$ and $y \not\geq_R x$. Some orders on $\mathbf{N}^E$ are *induced* by a real valued function f. In this case, $x \geq_R y$ if and only if $f(x) \geq f(y)$ $(x, y \in \mathbf{N}^E)$.

Again, from Federgruen and Groenevelt (1986) we have the following definition.

Definition 2 *A complete order R is called* weakly concave *if it satisfies*

$(R1)$ *if* $y \geq x, x \geq_R x + e^i$, *then* $y \geq_R y + e^i, i \in E$;

$(R2)$ *if* $y \geq x, x_i = y_i$ *and* $x + e^i >_R x + e^j$, *then* $y + e^i >_R y + e^j, i, j \in E$.

It is shown in Federgruen and Groenevelt (1986) that for resource allocation problems with objectives that are specified by a weakly concave complete order on R^E, and with constraints that determine a polymatroid, the greedy allocation procedure results in an optimal solution. In order to use this result in the current context, first note that the constraint set consisting of upper bounds on $\mathbf{S}$ induces a feasible region which is a polymatroid. Next, let $f(\mathbf{S}) = \sum_{i=1}^{K} TH_i(\mathbf{S})$ (for the loss system, where TH_i denotes the throughput in channel i and $\mathbf{S}$ is the server allocation vector), and show that the order induced by f is weakly concave. Note that staffing costs which are linear in the number of servers preserve the weak concavity of the objective function.

Proposition 3 *Total throughput, $f(\mathbf{S})$, as defined above, satisfies the conditions R1 and R2 inducing an order that is weakly concave.*

The proof of this proposition is given in the Appendix. This result will enable the use of a greedy allocation procedure to determine optimal staffing levels for the loss case.

This paper was motivated by staffing problems in retail banking call centers, where the system with reneges constitutes the realistic case. Hence, the remaining parts of the analysis will focus on a system with reneges. Implementation and testing of the greedy allocation scheme are not part of the analysis in this study. The paper by Federgruen and Groenevelt (1986) as well as references therein deal with this problem. It should be noted that the greedy allocation procedure for the loss system can be made more efficient by realizing

that Proposition 1 shown earlier reduces the number of possible server allocations for this system.

5 Properties of the Renege Case

Given the simplicity with which one can implement a greedy allocation scheme, it would be desirable to show the optimality of this algorithm for the case when calls can be put on hold and are allowed to renege. However, earlier experience with this system indicates that total throughput in this instance can exhibit irregular behavior with respect to the total number of servers. In Akşin and Harker (1996a), a numerical example is provided to illustrate that increasing the number of servers where servers are allocated optimally leads to non-monotonic system throughput that is clearly not concave. Furthermore, the qualitative characteristics of total throughput change as a function of system parameters.

Next, a numerical example is provided where condition R1 from the previous section is violated for a system with reneges. Consider a system with reneges where the server allocation vectors $\mathbf{S}^x$ and $\mathbf{S}^y$ correspond to x and y in Definition 2. Let $\mathbf{S}^x \geq_R \mathbf{S}^y$ if and only if $\sum_{k=1}^{K} TH_k(\mathbf{S}^x) \geq \sum_{k=1}^{K} TH_k(\mathbf{S}^y)$. In order to satisfy R1, the server allocation vectors need to satisfy the following condition:

$$\text{if } \mathbf{S}^y \geq_R \mathbf{S}^x, \ \sum_{k=1}^{K} TH_k(\mathbf{S}^x) \geq \sum_{k=1}^{K} TH_k(\mathbf{S}^x + e_i), \ \text{ then } \sum_{k=1}^{K} TH_k(\mathbf{S}^y) \geq \sum_{k=1}^{K} TH_k(\mathbf{S}^y + e_i).$$

For the numerical example, let $K = 3$, $\lambda = (1,1,1)$, $\mu = (5,4,3)$, $\alpha = (1,0.5,0.1)$, $T = (6,6,6)$, $\mathbf{S}^x = (1,1,1)$, and $\mathbf{S}^y = (1,2,2)$. Then $\mathbf{S}^x + e_1 = (2,1,1)$ and $\mathbf{S}^y + e_1 = (2,2,2)$. For this system, one can calculate throughputs with the different server allocation vectors as:

$$\begin{aligned}
\sum_{k=1}^{3} TH_k(\mathbf{S}^x) &= 2.9278 \\
\sum_{k=1}^{3} TH_k(\mathbf{S}^x + e_1) &= 2.91863 \\
\sum_{k=1}^{3} TH_k(\mathbf{S}^y) &= 2.95313 \\
\sum_{k=1}^{3} TH_k(\mathbf{S}^y + e_i) &= 2.9582.
\end{aligned}$$

Hence, condition R1 in Section 4 is violated. Given this example, one is unable to replicate the results for the loss system that were obtained in Section 4.

To get a better feel for the structural relationship between number of servers and throughput in a system with reneges, it is useful to focus on a single class system. By taking away the additional complexity of optimally allocating a given number of servers among different classes, and removing the interaction effect on performance in the various classes through the shared processor, one can focus on the relationship between number of servers, blocked calls, and reneging calls. Results from a set of numerical examples are shown in the following graphs. In all three examples, $\mu = 0.55$ and $\alpha = 2.51$. The first example has $\lambda = 1.0$, the second $\lambda = 10.0$, and the third $\lambda = 0.2$. Blocking probabilities, reneging probabilities, and throughput as a function of the number of servers S are plotted in Figure 1. For better readability, all points have been joined in the graphs. The column on the left depicts the case with $\lambda = 1.0$, the column in the middle the case with $\lambda = 10.0$, and the column on the right the case with $\lambda = 0.2$. It is clear from these examples that, while reneging probabilities decrease as the number of servers is increased, blocking probabilities increase; however, the shapes of the curves can differ depending on problem parameters. As a result, throughput can exhibit non-monotonic behavior, first increasing and then decreasing with additional servers as in the case of the first example.

It is not surprising that once the throughputs from each class are added and the impact of congestion in one class is felt by others, total system throughput exhibits behavior that is hard to characterize qualitatively. In particular, recall that in the original system, additional servers in one class act as a way of identifying the priority of a certain type of call by allowing a higher allocation of processing capacity to this call type. Hence, when one considers the entire system, blocking probabilities in each class can decrease or increase as a function of the number of servers allocated to that class, depending on congestion elsewhere in the system and problem parameters in general.

Before going on to the next section, where a staffing heuristic for the renege system is proposed, it is interesting to make the following observation. In many call centers, staffing is performed heuristically by managers, who draw on past experience to adjust the number of service representatives to changing call volumes. The typical way to proceed in this type of a decision making environment is to start adding staff, as abandoned calls are observed to increase. The above examples indicate that for a single access channel center that is experiencing significant blocked calls to start out with, adding additional servers will have a

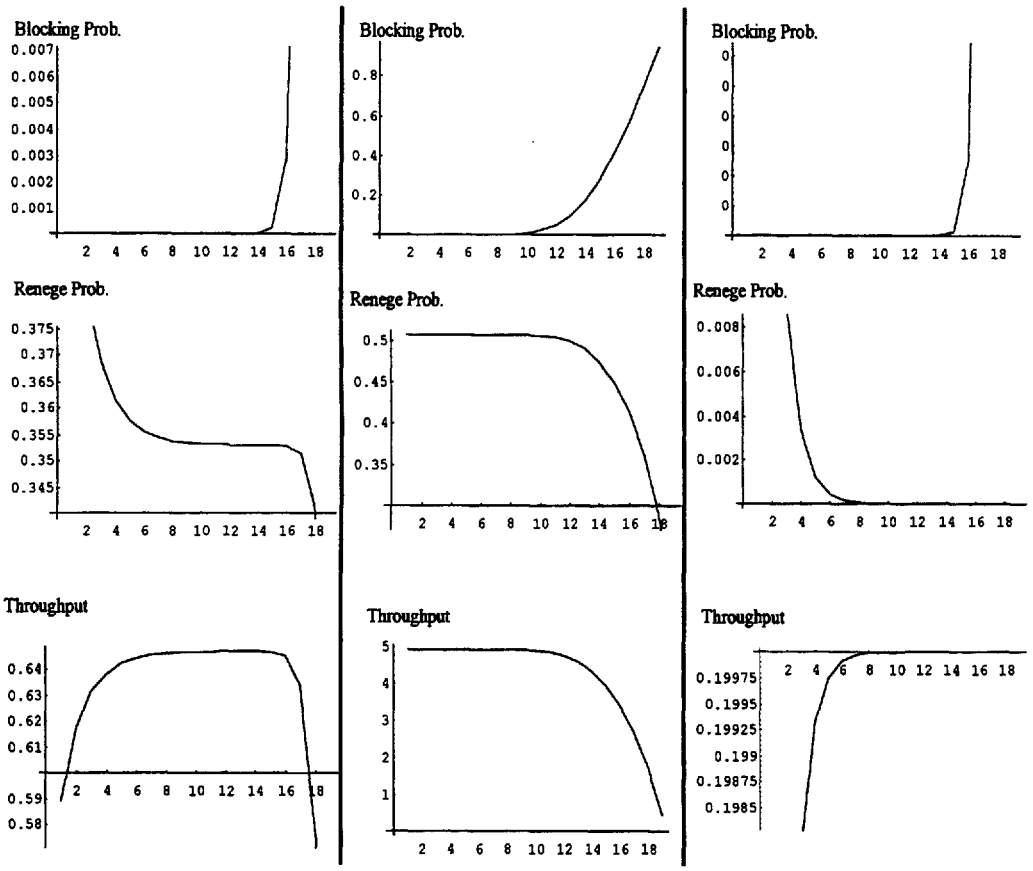

Figure 1: Blocking probabilities, renege probabilities, and throughput for first, second, and third examples (from left to right) as a function of number of servers.

deteriorating effect on throughput. For such systems, additional service representatives will decrease abandoned calls, and increase blocked calls, resulting in less overall throughput. For this type of a center, it may be more economical to add trunks, given the fact that a phone line is cheaper than a service representative. Clearly, this will still constitute a short term solution, since the true constraining resource in this instance is the computing resource. This type of a situation could be relevant for smaller centers, or for those that are experiencing rapid growth. It is clear that a similar situation could occur in a multi-class call center. However, disentangling the dynamics in this instance is much harder, given earlier observations on throughput and its dependence on problem parameters.

6 Staffing a Call Center

In this section, a heuristic to staff a call center is proposed. In terms of the model, this translates into optimizing the system with reneges. First, characteristics of a single class system are explored more formally. These characteristics verify that a simple search procedure yields the optimal number of servers in a system with single call types. Using this fact, a heuristic is developed for the multiple call type case, which decomposes a call center into single call type systems.

6.1 The Case with Single Call Types

Numerical examples in the previous section illustrated that throughput in a system with single call types exhibits unimodal behavior as a function of the number of servers. In other words, throughput is either increasing, decreasing, or starts out increasing and then decreases as the number of servers is increased. This structural characteristic of throughput is formally stated next, following some preliminary results. All of the proofs can be found in the Appendix.

Lemma 1 *Renege probability is monotonically nonincreasing in the number of servers S.*

Before stating the next lemma, the following definition from the book by Shaked and Shantikumar (1994) is useful.

Definition 3 *(Shaked and Shantikumar 1994)*
Consider a family $\{X(\theta), \theta \in \Theta\}$ *of random variables. Let* $\theta_i \in \Theta$, $i = 1,2,3,4$ *be any four values such that* $\theta_1 \leq \theta_2 \leq \theta_3 \leq \theta_4$ *and* $\theta_1 + \theta_4 = \theta_2 + \theta_3$. *If there exists four random variables* $\hat{X}_i$, $i = 1,2,3,4$ *defined on a common probability space, such that* $\hat{X}_i =_{st} X(\theta_i)$, $i = 1,2,3,4$ *and*
(i) $\max[\hat{X}_2, \hat{X}_3] \leq \hat{X}_4$ *a.s. and*
(ii) $\hat{X}_2 + \hat{X}_3 \leq \hat{X}_1 + \hat{X}_4$ *a.s., then* $\{X(\theta), \theta \in \Theta\}$ *is said to be stochastically increasing and convex in the sample path sense.*

Lemma 2 *The blocking probability,* $\{B(\theta), \theta \in \Theta\}$, *(with* $\theta = S$ *and* $\Theta = S : S < T$*) in a single class loss system with reneges, is stochastically increasing and convex in the sample path sense.*

Proposition 4 *Throughput in a single class system with reneging is unimodal as a function of the number of servers* S.

Unimodality of throughput is a useful property. For a single class system, this result enables identification of the optimal number of servers using a simple search procedure.

6.2 A Staffing Algorithm

Results from the previous subsection indicate that if one can decompose the call center into single class systems in a way that captures the interaction between classes in the original system, one can then independently determine the optimal staffing levels in each single class subsystem. This idea will be used in the algorithms proposed below. In particular, the algorithms will start by decomposing the original system into individual single class systems, will determine staffing levels for each single class system, and then go back to the original system and iterate the same procedure. The algorithms will stop once an equilibrium is attained. Slight variations in the way the system is decomposed and the individual class staffing is performed lead to three variations of this basic algorithm.

From earlier discussion, it is known that the interaction between classes in a call center can be captured as a change in the service rates. This is characterized by the state dependent service rates $\mu(\mathbf{n})$ in a renege system. Thus, a multi class renege system can be treated as

K independent single class systems, where service rates μ_k in each class are modified such that they are as close as possible to the original rates

$$\mu_k(\mathbf{n}) = \frac{\mu_k min(n_k, S_k)}{\sum_{i=1}^{K} min(n_i, S_i)}.$$

These rates change dynamically as the state vector $\mathbf{n}$ changes. The algorithms that follow will attempt to approximate these state dependent service rates in steady state. Three possibilities for decomposing are suggested, leading to the three algorithms described below.

Algorithm 4.1

Step 1 *Initialize: Set iteration count to* $j = 0$. *Select an initial server allocation vector* $\mathbf{S}^0 = (S_1^0, S_2^0, \ldots, S_K^0)$.

Step 2 *Determine new* $\mu_k's$: *For each class* $k = 1, \ldots, K$ *set* $\mu_k' = \mu_k S_k^j / (S_1^j + S_2^j + \ldots + S_k^j)$. *Set* $j := j + 1$.

Step 3 *Optimize each subsystem: For each subsystem* $k = 1, \ldots, K$ *with state dependent arrival rate* $\lambda_k(n_k)$, *service rate* μ_k', *and renege rate* α_k, *determine the number of servers that maximize profits in class* k *as* S_k^j.

Step 4 *Stopping condition: If* $\mathbf{S}^j = \mathbf{S}^{j-1}$ *stop. Else, go to Step 2.*

Step 2 of this algorithm constitutes the decomposition step. Note that the terms $min(n_k, S_k)$ in the state dependent service rate expression are approximated by S_k. Under a heavy traffic scenario, this would constitute an exact expression for $\mu_k(\mathbf{n})$ in steady state, however for lower traffic it constitutes an approximation. An alternative to Step 2 is proposed to identify another approximation which may be closer to the exact case, wherein the expected number of customers in each class X_k, are substituted for n_k in $min(n_k, S_k)$.

Algorithm 4.2

This algorithm is identical to Algorithm 4.1, except for Step 2.

Step 2′ : *Determine new* μ_k'*s: For each class* $k = 1, \ldots, K$ *set* $\mu_k' = \mu_k min(X_k, S_k)/(min(X_1, S_1) + \ldots + min(X_K, S_k))$, *where* X_k *is determined for each decomposed subsystem* $k = 1, 2, \ldots, K$. *Set* $j := j + 1$.

This new Step 2′ is still an approximation, since the X_k's are determined for the subsystems instead of the entire system and constitute an average for the term n_k. The advantage of this approximation is computational, since at each iteration of the algorithm, all computations are made for a single class system. It is obvious from the analysis in Akşin and Harker (1996a) that determining steady state distributions for a single class system is much simpler than in a multi-class system. Comparing it to the earlier algorithm, one notes that some additional computation is involved in an implementation that uses Step 2′, since this requires an additional determination of the expected number of customers in class k, X_k. However this additional effort is negligible, since the steady state probabilities $\pi_k(n_k)$ are already being computed for the optimization in Step 3 in the original algorithm.

In Step 3, both algorithms optimize the profits in each class, before proceeding to another iteration or stopping. This step performs a search for each class, until the individually optimal number of servers have been determined in each subsystem. In an effort to keep the decomposed system as close as possible to the original system, one can alter Step 3 slightly, requiring that only one iteration of the search procedure in Step 3 be performed before proceeding to Step 4. This procedure will reallocate the processor every time the server allocation is changed, thus possibly achieving a closer approximation to the original system. This version of the algorithm is the same as Algorithm 4.2, except for Step 3, which is restated below.

Algorithm 4.3

Same as Algorithm 4.2, except for Step 3:

Step 3′ *Perform one iteration of the search: For each subsystem, determine the profits for* S^{j-1}, $S^{j-1} + 1$, *and* $S^{j-1} - 1$. Set $S^j = \max\{S^{j-1}, S^{j-1} + 1, S^{j-1} + 1\}$.

The performance of all three algorithms are evaluated in the following section.

6.3 Numerical Examples

In the absence of standard test problems, and given the lack of earlier methods to determine optimal staffing levels in the system with reneges, performance of the algorithms are evaluated in two different ways in the sequel. First, for small three class systems, the optimal server allocation is determined by complete enumeration. For two sets of examples, these results are compared to the ones obtained by the proposed staffing algorithms in Table 1 and Table 2. The first column states the problem parameters that change between examples. In all examples reported in Table 1 and Table 2, the number of trunks is fixed at $\mathbf{T} = (8, 8, 8)$ and the cost per server is taken as 200 per time period. The revenue rates per call are taken as $\mathbf{v} = (1, 1, 1)$ for the examples in 1 and as $\mathbf{v} = (1.5, 1.5, 1.5)$ for the examples in 2. To convert the throughput rate to throughput per time period, the revenue term is multiplied by 43200 in both Table 1, and Table 2. In both cases, 43200 could be interpreted as the minutes in a month for a call center that operates 24 hours a day, seven days a week. The column labeled "Exact" reports the optimal server allocation vector obtained by complete enumeration. Once again, no approximations are used in computing these numbers. The second number in each cell of the tables is the percentage of the optimal system profits achieved by each server allocation. Thus 99.5 would indicate that system profits with the given server allocation are 99.5% of the optimal profits that can be obtained with an optimal server allocation. Columns three through five report the results obtained by Algorithms 4.1, 4.2, and 4.3 respectively. It should be noted that for certain examples, some of the algorithms do not converge to a solution (indicated by a $\star$), cycling between server allocations. Whenever this is the case, the best allocation in terms of system profits that the algorithm is cycling between is reported. The last column in the tables tabulates the server allocation obtained by assuming no processor sharing; that is, by optimizing the profits in each class individually.

Several observations can be made. For the first set of examples, tabulated in Table 1, the profits obtained by the various server allocations are very close to the optimum; hence, we see that the worst allocation reported in the table achieves 94.01% percent of the optimal profits. This is, in part, a function of our choice of problem parameters and also the result of having very small examples where a change in the number of servers translates into a very small change in throughput. In both sets of examples, none of the proposed algorithms are clearly dominant. Since all three algorithms evaluate performance in different classes independently, they are very fast. This suggests that one could easily determine the server

Table 1: Comparison of algorithms to exact optimum for systems with lower load and $v = (1, 1, 1)$

Parameters	Exact	Algorithm 1	Algorithm 2	Algorithm 3	Individual
$\lambda = (0.8, 0.5, 1.6)$ $\mu = (6, 6.5, 5.9)$ $\alpha = (0.75, 1.5, 0.55)$	(1,1,2) 100	$(2, 1, 4)^*$ 99.72	$(2, 1, 3)^*$ 99.85	$(3, 2, 4)^*$ 99.44	(1,1,2) 100
$\lambda = (2.0, 1.51, 1.1)$ $\mu = (6, 6.5, 5.9)$ $\alpha = (0.75, 1.5, 0.55)$	(4,3,2) 100	(4,5,4) 99.71	(4,4,2) 99.94	(4,4,2) 99.94	(2,2,1) 99.82
$\lambda = (2.0, 1.51, 1.1)$ $\mu = (1.6, 1.65, 2.59)$ $\alpha = (0.75, 1.5, 0.55)$	(2,1,1) 100	(1,2,4) 99.31	(1,3,4) 99.03	(1,3,4) 99.03	(3,5,3) 96.64
$\lambda = (0.8, 0.5, 1.6)$ $\mu = (1.6, 1.65, 2.59)$ $\alpha = (0.75, 1.5, 0.55)$	(1,1,4) 100	(4,3,1) 94.01	(4,3,2) 95.41	(4,3,2) 95.41	(3,2,4) 97.54
$\lambda = (2.0, 1.51, 1.1)$ $\mu = (6, 6.5, 5.9)$ $\alpha = (5, 2.5, 5.5)$	(4,3,2) 100	(5,5,4) 99.71	(5,5,3) 99.79	(5,5,3) 99.79	(3,2,2) 99.95
$\lambda = (0.8, 0.5, 1.6)$ $\mu = (6, 6.5, 5.9)$ $\alpha = (5, 2.5, 5.5)$	(1,1,3) 100	$(3, 2, 4)^*$ 99.60	$(2, 1, 3)^*$ 99.96	$(2, 2, 3)^*$ 99.83	(1,1,2) 99.97
$\lambda = (2.0, 1.51, 1.1)$ $\mu = (1.6, 1.65, 2.59)$ $\alpha = (5, 2.5, 5.5)$	(2,1,1) 100	(2,3,4) 99.42	(2,3,4) 99.42	(2,3,4) 99.42	(5,5,3) 98.62
$\lambda = (0.8, 0.5, 1.6)$ $\mu = (1.6, 1.65, 2.59)$ $\alpha = (5, 2.5, 5.5)$	(1,1,5) 100	(2,3,4) 98.99	(2,2,5) 99.42	(3,2,5) 99.08	(3,2,5) 99.08

Table 2: Comparison of algorithms to exact optimum for systems with higher load and $v = (1.5, 1.5, 1.5)$

Parameters	Exact	Algorithm 1	Algorithm 2	Algorithm 3	Individual
$\lambda = (9.0, 7.5, 3.5)$ $\mu = (8.6, 6.0, 3.0)$ $\alpha = (3.5, 1.5, 1.5)$	(2,1,1) 100	(2,1,1) 100	(2,1,1) 100	(2,1,1) 100	(4,2,4) 93.3
$\lambda = (9.0, 7.5, 3.5)$ $\mu = (4.5, 6.0, 6.0)$ $\alpha = (3.5, 1.5, 1.5)$	(1,1,1) 100	(1,1,5) 97.8	(1,1,4) 98.2	(1,1,4) 98.2	(2,2,5) 95.9
$\lambda = (4.0, 3.5, 6.0)$ $\mu = (8.6, 6.0, 3.0)$ $\alpha = (3.5, 1.5, 1.5)$	(4,1,1) 100	(5,3,1) 99.6	(5,3,1) 99.6	(5,3,1) 99.6	(5,5,1) 98.5
$\lambda = (4.0, 3.5, 6.0)$ $\mu = (4.5, 6.0, 6.0)$ $\alpha = (3.5, 1.5, 1.5)$	(1,1,2) 100	(3,3,1) 97.7	(3,3,1) 97.7	(3,3,1) 97.7	(5,5,4) 90.3
$\lambda = (9.0, 7.5, 3.5)$ $\mu = (8.6, 6.0, 3.0)$ $\alpha = (2.0, 1.0, 1.5)$	(1,1,1) 100	(1,1,1) 100	(1,1,1) 100	(1,1,1) 100	(3,2,4) 91.1
$\lambda = (9.0, 7.5, 3.5)$ $\mu = (4.5, 6.0, 6.0)$ $\alpha = (2.0, 1.0, 1.5)$	(1,1,1) 100	(1,1,5) 95.8	(1,1,4) 96.5	(1,1,4) 96.5	(1,2,5) 94.8
$\lambda = (4.0, 3.5, 6.0)$ $\mu = (8.6, 6.0, 3.0)$ $\alpha = (2.0, 1.0, 1.5)$	(2,1,1) 100	(5,1,1) 99.7	(4,2,1) 99.8	(4,2,1) 99.8	(5,5,1) 97.7
$\lambda = (4.0, 3.5, 6.0)$ $\mu = (4.5, 6.0, 6.0)$ $\alpha = (2.0, 1.0, 1.5)$	(1,1,2) 100	(2,2,1) 98.5	(2,2,1) 98.5	(2,2,1) 98.5	(5,5,4) 87.6

Table 3: Problem parameters for a realistic example

Parameter	1	2	3	4	5	6	7	8	9
λ	4.35	1.5	0.495	0.9	0.3	0.375	9.0	0.3	0.75
μ	0.48	2.8	18.0	8.0	28.0	6.0	0.75	30.0	30.0
α	5.455	0.75	5.0	3.33	2.857	2.5	6.0	6.0	6.0
v	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
T	216	216	216	216	216	216	132	12	7

allocation using all three, and then select the one that yields the highest profits.

It is useful to compare the results in Table 1 to those in Table 2. First note that by construction, most of the examples in Table 1 have lower values of λ/μ. One cannot quantify the exact differences in load, but qualitatively one can say that most of the examples in the first table represent systems with lower loads. Intuitively, one would expect the impact of the shared processor to be felt less for systems with lower loads. This intuition is supported by the examples in Tables 1 and 2. For the examples in Table 1, allocating servers independently ignoring the shared processor, as reported in the last column, results in a better server allocation compared to the three algorithms in four out of eight cases. For the examples in Table 2 on the other hand, where the load is typically higher, one of the proposed algorithms performs better than the individual optimization for all eight cases. For these examples, ignoring the shared processor typically results in over staffing compared to the optimal allocation. Similarly, the difference in performance in terms of percentages of optimal profits between the allocations generated by the algorithms and the individual optimization, can be as high as 10.9%.

Enumeration is not possible for larger systems. However, real-sized call centers are very large, typically having hundreds of trunks and service representatives. Hence, it is important to analyze the performance of the algorithms for larger problems. A larger problem is analyzed next, that is representative of a real call center. The data for this example is mostly taken from a call center of a major retail bank in the U.S. and is summarized in Table 3. The call center in this example has nine access channels, the last three of which are voice response unit (VRU) channels. The VRU channels provide fully automated service, hence are not allocated any servers. For these channels S_k and T_k are taken to be identical.

Table 4: Server allocations generated by the algorithms: the realistic problem

Algorithm	Server Allocation	Profit
Algorithm 1	S=(1,1,2,1,1,1,132,12,7)	988,214
Algorithm 2	S=(1,10,2,4,1,2,132,12,7)	999,911
Algorithm 3	S=(1,10,2,4,1,2,132,12,7)	999,911
Individual	S=(2,5,1,1,1,1,132,12,7)	991,639

Reneging can occur during service, implying that $r_k(n_k) = \alpha_k n_k$ for $k = 7, 8, 9$. These channels share the same processor as the rest of the channels, so their congestion will impact the server allocation generated by the various algorithms. The effect of the VRU's on the remaining system are captured in Step 2 of the algorithms, when resetting the μ_k''s. As in earlier examples, the cost of a server is taken as 200 per 43200 minutes.

The server allocations generated by the various algorithms are tabulated in the second column of Table 4. Once again, the allocation determined by optimizing each of the classes individually is also reported. In order to evaluate performance in the center with each allocation, the Monte- Carlo summation technique implemented in Akşin and Harker (1996) is used. This technique provides estimates and confidence intervals for blocking and reneging probabilities, as well as an estimate for system revenues, by sampling from a sampling function resembling the product-form solution. All results reported in this chapter are based on 100,000 iterations of the Monte-Carlo summation procedure. Importance sampling is performed heuristically. It should be noted that the results may change numerically with different number of iterations and different importance sampling parameters; however, qualitatively, they will be similar to the results reported herein. The third column in Table 4 reports the system profits for the different allocations.

Now consider the same center under increased traffic. Specifically, double all λ_k's in Table 3, yielding the new call arrival rates as

$$\lambda = (8.7, 3.0, 0.99, 1.8, 0.6, 0.75, 18.0, 0.6, 1.5).$$

The new server allocations and associated system profits for this example are tabulated in Table 5. All three algorithms and the individual optimization yield server allocations that result in system profits that are relatively close. In the first example, the algorithms do

Table 5: Server allocations generated by the algorithms: the case of increased traffic

Algorithm	Server Allocation	Profit
Algorithm 1	S=(1,1,2,1,4,1,132,12,7)	1,962,903
Algorithm 2	S=(1,1,2,5,1,3,132,12,7)	1,972,678
Algorithm 3	S=(1,1,2,5,2 ,3,132,12,7)	1,975,373
Individual	S=(1,15,1,2,1,1,132,12,7)	2,006,402

slightly better than the individual optimization, while in the second example the individual optimization yields the best allocation.

Looking at both examples, one can discern the following pattern. Whenever the arrival rate λ_k in a class exceeds its service requirement μ_k, specifically Class 1 in the first case, and Classes 1 and 2 in the second, once traffic intensities have been doubled, all three algorithms allocate a single server. Optimization of each class individually yields similar results for Class 1. For Class 2 in the second example, individual optimization allocates 15 servers. This is also the case when the individual optimization outperforms the algorithms. These results are consistent with the three examples in Section 5, where it was shown that with very high arrival rates compared to the service rate, throughput can be decreasing as a function of server allocation. However, given the high number of VRU trunks present in the current examples, one can conjecture that these high load classes should be allocated more servers than just one. Unless more servers are allocated to these classes, their priorities in terms of utilization of the shared processor, will be much lower than what is optimal. It seems that the algorithms, that are essentially all based on the optimization of an individual class, fail for this particular case. Specifically, the individual class optimization fails to capture the interaction effect between classes for this example, and due to the highly asymmetric nature of the parameters, the algorithms fail to rectify this as well. In order to test this conjecture and provide a server allocation that improves on those generated by the various algorithms, an improvement heuristic is proposed next.

The improvement heuristic takes a server allocation generated by one of the algorithms as a starting point and tries to improve on it by adding servers to those classes which have $\lambda_k > \mu_k$. Instead of using the performance of each individual class to identify improvements, it uses the original system's performance. In other words, it uses the Monte-Carlo summation

procedure to evaluate profits for every server allocation it considers. This algorithm is stated more formally below.

Algorithm 4.4

Step 0 *Take a server allocation generated by one of the earlier algorithms as initial point* $\mathbf{S}^0$. *Set iteration count to* $j = 1$. *Initialize the step size* σ_s *to some integer greater than or equal to one. Compute system profits* $P(S^0)$ *using the Monte-Carlo summation routine.*

Step 1 *For all classes* k *with* $\lambda_k > \mu_k$, *consider* $S_k^j = S_k^{j-1} + \sigma_s$.

Step 2 *For all these* k, *evaluate total system profit* $P(S_k^j)$ *using the Monte-Carlo summation routine. If* $P(S_k^j) \geq P(S_k^{j-1})$, set $S_k^j = S_k^{j-1} + \sigma_s$, $j = j + 1$.

Step 3 *If* $\mathbf{S}_k^{j-1} = \mathbf{S}_k^j$, *and* $\sigma_s \neq 1$ set $\sigma_s = \sigma_s - 1$. *Goto Step 1. If* $\sigma_s = 1$ *stop.*

The step size that is gradually reduced to one is useful in preventing the heuristic from settling on an early local optimum. Results, obtained by using the best of the server allocations generated by the algorithms and the server allocation obtained by individual class optimization as starting points, are reported in Table 6 for

$$\lambda^1 = (4.35, 1.5, 0.495, 0.9, 0.3, 0.375, 9.0, 0.3, 0.75)$$

and for

$$\lambda^2 = (8.7, 3.0, 0.99, 1.8, 0.6, 0.75, 18.0, 0.6, 1.5).$$

All of the examples use $\sigma_s = 5$ as a starting point for the step size. For both examples, the staffing that is obtained by the improvement heuristics is clearly better than those obtained by the algorithms or the individual optimization. It is also interesting to note that taking the initial server allocation as that which is generated by the best one of the algorithms in each case results in higher system profits than the case when the individual optimization allocation is taken as the starting point.

Table 6: The Improvement Heuristic

Arrival rate	Initial Allocation	Improved Allocation	Profit
λ^1	S=(1,10,2,4,1,2,132,12,7)	S=(15,10,2,4,1,2,132,12,7)	1,122,498
λ^1	S=(2,5,1,1,1,1,132,12,7)	S=(13,5,1,1,1,1,132,12,7)	1,087,880
λ^2	S=(1,1,2,5,2,3,132,12,7)	S=(14,16,2,5,2 ,3,132,12,7)	2,311,664
λ^2	S=(1,15,1,2,1,1,132,12,7)	S=(6,28,1,2,1,1,132,12,7)	2,258,461

6.4 Summary of Algorithm Results

Due to the large size of call center staffing problems in practice, one cannot use the entire system's performance at each iteration of an allocation procedure. To deal with such real-sized problems, a decomposition approach has been developed above, making use of the performance of individual classes to allocate servers. While this approach seems to work well for smaller and more symmetrical systems, it can understaff classes with high loads in the presence of preallocated VRU trunks that share the same processor. In this case, an improvement heuristic is proposed that makes use of the entire system's performance at each iteration, in order to allocate servers to a limited subset of the original channels. A routine that allocates servers to the entire call center using this latter approach would be computationally very expensive. However, by coupling the earlier staffing algorithms with the improvement heuristic, one is able to generate good server allocations in a reasonable time. The performance of the improvement heuristic is summarized in Table 7, where all results are reported as a percentage of the best solution found. Note that for these examples, the improvement heuristic, using as a starting point the best server allocation generated among the three algorithms, clearly outperforms the improvement heuristic which uses as a starting point the server allocation generated by the individual class optimization. Similarly, the three class examples in Table 2 indicate that whenever the impact of processor sharing is felt, the three decomposition algorithms perform significantly better than the individual class optimization.

The importance of selecting the right number of iterations and importance sampling parameters in the Monte-Carlo summation routine is apparent. While a lower number of iterations allows the improvement heuristic to terminate faster, the tradeoff is loss of accuracy in the estimation. Similarly, better importance sampling will result in variance

Table 7: Summary of results for the large problems

Arrival rate	Individual	Improved Individual	Algorithm 3	Improved Algorithm 3
λ_1	88.34%	96.92%	89.08%	100%
λ_2	86.79%	97.70%	85.45%	100%

reduction at each iteration, yielding tighter confidence intervals for blocking and reneging probabilities. These issues need to be investigated further in the context of the model herein, in order to improve the reliability of Monte-Carlo summation used within optimization procedures.

7 Concluding Remarks

This paper has looked at a static staffing problem for processor shared loss systems with multiple call types. For the pure loss case that does not allow for reneging behavior, it is shown that a greedy allocation scheme will yield the optimal staffing levels. It is furthermore established that for a single class renege system, the same result still holds. For multi-class renege systems, it is known from results in Akşin and Harker (1996a) that throughput can behave irregularly as a function of server allocation. For this latter problem, which also constitutes the relevant case for applications in call center management, heuristics are developed to solve the staffing problem.

None of the staffing algorithms clearly dominate; however, Algorithm 4.2 and Algorithm 4.3 seem to perform slightly better than Algorithm 4.1 in many of the examples. For a highly asymmetrical large example, it is shown that performance can be enhanced further through an improvement heuristic. While it seems that the allocations generated by the latter heuristic are relatively close to the optimal server allocation, further experimentation with real sized problems is necessary to gain better insight into its performance.

As noted above, the importance of improving the Monte-Carlo summation routine is obvious from its use in the improvement heuristic. In the current implementation, importance sampling parameters were fixed to a single value throughout the improvement heuristic. It seems to be the case that changing these parameters as a function of the server alloca-

tion would yield tighter confidence intervals at every iteration of the algorithm. There is a clear need for a more formal study of importance sampling for these systems. A Monte-Carlo summation routine that can automatically set its importance sampling parameters, preferably close to optimal, coupled with simulated annealing could lead to effective staffing routines like the improvement heuristic. Deriving expressions for confidence intervals for system revenues is another important issue for future research.

The staffing algorithms in this paper assume that phone trunks are designated for different call classes. Frequently in call centers, service representatives will be designated to specific call classes, however 800 trunks will be shared across the call center. To solve the staffing problem for this type of a center, one needs to determine the best way of allocating these shared trunks to specific call types in order to enable class-wise decomposition. It remains as a future research question to determine the best way of decomposing these systems with shared 800 trunks.

Staffing is a complex operations function, which consists of managing capacity in terms of human resources across time periods, locations and functions within an organization. The staffing problem in this paper is only a partial snapshot of the daily staffing decisions faced by a call center. The most natural extension of this research is one which looks at the staffing function across multiple time periods. In terms of the current analysis, this would translate into a series of staffing problems as the one herein for each time period, linked by staff movements like hiring, promotion, training, and firing. From a methodological perspective, this would involve a challenging optimization problem, given the increase in the state space of the problem, and the difficulty of maximizing a nonlinear, integer objective function across this enlarged state space.

Phone centers operate under highly seasonal and uncertain demand. The challenge of managing a phone center lies in the need to anticipate and respond to changing demand conditions by matching available supply to these needs such that service levels and call quality does not deteriorate. Thus, flexibility on the supply side is valued highly. It is in this setting, that cross-training practices have emerged as a powerful technique to balance supply and demand. Cross-training a service repesentative not only provides job enrichment for the representative, but also allows flexible assignments of agents to calls wherever the need arises. While the importance of cross-training practices have been suggested by many, formal analysis of its impact on operations in a call center have not been done. Solving the dynamic staffing problem proposed above would enable an analysis of cross-training

practices from an operations standpoint, identifying the appropriate scale and scope of this practice in a call center. This analysis would treat the problem from an operations perspective, and would not attempt to establish the behavioral or organizational implications of cross-training, which in itself remains an interesting question for further exploration.

The problem herein assumes that technology resources are given and fixed. An important issue in managing call centers today is determining implementation priorities; that is, determining when to restaff and when to resize technology. The current study could be used as a stepping stone for an exploration that looks at issues of sizing technology and prioritizing restructuring projects in call centers.

Finally, in an environment of call volume growth for call centers, it is important to make the distinction between the time to restructure an existing center and the time to open a new center. The model can be used to identify the costs of restructuring and the potential costs and benefits of opening a new center. In this regard, it can be used as a tool to aid decisions related to capacity upgrades and expansions. These issues will be analyzed further in future research.[1]

References

[1] Agnihothri, S.R. and Taylor, P.F. "Staffing a centralized appointment scheduling department in Lourdes Hospital". *Interfaces*, 21:5:1–11, 1991.

[2] Akşin, O.Z. and Harker, P.T. "Modeling a phone center: Analysis of a multi-channel, multi-resource,processor shared loss system". *Working Paper, Financial Institutions Center*, 1996.

[3] Akşin, O.Z. and Harker, P.T. "To sell or not to sell: Determining the tradeoffs between service and sales in retail banking call centers". *Working Paper, Financial Institutions Center*, 1996.

[4] Andrews, B. and Parsons, H. "Establishing telephone-agent staffing levels through economic optimization". *Interfaces*, 23:2:14–20, 1993.

[5] Boxma, O.J., Rinooy Kan, A.H.G. and van Vliet, M. "Machine allocation problems in manufacturing networks". *European Journal of Operational Research*, 45:47–54, 1990.

[1]This research was funded by a generous grant from the Alfred P. Sloan Foundation.

[6] Callahan, B.B. and Khan, M.R. "Planning laboratory staffing with a queueing model". *European Journal of Operational Research*, 67:321–331, 1993.

[7] Chaiken, J.M. and Larson, R.C. "Methods for allocating urban emergency units: a survey.". *Management Science*, 19:4:110–130, 1972.

[8] Dallery, Y. and Stecke, K. E. "On the optimal allocation of servers and workloads in closed queueing networks". *Operations Research*, 38:4:694–703, 1990.

[9] De Waal, P. "A constrained optimization problem for a processor sharing queue". *Naval Research Logistics*, 40:719–731, 1993.

[10] De Waal, P.R. and Van Dijk, N.M. "Monotonicity of performance measures in a processor sharing queue". *Performance Evaluation*, 12:5–16, 1991.

[11] Federgruen, A. and Groenevelt, H. "The greedy procedure for resource allocation problems: necessary and sufficient conditions for optimality ". *Operation Research*, 34:6:909–918, 1986.

[12] Foshini, G.J. and Gopinath, B. "Sharing memory optimally ". *IEEE Transactions on Communications*, COM-31:3:352–360, 1983.

[13] Frenk, H., Labbe, M., van Vliet, M., and Zhang, S. "Improved algorithms for machine allocation in manufacturing systems". *Operations Research*, 42:3:523–530, 1994.

[14] Frostig, E. "On the optimal assignment of servers in a two stations tandem queue with no intermediate waiting room". *Operations Research Letters*, 13:13–18, 1993.

[15] Gaballa, A. and Pearce, W. "Telephone sales manpower planning at Qantas ". *Interfaces*, 9:3:1–9, 1979.

[16] Green, L. and Kolesar, P. "The feasibility of one officer patrol in New York City". *Management Science*, 30:8:964–981, 1984.

[17] Harris, C.M., Hoffman, K.L., and Saunders, P.B. "Modeling the IRS Telephone Taxpayer Information System". *Operations Research*, 35:4:504–523, 1987.

[18] Krajewski, L.J. and Henderson, J.C. "Decision making in the public sector: an application of goal interval programming for disaggregation in the post office". *in Ritzman, L.P, Krajewski, L.J, Berry, W.L., Goodman, S.H., Hardy, S.T. and Vitt, L.D. (eds.) Disaggregation Problems in Manufacturing and Service Organizations, Martinus Nijhoff, Boston*, 1979.

[19] Larson, R.C. "Approximating the performance of urban emergency service systems". *Operations Research*, 23:5:845–868, 1975.

[20] Lee, H.F. , Srinivasan, M.M., and Yano, C.A. "Characteristics of optimal workload allocation for closed queueing networks". *Performance Evaluation*, 12:255–268, 1991.

[21] Mabert, V.A. "The detail scheduling of a part-time work force:A case study of teller staffing". *Decision Science*, 8:109–120, 1977.

[22] Mabert, V.A. "Staffing and equipment decisions for services: an experimental analysis ". *Journal of Operations Management*, 6:3:273–281, 1986.

[23] Maier-Rothe,C. and Wolfe,H.C. "Cyclical scheduling and allocation of nursing staff". *Socio-Economic Planning Sciences*, 7:5:471–487, 1973.

[24] McMahon, S.P. and Long, R.A. "Developing cost-effective branch-staffing strategies". *Bankers Magazine*, 174:1:68–75, 1991.

[25] Quinn, P., Andrews, B., and Parsons, H. "Allocating telecommunications resources at L.L. Bean, Inc.". *Interfaces*, 21:1:75–91, 1991.

[26] Rising, E.J. and Kaminsky,F.C. "Analytical scheduling of small nursing teams". *International Journal of Production Research*, 9:1:169–179, 1971.

[27] Shaked, M. and Shantikumar, J.G. *Stochastic Orders and their Applications*. Academic Press, Inc., New York, 1994.

[28] Shantikumar, J.G. and Yao, D.D. "Optimal server allocation in a system of multi-server stations". *Management Science*, 33:9:1173–1180, 1987.

[29] Shantikumar, J.G. and Yao, D.D. "On server allocation in multiple center manufacturing systems ". *Operations Research*, 36:2:333–342, 1988.

[30] Sze, D.Y. "A queueing model for telephone operator staffing". *Operations Research*, 32:2:229–249, 1984.

[31] Treleven, M. "A review of the dual resource constrained system research". *IIE Transactions*, 21:3:279–287, 1989.

[32] Van der Velde, M. "Teller staffing: Managing costs, stemming turnover". *Bank Administration*, pages 18–20, 1988.

[33] Vitt, L.D. "Multi-level police patrol planning". *Ritzman, L.P, Krajewski, L.J, Berry, W.L., Goodman, S.H., Hardy, S.T. and Vitt, L.D. (eds.) Disaggregation Problems in Manufacturing and Service Organizations, Martinus Nijhoff, Boston*, 1979.

[34] Yamazaki, G. and Sakasegawa, H. "An optimal design problem for limited processor sharing systems". *Management Science*, 33:8:1010–1019, 1987.

Staffing an Inbound Call Center
APPENDIX

Proof of Proposition 3

Proof of ($R1$). Let $\mathbf{S}$ and $\mathbf{U}$ be two allocation vectors with $\mathbf{U} \geq \mathbf{S}$. The total throughput function $f(\mathbf{S})$ is directionally concave in the server allocation vector $\mathbf{S}$, as stated in Proposition 2. It is well known that a directionally concave function is submodular. For the total throughput function, this implies by the definition of submodularity that

$$\sum_k TH_k(\mathbf{S}+e_i+e_j)+\sum_k TH_k(\mathbf{S}) \leq \sum_k TH_k(\mathbf{S}+e_i)+\sum_k TH_k(\mathbf{S}+e_j). \tag{7}$$

To show ($R1$), one needs to show that for $\mathbf{U} \geq \mathbf{S}$, if $\sum_k TH_k(\mathbf{S}) \geq \sum_k TH_k(\mathbf{S}+e_i)$, then one also has $\sum_k TH_k(\mathbf{U}) \geq \sum_k TH_k(\mathbf{U}+e_i)$. Using this in (7), one notes that given

$$\sum_k TH_k(\mathbf{S}) \geq \sum_k TH_k(\mathbf{S}+e_i), \tag{8}$$

one will have

$$\sum_k TH_k(\mathbf{S}+e_i+e_j) \leq \sum_k TH_k(\mathbf{S}+e_j). \tag{9}$$

Now letting $\mathbf{S}' = \mathbf{S}+e_j$ in (9), note that this would constitute the desired result if $\mathbf{S}'$ were equal to $\mathbf{U}$. For the case when this is not true, a recursive argument, where one replaces $\mathbf{S}$ with the $\mathbf{S}'$ thus obtained, and goes through the above argument will yield the desired result after a finite number of iterations. In the final round of this recursive argument, one will have $\mathbf{S}' = \mathbf{U}$ and hence the desired result.

Proof of ($R2$). For $\mathbf{U} \geq \mathbf{S}$, it is given that $S_i = U_i$ and also that

$$\sum_k TH_k(\mathbf{S}+e^i) \geq \sum_k TH_k(\mathbf{S}+e^j). \tag{10}$$

To verify that $R2$ holds, it is necessary to show that the above two conditions imply

$$\sum_k TH_k(\mathbf{U}+e^i) \geq \sum_k TH_k(\mathbf{U}+e^j). \tag{11}$$

From (10), it is known that the increase in throughput from adding a server to class i results in a higher net increase in total throughput than an additional server in class j. More formally, let $\delta^+ TH_i(S_i)$ denote the increase in the throughput of class i when one server is

added to S_i existing servers. Similarly, let $\delta^-_{(i)}TH_i(S_i)$ denote the decrease in throughput of all classes $k \neq i$ when a server is added to class i and all other server allocations remain the same. Using this new notation, (10) implies that

$$\delta^+TH_i(S_i) - \delta^-_{(i)}TH_i(S_i) \geq \delta^+TH_j(S_j) - \delta^-_{(j)}TH_j(S_j). \tag{12}$$

From Proposition 2, it is known that total throughput in the loss system is increasing and directionally concave in $\mathbf{S}$. It is well known that directional concavity implies concavity in each S_k for $k = 1, \ldots, K$. Clearly $TH_i(\mathbf{U}) \leq TH_i(\mathbf{S})$ since $\mathbf{U} \geq \mathbf{S}$ and additional servers are added to all classes except class i (i.e. $S_i = U_i$). But then, by concavity, $\delta^+TH_i(U_i) \geq \delta^+TH_i(S_i)$. Again, by concavity, one will have $\delta^+TH_j(U_j) \leq \delta^+TH_j(S_j)$ since $U_j \geq S_j$. Similarly, $\delta^-_{(i)}TH_i(U_i) \leq \delta^-_{(i)}TH_i(S_i)$ and $\delta^-_{(j)}TH_j(U_i) \geq \delta^-_{(j)}TH_j(S_j)$. Putting all of this together, one has

$$\delta^+TH_i(U_i) - \delta^-_{(i)}TH_i(U_i) \geq \delta^+TH_i(S_i) - \delta^-_{(i)}TH_i(S_i), \tag{13}$$

and

$$\delta^+TH_j(U_j) - \delta^-_{(j)}TH_j(U_j) \leq \delta^+TH_j(S_j) - \delta^-_{(j)}TH_j(S_j). \tag{14}$$

The desired result in (11) follows from (12),(13), and (14). □

Proof of Lemma 1

Consider two systems, one with S servers and the other with $S + 1$ servers, otherwise identical. Denote the equilibrium distribution of the first system by

$$\pi(n) = \frac{\psi(n)}{G},$$

and the second one as

$$\pi'(n) = \frac{\psi'(n)}{G'}.$$

Similarly the weights

$$\frac{r_k(n_k)}{\sum_{k=1}^{K}(\mu_k(\mathbf{n}) + r_k(n_k) + \lambda_k)}$$

in the expression for renege probability in (3), where $K = 1$ for the current single class system, are labeled as $\omega(n)$ and $\omega'(n)$ respectively for the two systems. It is first shown that for $n = 0, 1, \ldots, S$, $\psi(n) = \psi'(n)$, and for $n = S+1, S+2, \ldots, T$ $\psi'(n) \geq \psi(n)$. Note that $\omega(n) \geq \omega'(n)$ for all $n = 0, \ldots, T$. For a system with single call types, the ψ function takes the following form

$$\psi(n) = \frac{(min(n,S))!\lambda^n(min(n,S))^{(n-S)^+}}{\prod_{j=1}^{n}(\mu + r(j))min(j,S)}. \tag{15}$$

For $n \leq S$ one will have

$$\psi(n) = \frac{\lambda^n}{\mu^n} = \psi'(n),$$

and for $n > S$

$$\psi'(n) = \frac{\mu + (n-S)\alpha}{\mu}\psi(n),$$

or equivalently $\psi'(n) \geq \psi(n)$. The weights ω take the form

$$\omega(n) = \frac{\alpha(n-S)^+}{\alpha(n-S)^+ + \lambda + \mu} > \frac{\alpha(n-S-1)^+}{\alpha(n-S-1)^+ + \lambda + \mu} = \omega'(n).$$

Let $\delta_n = \psi'(n) - \psi(n)$ with $n = 0, 1, \ldots, T$. To prove the desired result, one needs to show that

$$\frac{\sum_{n\in\mathcal{A}} \omega'(n)\psi'(n)}{\sum_{n\in\mathcal{A}} \psi'(n)} \leq \frac{\sum_{n\in\mathcal{A}} \omega(n)\psi(n)}{\sum_{n\in\mathcal{A}} \psi(n)}. \tag{16}$$

Equation (16) can equivalently be stated as

$$\frac{\sum_{n\in\mathcal{A}} \omega'(n)\psi(n) + \sum_{n\in\mathcal{A}} \omega'(n)\delta(n)}{\sum_{n\in\mathcal{A}} \psi(n) + \sum_{n\in\mathcal{A}} \delta(n)} \leq \frac{\sum_{n\in\mathcal{A}} \omega(n)\psi(n)}{\sum_{n\in\mathcal{A}} \psi(n)},$$

or as

$$(\sum_{n\in\mathcal{A}} \omega'(n)\psi(n) + \sum_{n\in\mathcal{A}} \omega'(n)\delta(n))(\sum_{n\in\mathcal{A}} \psi(n)) \leq \sum_{n\in\mathcal{A}} \omega(n)\psi(n) \sum_{n\in\mathcal{A}} \psi(n) + \sum_{n\in\mathcal{A}} \omega(n)\psi(n) \sum_{n\in\mathcal{A}} \delta(n). \tag{17}$$

Apart from the difference in the weights, both sides of equation (17) are equal. Since $\omega'(n) \leq \omega(n)$ for all $n = 0, 1, \ldots, T$, the desired result in (16) follows. □

Proof of Lemma 2

In order to verify Lemma 2, note that the blocking probability in a single class system with reneges is identical to the blocking probability in a single class loss system with no reneges, where state dependent service rates $\mu(n)$ have been set to $\mu(n) = \mu + \alpha(n-S)^+$. Using this equivalence, the result is verified for a loss system with the appropriate modification in state dependent service rates. Consider four identical single class loss systems, with $\theta_1 = S$, $\theta_2 = S+1$, $\theta_3 = S+2$, and $\theta_4 = S+3$. Let the service requirement μ be the same in all four systems. Also let $\nu = sup_{n,i}\mu(n,i)$, where $\mu(n,i)$ denotes the state dependent service rate in system i when the state of the system is n. Let $\{\tau_n''\}$ be the sequence of Poisson event epochs with rate ν, which are independent of $\{\tau_n'\}$ and let $\{\tau_n, n \geq 1\} = \{\tau_n', n \geq 1\} \bigcup \{\tau_n'', n \geq 1\}$. Generate $\{U_n\}$, an iid sequence of uniform random variables on $[0, \nu]$, independent of $\{\tau_n\}$. Denote the number of customers in the ith system at time τ_n as $X^i(\tau_n)$. Also let $B^i(\tau_n)$ be

the cumulative number of blocked customers, in system i, at time τ_n. Let $X^i(0) = 0$ and $B^i(0) = 0$ for all $i = 1, 2, 3, 4$. In order to verify the lemma, one needs to show that the following conditions hold for all τ_n, $n \geq 0$:

$$\max[B^2(\tau_n), B^3(\tau_n)] \leq B^4(\tau_n), \tag{18}$$

$$B^2(\tau_n) + B^3(\tau_n) \leq B^1(\tau_n) + B^4(\tau_n). \tag{19}$$

In addition, the following condition on the states is imposed:

$$X^1(\tau_n) \leq X^2(\tau_n) \leq X^3(\tau_n) \leq X^4(\tau_n). \tag{20}$$

Notice that all three conditions are satisfied at $n = 0$. Next assume that they hold for τ_n. To complete the proof, one needs to show that (18), (19), and (20), still hold at τ_{n+1}. The uniformized Markov chains $B^i(\tau_n)$ and $X^i(\tau_n)$, $n \geq 1$, can be constructed as follows.

Case 1: Arrivals

$$B^i(\tau_{n+1}) = B^i(\tau_n) + (X^i(\tau_n) + 1 - \theta^i)^+ \tag{21}$$

$$X^i(\tau_{n+1}) = \min(X^i(\tau_n) + 1, \theta^i) \tag{22}$$

By construction, it is obvious that conditions (18) and (20) will be preserved at arrival epoch $n + 1$. The verification of the condition

$$B^2(\tau_{n+1}) + B^3(\tau_{n+1}) \leq B^1(\tau_{n+1}) + B^4(\tau_{n+1})$$

is identical to the one for the arrivals case in the proof of Proposition 3 in an earlier paper by the authors (Akşin and Harker (1996a)), thus is omitted herein. Having shown that (18), (19), and (20) are preserved at arrival epochs, the case at departure epochs is considered next.

Case 2: Departures

$$B^i(\tau_{n+1}) = B^i(\tau_n) \tag{23}$$

$$X^i(\tau_{n+1}) = X^i(\tau_n) - \mathbf{1}^i \quad i = 1, 2, 3, 4, \tag{24}$$

where

$$\mathbf{1}^i = \mathbf{1}\{U_n \in (0, \mu + \alpha(X^i - \theta^i)^+]\}.$$

The number of calls blocked will not change at departure epochs. Hence, it is sufficient to show that (20) is satisfied at each departure epoch.

First consider the case when $X^i(\tau_n) = X^{i+1}(\tau_n)$, $i = 1, 2, 3, 4$. Given the definition for $\mathbf{1}^i$s, this implies that

$$\begin{aligned} \text{if } \mathbf{1}^{i+1} = 1 &\rightarrow \mathbf{1}^i = 1 \\ \text{if } \mathbf{1}^{i+1} = 0 &\rightarrow \mathbf{1}^i = 0 \text{ or } 1. \end{aligned}$$

It is then clear that $X^i(\tau_{n+1}) \leq X^{i+1}(\tau_{n+1})$ for $i = 1, 2, 3$, as desired. Next consider the case when $X^i(\tau_n) < X^{i+1}(\tau_n)$. Again, using the definition for $\mathbf{1}^i$s, this yields the conditions

$$\begin{aligned} \text{if } \mathbf{1}^i = 1 &\rightarrow \mathbf{1}^{i+1} = 0 \\ \text{if } \mathbf{1}^i = 0 &\rightarrow \mathbf{1}^{i+1} = 0 \text{ or } 1, \end{aligned}$$

ensuring that $X^i(\tau_{n+1}) \leq X^{i+1}(\tau_{n+1})$ for $i = 1, 2, 3$. □

Proof of Proposition 4

To verify the unimodality of throughput, it is sufficient to show that once throughput starts to decrease as a function of the number of servers, it will keep decreasing as S is increased. First recall that throughput for a single class system can be expressed in terms of the blocking and renege probabilities as follows:

$$TH(S) = \lambda(1 - B(S))(1 - R(S)).$$

Consider three systems, with $S^1 = S$, $S^2 = S + 1$, and $S^3 = S + 2$ servers, respectively. To simplify the ensuing notation, let $\bar{B}(S^i) = (1 - B(S^i))$, and $\bar{R}(S^i) = (1 - R(S^i))$ for $i = 1, 2, 3$. Also let

$$\begin{aligned} d_{R^i} &= R(S^i) - R(S^{i+1}) \quad i = 1, 2; \\ d_{B^i} &= B(S^{i+1}) - B(S^i) \quad i = 1, 2. \end{aligned}$$

With this new notation, throughput can be restated as

$$TH(S) = \lambda \bar{B}(S)\bar{R}(S). \tag{25}$$

Given Lemma 1, $\bar{R}(S^i)$ will be monotonically nondecreasing as a function of i. Similarly Lemma 2 implies that $\bar{B}(S^i)$ will be monotonically nonincreasing as a function of i.

As noted initially, to prove the unimodality of throughput, it is sufficient to show that if

$$\bar{R}(S^1)\bar{B}(S^1) \geq \bar{R}(S^2)\bar{B}(S^2), \tag{26}$$

then

$$\bar{R}(S^2)\bar{B}(S^2) \geq \bar{R}(S^3)\bar{B}(S^3). \tag{27}$$

Some algebraic rearrangement yields the equivalent of (26) as

$$\bar{B}(S^1)d_{R^1} \leq \bar{R}(S^2)d_{B^1}. \tag{28}$$

Due to the convexity of blocking probabilities as a function of the number of servers, $d_{B^1} \leq d_{B^2}$. Monotonicity properties imply that $\bar{R}(S^3) \geq \bar{R}(S^2)$, and $\bar{B}(S^2) \leq \bar{B}(S^1)$. The nature of the change in d_{R^1} is not known. One could have $d_{R^1} \geq d_{R^2}$ or $d_{R^1} \leq d_{R^2}$, depending on problem parameters. Consider the case when $d_{R^1} \geq d_{R^2}$. Observe that all terms on the left hand side of (28) are decreasing as a function of i, and all terms on the right hand side are increasing in i. Using this observation, one can state that for the case when $d_{R^1} \geq d_{R^2}$, given (28), equation (27) will always hold. This proves the unimodality of throughput for the first case.

Next consider the case when $d_{R^1} \leq d_{R^2}$. All of the other relationships remain the same. In order to claim that the condition in (28) is preserved when the number of servers is increased, one needs to show that the increase in d_{R^1} on the left hand side is less than the increase in $(\bar{R}(S^2))$ on the right hand side. Notice that

$$\bar{R}(S^3) - \bar{R}(S^2) = R(S^2) - R(S^3), \tag{29}$$

$$d_{R^2} - d_{R^1} = (R(S^2) - R(S^3)) + (R(S^2) - R(S^1)). \tag{30}$$

By earlier shown monotonicity, $(R(S^2) - R(S^1)) \leq 0$. But then (29) will always be greater than (30), yielding the desired result. This concludes the proof of unimodality of throughput for the case when $d_{R^1} \leq d_{R^2}$, yielding the desired result in Proposition 4. □