Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Routing and Staffing in Customer Service Chat Systems with Impatient Customers

Tolga Tezcan, Jiheng Zhang

To cite this article:
Tolga Tezcan, Jiheng Zhang (2014) Routing and Staffing in Customer Service Chat Systems with Impatient Customers. Operations Research

Published online in Articles in Advance 11 Jun 2014

. http://dx.doi.org/10.1287/opre.2014.1284

OPERATIONS RESEARCH
Articles in Advance, pp. 1–14
ISSN 0030-364X (print) | ISSN 1526-5463 (online)

http://dx.doi.org/10.1287/opre.2014.1284

Routing and Staffing in Customer Service Chat Systems with Impatient Customers

Tolga Tezcan
University of Rochester, Simon School of Business, Rochester, New York 14627, tolga.tezcan@simon.rochester.edu

Jiheng Zhang
Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong, j.zhang@ust.hk

We consider customer service chat (CSC) systems where customers can receive real time service from agents using an instant messaging (IM) application over the Internet. A unique feature of these systems is that agents can serve multiple customers simultaneously. The number of customers that an agent is serving determines the rate at which each customer assigned to that agent receives service. We consider the staffing problem in CSC systems with impatient customers where the objective is to minimize the number of agents while providing a certain service level. The service level is measured in terms of the proportion of customers who abandon the system in the long run. First we propose effective routing policies based on a static planning LP, both for the cases when the arrival rate is observable and for when the rate is unobservable. We show that these routing policies minimize the proportion of abandoning customers in the long run asymptotically for large systems. We also prove that the staffing solution obtained from a staffing LP, when used with the proposed routing policies, is asymptotically optimal. We illustrate the effectiveness of our solution procedure in systems with small to large sizes via numerical and simulation experiments.

Subject classifications: call center management; queuing theory and stochastic methods.
Area of review: Stochastic Models.
History: Received August 2012; revision received June 2013; accepted January 2014. Published online in *Articles in Advance*.

1. Introduction

Customer service chat (CSC) is a service provided by companies to allow users to communicate in real time with a customer service agent using the Internet. It is based on an instant messaging (IM) application that is usually accessible from a company's website. IM applications are text-based one-on-one interaction tools that allow individuals to communicate with each other while they perform other computer-based tasks. The CSC service is usually a part of a company's customer contact center that also includes telephone and email customer service. In terms of responsiveness, CSC service can be placed between telephone and email customer service. Unlike traditional email service (which lacks immediacy and interactivity), it is real time. However, it is somewhat slower compared to the telephone service since a user still has to wait for the agent to read each message and type the response; see Koyama (1998) and Broughton (2001).

CSC systems have been shown to be operationally less costly than telephone customer service support and about the same as email support; see Bannan (2000) and Andrews and Haworth (2002). Moreover, Shae et al. (2007) demonstrate that a CSC system can perform better than a telephone service can across different key performance measures, including first call resolution and end user satisfaction. Because of their unique features, such as their ability to allow collaborative browsing or screen sharing, the CSC systems are considered to be an especially important way to interact with customers for computer and software companies, Web based companies (such as retail e-commerce), and libraries to provide reference service, among others; see Francoeur (2001), Bannan (2000), and TELUS International (2011). In addition, customers can carry out other tasks on their computers while the agent is trying to figure out a solution for their problem. CSC systems also have some certain disadvantages such as the technical proficiency required from customers, the lack of a natural conversation setting in a chat environment, and the interaction being slowed down by the typing of messages, which leads to users losing patience and sometimes abruptly terminating their chat sessions; see Koyama (1998) and Oder (2001). However, there is a growing industry of companies providing software applications for CSC systems, making them more accessible for customers and easier to manage for service providers, see for example TELUS International (2011). Because of its popularity among the computer savvy younger generation, see TELUS International (2011), its prevalence in customer service is only expected to increase in the future.

As in other customer service systems, the quality of service is an important determinant of the success of CSC systems, and better service quality helps overcome some of the aforementioned obstacles in providing effective chat service. In this paper we consider the staffing and routing problems in CSC systems with impatient customers with the

objective of providing good quality of service. That agents can serve multiple customers provides a unique challenge for the management of these systems. The current literature on management of service systems mainly focuses on systems in which an agent only helps one customer at a time; see Gans et al. (2003). There is a vast literature on processor sharing queues (where an agent can serve multiple customers simultaneously) that can be found mainly in communications literature; see Zhang et al. (2009) and the references therein. However, the CSC systems have different features that are not covered by this literature. Specifically, the main assumption in this literature is that there is a single server (or a pool of servers that work jointly) serving all or a subset of customers in the system. In a CSC system, there is a pool of agents with limited parallel processing capability, and each customer interacts with the same agent until that customer leaves the system. Therefore, we will build new analytical tools to analyze these systems.

Impact of multitasking on service systems and the routing problem: Before we study how to manage CSC systems, it is important to highlight the effect of multitasking on the productivity of agents. Intuitively, chatting with more customers should increase the productivity of the agent and it should also increase the time to complete service, but it is not clear to what extent. To the best of our knowledge there is no comprehensive empirical study on the effect of multitasking in chat service systems. Shae et al. (2007) demonstrate that agents can handle three simultaneous chat sessions without performance degradation (they recommend experimenting with more chat sessions until there is performance degradation but they stop at three). Based on data from a company that is operating a large IM-based service center, Luo and Zhang (2013) observe that the total service rate of an agent is not even monotone in the number of customers the agent is helping. They observe that the average total service rate becomes lower when an agent helps four customers instead of three but it becomes higher than the service rate with three customers when the agent helps five customers (p. 329, Figure 1). Given that agents have different service rates at different levels of multitasking, how to route the arriving customer to available agents can significantly affect the system performance. Thus, routing is quite an important decision to meet a certain service level as well as minimizing the staffing level.

In general, the main advantage of multitasking in a service system is that a multitasking agent can switch between tasks whenever the current task the agent is doing is not available or when some feedback is needed from the customer or another resource, e.g., an agent waiting for a response from a customer while providing chat service, a physician waiting for the result of lab tests in an emergency room. However, the effects of multitasking on the speed of carrying out a task can be different in different contexts. Because of cognitive switching costs, it has been observed in experiments that multitasking may lower productivity and quality in completing certain tasks (see Gladstones et al. 1989 and Pashler 1994). Similar effects have also been observed in service systems, referred to as slow-down in Bob and Terwiesch (2013). Surprisingly, increased multitasking or load may also increase the speeds of carrying out certain tasks, referred to as rushing, because of, for example, increased pressure to complete the tasks. There are several papers that provide empirical evidence that rushing exists in service systems; specifically, Hasija et al. (2010) observe rushing behavior of agents providing email service, KC and Terwiesch (2009) of hospital employees responsible for transporting patients, Tan and Netessine (2014) of servers in a restaurant chain, and Staats and Gino (2012) of workers in a production simulation game.

In summary, there seems to be no universal trend in agent efficiency while multitasking (and so in service rates in chat service systems). Because of slow-down, the service rates tend to decrease but because of rushing and reduced waiting for the customer responses, the service rate may increase with increased multitasking. In addition, if the quality of service and its effect on service times are taken into account (for example, customers are unlikely to end a chat session until their problems are resolved successfully), it is less likely that a universal trend is ubiquitous in all applications. Therefore we analyze chat systems when increased multitasking may have effects of varying degrees on service times.

Our contribution: To incorporate the effect of multitasking in our model, we assume that the service rate of an agent is determined by the number of customers that agent is serving. (We use the term "level i" to refer to the activity (or task) of helping i customers at the same time, and an agent is said to be at level i if that agent is chatting with i customers.) Our main goal is to find effective routing policies and then the minimum staffing level in a CSC system to meet a service level requirement given the arrival rate and service speeds. The service level is measured in terms of the abandonment probability of a customer. En route to achieve this goal, we first identify conditions that determine what level of multitasking is *efficient* (or inefficient). These conditions are based on the optimal solution of a static planning problem, which we refer to as the routing LP. Given a fixed arrival rate and a staffing level, the routing LP determines how many customers each agent should help on average in the long run. We prove using the dual of this LP that inefficient levels should never be used, providing guidance on how to take advantage of the multitasking capability of agents.

Although the routing LP provides a solution for the long-run allocation of agents into different levels, it is not clear how CSC systems can be managed in real time (by allocating arriving customers to available agents) to achieve such long-run performance. This becomes especially challenging when the arrival rate is not observable. We next focus on the routing problem to determine to which agent an arriving customer should be routed. We propose two policies; one that assumes the basic levels (levels that the routing LP allocates agents to) are known, and a second one that is somewhat similar but estimates the basic levels seamlessly

and automatically based on a virtual system. The virtual system is similar to the CSC model but allows a certain type of preemption that is not possible (or desirable) in actual CSC systems. We prove that the proposed policies are asymptotically optimal in minimizing the probability of abandonment in steady state asymptotically for large systems when service and abandonment times are exponential. In addition, we provide conditions for a simple static priority policy, referred to as lightest-load-first, that routes a customer to the least busy agent to be asymptotically optimal. Once effective routing policies are identified, we turn our attention to the staffing problem. We formulate another LP, referred to as the staffing LP, whose optimal solution gives our proposed staffing solution. We prove that the proposed staffing levels are asymptotically optimal for large CSC systems when customers are routed according to the policies we propose.

Our solution approach is based on a "fluid limit" analysis similar to Atar et al. (2010, 2011), Behzad and Tezcan (2012), and also Bassamboo and Randhawa (2010) and Bassamboo et al. (2010). We consider a sequence of systems where the arrival rate and the number of servers get large but the service time distribution and service level requirement remain fixed. We then analyze the steady state of this system and characterize the abandonment probability. A critical part of our analysis is proving that the fluid limits of the CSC system under the proposed policies achieve the desired steady state. We prove this result using carefully constructed Lyapunov functions.

We illustrate via numerical examples that the proposed routing algorithms and staffing solutions perform remarkably well in systems with sizes ranging from small to large. Even in systems with an optimal number of agents fewer than 10, our staffing LP misestimates the optimal solution by at most 1 agent. In general, we show that the solutions found by the staffing LP are very close to the best solutions found by a simulation search procedure, with a difference less than two agents (or less than 4%) across systems with different sizes. It is also interesting to note that the optimality gap between these two solutions does not seem to grow with system size, reminiscent of the results in Bassamboo et al. (2010) and Bassamboo and Randhawa (2010). In Bassamboo et al. (2010) and Bassamboo and Randhawa (2010), the fluid limit analysis has been shown to provide very accurate staffing solutions for systems where agents can only serve a single customer at a time when the arrival rate is random.

This paper is a part of authors' continuing research effort on analyzing CSC systems. In Luo and Zhang (2013), CSC systems without abandonment are analyzed under the lightest-load-first policy. They consider a cost function that depends on what level agents are at and the number of customers in queue. The decision variables are the staffing level and the maximum number of customers an agent can serve. In this paper we extend their analysis in two important directions. First we consider systems with impatient customers and second we identify asymptotically optimal routing policies. We are also able to focus on steady state quantities. By considering more general policies, we are able not only to identify efficient and inefficient levels but also to show when the lightest-load-first is asymptotically optimal. In a follow-up paper (Tezcan and Zhang 2013), we consider CSC systems with general service and abandonment times. We extend our analysis to determine inefficient and efficient levels in that more general setting and identify the maximum number of customers a server should serve when customers are impatient by extensively relying on the results in this paper. Then we offer approximations for the system performance using fluid limits.

The rest of this paper is organized as follows. In §2, we present the CSC queueing model and we formulate the routing and the staffing problems and the asymptotic regime we consider. We present the routing LP in §3 along with its optimal solution. In §4, we propose routing policies first for the case when the basic levels are known and then when they are not known. We also prove that they are asymptotically optimal in minimizing the probability of abandonment from the system for a given staffing level. We turn our attention to the staffing problem in §5. We present the results of our numerical experiments in §6 and conclude in §7. The proofs of all the results that appear in the paper are presented in appendix that is placed in the electronic companion (available as supplemental material at http://dx.doi.org/10.1287/opre.2014.1284).

2. Customer Service Chat Model

This section consists of three parts. First we explain and motivate the analytical model we use to analyze the CSC systems in §2.1. Then we introduce the details of the routing and staffing problems we consider in §2.2. Finally, we introduce the asymptotic regime we use in our analysis in §2.3.

2.1. Model

Chat service systems: In a CSC system, a customer interacts with an agent by sending and receiving written messages on an IM application in (almost) real time. An agent, on the other hand, usually handles multiple customers simultaneously and sends replies to customers' messages in a certain order. The actual dynamics of this process are quite complex. Service of each customer consists of a (random) number of back and forth sessions with the agent. Each such session consists of the customer typing a message and then the agent typing a reply. Also each customer takes a random amount of time to type a message and the length of the message may depend on other factors, such as how long the chat service has been going on, how long the previous messages were, etc. How an agent prioritizes customer messages is also not standard. It may depend on the content of a message, the agent's preference, or the customer's assigned priority, among other factors. Also, customers leave when their service is completed and new customers join the chat system, changing each agent's work load intermittently. As a

result, modeling every detail of a chat service session is quite complex and it is unlikely to lead to analytical insights. Instead we use a processor sharing model, as we explain next. Processor sharing models have been used extensively to establish the performance of round-robin type polling policies (see, for example, Kleinrock 1976).

Processor sharing model: In a processor sharing model, the customers being served by an agent share the agent's service capacity simultaneously. Therefore, the agent's capacity is assumed to be infinitely divisible. Although this is not exactly the case in service chat systems, because of the complexity of the actual system and the similarity of these systems to a processor sharing model (since agents serve customers in a manner similar to round robin), a processor sharing model seems appropriate for customer chat service models. We explain the details of the processor sharing model we use next.

We assume that each agent can serve up to I customers at the same time. If all the agents are serving I customers at the time a customer arrives, the arriving customer joins the queue. An agent serving fewer than I customers is said to be available. When a customer arrives to the system, that customer is assigned to one of the available agents (as explained below) if there are any. Customers waiting in the queue are served according to the first-come-first-serve policy. We assume that agents work in a nonidling fashion: if they finish serving a customer and the queue is nonempty at that instant, they start serving the next customer in queue. Also, once a customer is assigned to an agent, that customer receives service only from that agent until the customer leaves the system (because of service completion or abandonment). Similarly, an agent serves only those customers that are assigned to him, and preemptions or hand-offs are not allowed. Let μ_i denote the rate a customer receives service when an agent at level i is serving that customer. The total rate an agent at level i is providing service, denoted by d_i, is therefore equal to $i\mu_i$.

Each arriving customer requires a random amount of service. When the total amount of service a customer receives reaches the customer's service requirement, that customer leaves the system. We assume that each customer's service amount is exponentially distributed with mean 1, without loss of generality since the time unit can be rescaled. Customers are assumed to arrive to the system according to a renewal process with rate λ, which is assumed to be time independent.

We consider CSC systems where customers have limited patience. To model this behavior we assume that associated with customer k there is a pair of independent exponential random variables (ν_k^q, ν_k). Random variable ν_k^q denotes the kth customer's patience time while waiting in queue; that is, if the kth customer's waiting time in queue exceeds ν_k^q, that customer abandons the system immediately without receiving service. Random variable ν_k denotes the kth customer's patience time while being served; that is, if the time in service for the kth customer exceeds ν_k, the customer abandons the system immediately without finishing service. (We again emphasize that the time in service for a customer depends on what level the agent serving that customer is at.) We assume that $\{(\nu_k^q, \nu_k)\}$ is an i.i.d. sequence. We also assume that the sequences of patience, service, and interarrival times are mutually independent. Also ν_i^q and ν_i are assumed to be exponential with rates γ and ν, respectively.

A few comments are in place here. The independence of ν_k^q and ν_k is assumed for analytical tractability, not based on empirical evidence. We relax this assumption in Tezcan and Zhang (2013) and build approximations when they are dependent, but we do not consider this more general case in this paper. In addition, there are other details of the abandonment behavior of customers that our model may not be able capture. For example, customers in service may be more prone to abandon the system while waiting for their agent to respond rather than while it is their turn to type a message. However, as explained previously, modeling chat sessions in such detail does not seem to lead to insightful results. Hence, in a manner similar to how customer patience while waiting for service in call center applications is modeled, see, for example, Gans et al. (2003), we model customer's impatience using the random variables $\{\nu_k\}$.

Next we describe our assumptions on system parameters. First we define

$$\hat{d}_i = i(\mu_i + \nu), \tag{1}$$

the total rate customers leave an agent that is at level i and

$$P_i^{Ab} = \nu/(\mu_i + \nu) \tag{2}$$

denotes the probability that a customer abandons the system if the customer is served at rate μ_i. When an additional customer is assigned to an agent, it's likely that the agent will not be able to spend as much time with each customer as before since his effort must be divided among more customers now. Therefore, it is expected that μ_i is decreasing in i. Hence we assume that

$$\mu_i > \mu_{i+1}, \quad \text{for } i = 1, 2, \ldots, I-1, \tag{3}$$

(recall that I is the maximum number of customers an agent can serve). However, we expect that d_i should be increasing in i until a certain level. This is because when there are more customers assigned to an agent, the agent will have to wait less for a customer to send a new message. We observed in practice that the average time an agent spends to answer each customer message increases as that agent handles more customers. This is because the agent has to handle more tasks when there are more customers. After a certain number of customers, it is not possible for an agent to handle all customers simultaneously in an effective way. Therefore, we assume that each agent can handle at most I customers.

Inefficient levels: Next we present a classification of levels into two different categories that plays a fundamental role in our analysis. A level i is said to be *inefficient* if

$$\hat{d}_i < \hat{d}_{i'} \quad \text{for some } 1 \leqslant i' < i \tag{4}$$

or if there exist $1 \leqslant i_1 < i \leqslant i_2 \leqslant I$ such that

$$\hat{d}_i \leqslant \frac{i_1 - i}{i_2 - i_1}\hat{d}_{i_2} + \frac{i_2 - i}{i_2 - i_1}\hat{d}_{i_1}. \tag{5}$$

Other levels are referred to as efficient levels. With a little algebra, it can be checked that (5) is equivalent to

$$d_i \leqslant \frac{i_1 - i}{i_2 - i_1}d_{i_2} + \frac{i_2 - i}{i_2 - i_1}d_{i_1}.$$

Loosely speaking, condition (5) implies that instead of having an agent spend one unit of time at an inefficient level i, we can have the agent work a certain fraction of that time at level i_1 and the remaining fraction at level i_2 to achieve a higher throughput without increasing the abandonment rate.

We let $\mathcal{F} = \{i_1, i_2, \ldots, i_J\}$ denote the indices of the efficient levels and $\mathcal{F}^c$ the inefficient levels. We assume that $I \in \mathcal{F}$ and that

$$(1 - P_I^{Ab})\hat{d}_I = d_I \geqslant (1 - P_i^{Ab})\hat{d}_i = d_i, \tag{6}$$

for all $i \leqslant I$. It can be shown that if (6) does not hold, it is optimal not to have agents at level I; a detailed analysis is presented in Tezcan and Zhang (2013). Also, by definition, $1 \in \mathcal{F}$.

Let $\mathcal{N}$ denote the set of the indices of the levels for which the total departure rate is less than that of a lower level. That is,

$$\mathcal{N} = \{i\colon \hat{d}_{i'} > \hat{d}_i, \text{ for some } i' < i\}.$$

Recall that by our assumption (6), $I \notin \mathcal{N}$. We also define

$$\mathcal{N}' = \{i\colon i + 1 \in \mathcal{N}\}.$$

With a slight abuse of terminology, we say that level i is in set $\mathcal{N}$ if $i \in \mathcal{N}$ from here on. We follow the same convention for other sets of indices as well.

2.2. Routing and Staffing Problems

An arriving customer may find multiple available agents at the time of arrival. We refer to the manner in which customers are routed to available agents as the routing policy. One of our objectives in this paper is to identify policies that minimize the probability of abandonment in steady state and to establish approximations for the performance of such policies. Given a routing policy π, let $Ab_\pi^{\lambda, N}(t)$ denote the total number of abandonments from the system (queue and service) by time t when the arrival rate is λ and the number of agents is N. Given the arrival rate λ and the staffing level N, we define $P^{Ab,\pi}(\lambda, N)$ by

$$P^{Ab,\pi}(\lambda, N) = \lim_{T\to\infty} E\left[\frac{Ab_\pi^{\lambda,N}(T)}{T\lambda}\right]. \tag{7}$$

The limit may not exist in general (or may depend on the initial state), but we will show below that it exists and it is unique for nonidling policies. The quantity $P^{Ab,\pi}(\lambda, N)$ can be viewed as the probability of abandonment of a customer who arrives at the chat system in steady state. For the rest of the paper we restrict our attention to nonidling policies that are admissible; specifically, we only consider policies that are Markovian.

Another related operational objective is to determine how many agents to staff the system with to achieve a certain service level. The service level is measured in terms of the abandonment probability from the system in the long run. Formally, the staffing problem can be formulated as follows: given λ and $p^{Ab} > 0$,

$$\begin{aligned} &\min_{\pi, N} \quad N \\ &\text{s.t.} \quad P^{Ab,\pi}(\lambda, N) \leqslant p^{Ab}. \end{aligned} \tag{8}$$

In this formulation the decision variables are the routing policy π and the staffing level N. The staffing problem is closely connected to the routing decisions since the performance of a system depends on how these decisions are made. The main focus of this paper is on the routing problem, but we return our attention to the staffing problem in §5 once we present our solution for the routing problem.

2.3. Asymptotic Framework

Because of the complexity of the CSC systems, it is not possible to carry out an exact analysis in general. Even for relatively less complex call center systems, when available, exact solutions do not provide much insight as to how to manage these systems. On the other hand, the asymptotic analysis has been successfully used in complex systems to study scheduling and staffing problems (see Gans et al. 2003 and Aksin et al. 2007). The asymptotic analysis is also useful in finding solutions that are easy to implement. In service systems, the many-server asymptotic analysis is especially applicable since in most systems there is a pool of agents providing similar services to customers (see Gans et al. 2003). Hence, we use a many-server asymptotic analysis to provide solutions for the routing problem when the system size is large. To identify asymptotically optimal routing policies, we consider a sequence of systems indexed by the arrival rate λ and assume that

$$N^\lambda = N\lambda + o(\lambda). \tag{9}$$

In applications, the size of the system is unlikely to affect the characteristics of arriving customers and the service rates. Therefore, we assume that the service and patience time distributions as well as the service rates at each level are not dependent on λ. A slightly different asymptotic regime is introduced in §5 for the staffing problem.

3. Static Planning Problem

Static planning problems play a fundamental role in the analysis of queueing systems; see Harrison (2000) and Williams (1998) among others. They provide insights into what the expected state of a queueing system should be under an effective control policy in the long run. In this section we formulate a static planning problem for the optimal allocation of agents among all levels, referred to as the *routing LP*. The objective of this LP is to minimize the probability of abandonment in a CSC system with a fixed arrival rate and staffing level. Also, the definition of the efficiency/inefficiency concept is based on the routing LP (or its dual). We show below that there should not be any agents at inefficient levels in the optimal solution of the routing LP. We also show at the end of this section that the routing LP provides a lower bound for the asymptotic performance of any routing policy. A closely related staffing LP will be presented in §5.

For fixed λ and N, consider

$$\min_{\{\lambda_i,\, i=1,\ldots,I,I+1\}} \quad P^{Ab} \tag{10}$$

$$\text{s.t.} \quad \lambda_{I+1} + \sum_{i=1}^{I} \lambda_i P_i^{Ab} = \lambda P^{Ab}, \tag{11}$$

$$\sum_{i=1}^{I} \frac{\lambda_i}{\hat{d}_i} \leqslant N, \tag{12}$$

$$\sum_{i=1}^{I+1} \lambda_i \geqslant \lambda, \tag{13}$$

$$\lambda_i \geqslant 0.$$

The objective of the routing LP is to minimize the probability of abandonment. The decision variables λ_i's can be viewed as the average rate the customers are served by agents at level i, for $1 \leqslant 1 \leqslant I$, and λ_{I+1} can be viewed as the rate customers abandon the queue. Constraint (11) is an approximation for the probability of abandonment, based on (2). (In the following sections we show that this approximation is asymptotically valid.) Constraint (12) states that λ_i's must be selected so as not to violate the capacity constraint, based on Little's law (the asymptotic validity of this claim will be established below). Constraint (13) implies that all arriving customers must depart from the system. We use $P^{Ab}(\lambda, N)$ to denote the optimal objective function value.

Now we obtain a closed-form solution for the routing LP. (We will show that the solution of the LP can be used to approximate the steady state of the proposed policies in the next section.) Let i^*_{j+1} denote the index of the lowest indexed efficient level such that

$$\hat{d}_{i^*_{j+1}} \geqslant \lambda/N. \tag{14}$$

Lemma 1. *If $\lambda \leqslant \hat{d}_1$, an optimal solution $\lambda^* = (\lambda^*_i,\ i = 1, \ldots, I, I+1)$ of the routing LP* (10) *is given by*

$$\lambda^*_1 = \lambda, \quad \textit{and} \quad \lambda^*_i = 0, \quad \textit{for } i \neq 1. \tag{15}$$

If $\hat{d}_I N > \lambda > \hat{d}_1$ an optimal solution of (10) *is given by*

$$\begin{aligned} \lambda^*_{i^*_j} &= \frac{\hat{d}_{i^*_j}}{\hat{d}_{i^*_{j+1}} - \hat{d}_{i^*_j}} (N\hat{d}_{i^*_{j+1}} - \lambda), \\ \lambda^*_{i^*_{j+1}} &= \lambda - \lambda_{i^*_j} = \frac{\hat{d}_{i^*_{j+1}}}{\hat{d}_{i^*_{j+1}} - \hat{d}_{i^*_j}} (\lambda - N\hat{d}_{i^*_j}), \end{aligned} \tag{16}$$

and $\lambda^*_i = 0$ for $i \neq i^*_j, i^*_{j+1}$. If $\lambda \geqslant \hat{d}_I N$, an optimal solution of (10) *is given by*

$$\lambda^*_I = \hat{d}_I N, \quad \lambda^*_{I+1} = \lambda - \hat{d}_I N \quad \textit{and} \quad \lambda^*_i = 0, \quad \textit{for } i \neq I, I+1. \tag{17}$$

The proof is presented in Appendix EC1. Because $\hat{d}_{i^*_{j+1}} > \hat{d}_{i^*_j}$ by (4), the solution given by (16) is well defined. When we want to make the dependence on λ and N explicit, we denote the optimal solution described in Lemma 1 by $\lambda^*(\lambda, N) = (\lambda^*_i(\lambda, N);\ i = 1, \ldots, I, I+1)$. In general there may be alternative optimal solutions of the routing LP (10), but from here on we only consider the optimal solution $\lambda^*(\lambda, N)$. We define

$$z^*_i(\lambda, N) = \lambda^*_i(\lambda, N)/\hat{d}_i, \quad i = 1, \ldots, I \tag{18}$$

and $z^*_0(\lambda, N) = N - \sum_{i=1}^{I} z^*_i(\lambda, N)$. Also, we set $z^*(\lambda, N) = (z^*_i(\lambda, N),\ i = 0, 1, \ldots, I)$ and refer to levels (whose indices are) in the set $i^*(\lambda, N) \triangleq \{i\colon z^*_i(\lambda, N) > 0\}$ as *basic levels*.

We highlight the fact that there can be at most two basic levels by Lemma 1. In addition, an inefficient level can never be basic. This motivates our classification of levels into two classes; efficient and inefficient. The intuition behind this result can best be explained by considering the case $\hat{d}_I N > \lambda > \hat{d}_1$. In words this result implies that in the optimal solution, the agents are distributed between two efficient levels (level i^*_j and level i^*_{j+1}) with lowest indices such that the total departure rate can be made equal to the arrival rate. Because the abandonment probability is increasing in the index of efficient levels, the feasible solution with the lowest indexed efficient levels is optimal.

We next show that the optimal solution of the LP provides an asymptotic lower bound for the abandonment probability under any policy.

Theorem 1. *Consider a sequence of chat service systems that satisfies* (9)*. Then for any sequence of admissible policies $\{\pi^\lambda\}$, we have*

$$\liminf_{\lambda \to \infty} P^{Ab,\,\pi^\lambda}(\lambda, N^\lambda) \geqslant P^{Ab}(1, N), \tag{19}$$

where $P^{Ab}(1, N)$ is the optimal value for the routing LP (10)*.*

The proof is placed in Appendix EC3.

4. Routing Policies

In §3 we proposed a static planning problem approach to obtain a solution for the routing problem in order to minimize the probability of abandonment. In CSC systems, arriving customers are assigned to one of the available agents at the time of their arrivals, and it is not clear how one can dynamically manage such a system to have the long-run behavior given by the solution of (10). In this section we propose routing policies to achieve the long-run proportions suggested by the solution of Lemma 1 for *fixed* arrival rate and number of servers. In §4.1 we consider the case when the basic levels are known, and we present some special cases when the proposed control policies can be simplified in §4.2. We treat the case when the basic levels are not known in §4.3.

4.1. When Basic Levels Are Known

In this section we consider the case when the basic levels i_{j_1} and i_{j_2} of the optimal solution $z^*(1, N)$ are known, with the possibility that $i_{j_1} = i_{j_2}$. (We drop * from the notation for simplicity.) Let $\mathcal{U}_{i_j} = \{i_j + 1, \ldots, i_{j+1} - 1\}$ if $i_j + 1 < i_{j+1}$ and $\varnothing$, otherwise; denote the set of the indices of the inefficient levels whose indices are between i_j and i_{j+1} (indices of two efficient levels) if there are any. Consider the following policy; at the time of a customer arrival, let i denote the index of the lowest indexed nonempty level, (*a level is said to be nonempty if there is at least one agent at that level and it is said to be empty otherwise*),

- if $i < i_{j_1}$, route the arriving customer to an agent serving i customers.
- if $i_{j_1} \leqslant i < i_{j_2}$, route the customer to an agent that is at the *highest* indexed nonempty level in $\mathcal{U}_{i_{j_1}}$. If all the levels in $\mathcal{U}_{i_{j_1}}$ are empty or if $\mathcal{U}_{i_{j_1}}$ is empty, route the customer to an agent at level i_{j_1}.
- if $i_{j_2} \leqslant i < I$ route the customer to an agent at the lowest indexed nonempty level $i' \notin \mathcal{N}' \backslash \{I\}$. If there is no such agent, route the customer to an agent in the highest indexed nonempty level in $\mathcal{N}'$.

A customer cannot be routed to an agent at level I. If all the agents are at level I, then the customer joins the queue. We denote this policy by $\pi^{\lambda,*}$ (We drop λ from the notation when λ is obvious from the context.) When there is only one basic level, i_{j_1}, we follow the same policy but there is no need for the second step; we only consider the first case, $i < i_{j_1}$, and the third case, $i \geqslant i_{j_1}$. We emphasize that in order to implement $\pi^{\lambda,*}$, i_{j_1} and i_{j_2} must be specified.

Example. We note that for fixed i_{j_1} and i_{j_2}, the proposed policy is a fixed priority policy. To illustrate the application of this policy, consider a system with $I = 8$. Let $i_{j_1} = 1$ and $i_{j_2} = 3$; hence, $\mathcal{U}_{i_{j_1}} = \{2\}$. Also assume that $\mathcal{N} = \{5, 7\}$ and so $\mathcal{N}' = \{4, 6\}$. Then the priorities of the levels under the proposed policy are given in the following order; $0, 2, 1, 3, 5, 7, 6, 4$.

Before we explain the intuition behind the proposed policy of $\pi^{\lambda,*}$ we focus on its asymptotic performance. Clearly, the set $i^*(\lambda, N^\lambda)$ of basic levels depends on λ; consequently, so does the proposed policy. We now show that the proposed policy is asymptotically optimal and that it is possible to obtain closed-form approximations for the performance of this policy using the optimal solution of the routing LP. We use $z^\lambda = (z_1^\lambda, z_2^\lambda, \ldots, z_I^\lambda)$ to denote the number of agents at each level in steady state under the proposed policy $\pi^{\lambda,*}$ and define $\bar{z}^\lambda = z^\lambda/\lambda$.

Theorem 2. *Consider a sequence of chat service systems that satisfies* (9)*. Then for the sequence of proposed policies* $\{\pi^{\lambda,*}\}$ *we have*

$$\lim_{\lambda\to\infty} P^{Ab,\,\pi^{\lambda,*}}(\lambda, N^\lambda) = P^{Ab}(1, N), \tag{20}$$

in addition

$$\bar{z}^\lambda \;\Rightarrow\; z^*(1, N) \text{ as } \lambda \to \infty,$$

where $\Rightarrow$ *indicates convergence in distribution.* (21)

It follows from this result and Theorem 1 that the sequence of policies $\{\pi^{\lambda,*}\}$ is an asymptotically optimal routing policy; see Appendix EC4 for a proof.

Discussion on the proposed policy: The proposed policy aims to have all agents in two efficient levels i_{j_1} and i_{j_2} as suggested by Lemma 1. By giving priority to levels with indices lower than i_{j_1}, it aims to have all the agents at level i_{j_1} and at levels with indices larger than i_{j_1}. However, if the index of the lowest indexed nonempty level is greater than or equal to i_{j_1} giving priority to the lowest indexed level may force the system to end up in a nonoptimal steady state. By setting the priority of levels in $\mathcal{U}_{i_{j_1}}$ increasing in their index, the policy aims to "empty" the inefficient levels starting from the one with the highest index in $\mathcal{U}_{i_{j_1}}$. (It is easy to come up with examples that if the priorities are decreasing in index, the steady state will be suboptimal.) The reason the priorities of levels in $\mathcal{N}$ and $\mathcal{N}'$ are set that way is similar. But now the issue is that because levels in $\mathcal{N}$ may have a lower total departure rate than those levels with a lower index, it may not be possible to have agents to go to lower levels if the priorities are not altered. By giving lower priorities to levels in $\mathcal{N}'$, the policy manages to empty levels in $\mathcal{N}$ first; then the levels in $\mathcal{N}'$ become empty after a certain time.

One might question the necessity of switching the priorities of levels in the manner described or the advantage of using the proposed policies over the simpler policies, such as the more intuitive lightest-load-first policy. It is not very difficult to come up with examples when not switching the priorities of levels in $\mathcal{U}_{i_{j_1}}$ and/or $\mathcal{N}$ and $\mathcal{N}'$ would result in suboptimal performance. However, in certain situations simpler policies may yield asymptotically optimal performance. We discuss two such cases in the following section. To what extent using a complex policy rather than

a simple policy improves the performance depends on the system parameters. In §6 we illustrate the difference in the performances of $\pi^{\lambda,*}$ and the lightest-load-first policy in a set of numerical experiments. In general, simpler policies fail to avoid having agents in inefficient levels; hence, the improvement in performance by using $\pi^{\lambda,*}$ over such policies depends on how inefficient the levels are that need to be avoided.

4.2. Special Cases

In this section we discuss two special cases when the proposed policy can be simplified significantly. First, if all the levels are efficient, i.e., $i \in \mathcal{F}$ for all $i = 1, \ldots, I$, the solution to the routing problem simplifies significantly. (It can easily be checked that if d_i is concave as a function of i, then all levels are efficient.) If $i \in \mathcal{F}$ for all $i = 1, \ldots, I$, then $i_{j_1} + 1 = i_{j_2}$ (if there are two basic levels), and $\mathcal{N} = \varnothing$. Hence, the policy π^* becomes a static priority policy that gives priority to the lowest indexed level (we denote it by π). Also, there is no need to estimate the basic levels i_{j_1} and i_{j_2} in this case. Actually the lightest-load-first policy is asymptotically optimal if $\mathcal{N}$ and $\mathcal{U}_{i_j}$ are both empty.

More insight can be gained about the lightest-load-first policy by considering another parsimonious policy. Consider the following policy suggested by an anonymous referee; assign the incoming customer to an agent to maximize the total service rate in the system, i.e., assign the customer to an agent at level i that maximizes $d_{i+1} - d_i$. When all the levels are efficient and level "0" has priority over all the other levels, this is the same policy as the lightest-load-first. To see this, it is enough to show that

$$d_{i+1} - d_i \geqslant d_i - d_{i-1}, \quad i = 1, \ldots, I-1 \tag{22}$$

if all levels are efficient. If (22) does not hold for some i then

$$2d_i \leqslant d_{i+1} + d_{i-1}.$$

Hence, i cannot be efficient by (5). Therefore, the lightest-load-first policy can be viewed as a greedy policy that tries to maximize the instantaneous total service rate at all times (when all the levels are efficient). When inefficient levels are present, the lightest-load-first policy will obviously not maximize the total service rate.

The second special case we consider is when

$$\hat{d}_1 < \hat{d}_2 < \cdots < \hat{d}_I. \tag{23}$$

If (23) holds, it is again possible to use a simpler policy (without needing to estimate the basic levels). Note that in this case $\mathcal{N} = \varnothing$. We propose the following policy. At the time of a customer arrival, let i denote the index of the lowest indexed nonempty level and i_j denote the index of the efficient level with the highest index below i or set $i_j = i$ if level i is efficient. Then route the customer to an agent in the *highest* indexed nonempty level in $\mathcal{U}_{i_j}$. If all the levels in $\mathcal{U}_{i_j}$ are empty or if $\mathcal{U}_{i_j}$ is empty, route the customer to an agent at level i_j. If all the agents are at level I, then the customer joins the queue. We denote this policy by π'.

One of the differences between $\pi^{\lambda,*}$ and π' is that in implementing π' one does not need to know the basic levels i_{j_1} and i_{j_2}. Instead, the priorities of all inefficient levels are changed. As an example, again consider a CSC system with $I = 8$ and assume that levels 2, 4, and 5 are inefficient. Then under π' the priorities of the levels are given in the following order; 0, 2, 1, 5, 4, 3, 6, 7. (Recall that under $\pi^{\lambda,*}$ the priorities depend on the basic levels hence on the arrival rate.) The following result establishes the optimality of π' when (23) holds.

THEOREM 3. *Consider a sequence of chat service systems that satisfies* (9) *and* (23)*. Then for the sequence of proposed policies* $\{\pi'\}$ *we have*

$$\lim_{\lambda\to\infty} P^{Ab,\pi'}(\lambda, N^\lambda) = P^{Ab}(1, N); \tag{24}$$

in addition,

$$\bar{z}^{\lambda,\pi'} \Rightarrow z^*(1, N), \quad \text{as } \lambda \to \infty. \tag{25}$$

4.3. Routing Policies: When Basic Levels Are Not Known

In this section we focus on the case when the basic levels are not known. For two special cases we already discussed how the proposed policy can be simplified and basic levels do not have to be estimated in §4.2. Next we treat the general case and we propose a policy that consists of two components: the virtual system that is used to identify basic levels and the actual system where the actual routing is done. Before introducing the virtual systems, we need to introduce the preemptive systems that form the foundation of the virtual systems.

Preemptive systems: Consider a system where preemption in the manner we explain next is allowed. First note that $\mathcal{U}_{i_j} = \{i_j + 1, \ldots, i_{j+1} - 1\}$, if $i_{j+1} > i_j + 1$, and $\mathcal{U}_{i_j}$ is an empty set otherwise. If there is an agent at one of the levels in $\mathcal{U}_{i_j}$ (assuming it is not empty), refer to it as inefficient-level agent (we argue below that there can be at most one agent at levels $\mathcal{U}_{i_j}$ under the preemptive regime). Assume that an agent at level i_{j+1} finishes serving a customer at time t; call this agent the service-completion agent. At the instant the agent at level i_{j+1} finishes service, one of the customers the inefficient-level agent is serving is handed off (whose service is said to be preempted) to the service-completion agent at this instant. Therefore, the level the service-completion agent is at does not change, and the inefficient-level agent (who hands off the customer) goes one level below. If there are no agents at one of the levels in $\mathcal{U}_{i_j}$ or if $\mathcal{U}_{i_j}$ is empty, no action is taken.

Except for the explained preemptions, the policy we use in this setting is similar to the one we proposed in the previous

section. Assume that at the time of a customer arrival, level i is the lowest indexed nonempty level. If $i \notin \mathcal{F}$, route the customer to level i. If $i \in \mathcal{F}$, route the customer to one of the agents at the highest indexed level in $\mathcal{U}_i$ if $\mathcal{U}_i$ is not empty and there is an agent at one of those levels. Otherwise, route the customer to level i. We denote this policy by π_P. We note that this policy does not use any information about the basic levels.

Now we argue that there can be at most one agent at one of the levels in $\mathcal{U}_{i_j}$, for any $i_j \in \mathcal{F}$, if customers are rearranged properly at time zero. Assume that at time zero there is more than one agent at levels in $\mathcal{U}_{i_j}$ ($\neq \varnothing$). The customers that are being served by these agents can be redistributed easily among these agents to make sure there is only one such agent left at one of the levels in $\mathcal{U}_{i_j}$. For example, an agent at level $i_j + 1$ can hand off one customer to another agent at one of the inefficient levels, if there is such an agent, and goes to level i_j, and the other agent's level is increased by one. Once the customers are rearranged at time zero, there can be at most one agent at a level in $\mathcal{U}_{i_j}$ under our preemptive policy.

We next show that π_P identifies the correct allocations of agents among all the levels in the long run in the preemptive systems. First, we need to make certain assumptions about the initial states of the systems to obtain meaningful limits. Let $Z_i^\lambda(t)$ denote the number of agents at level i and $Q^\lambda(t)$ denote the number of customers waiting in queue at time t. We set $Z^\lambda(t) = (Z_0^\lambda(t), \ldots, Z_I^\lambda(t))$, $\bar{Z}^\lambda(t) = Z^\lambda(t)/\lambda$, and $\bar{Q}^\lambda(t) = Q^\lambda(t)/\lambda$. We assume that

$$\lim_{\lambda\to\infty} \bar{Z}^\lambda(0) = \bar{Z}(0), \quad \text{a.s.}, \tag{26}$$

for some vector $\bar{Z}(0) = (\bar{Z}_0(0), \ldots, \bar{Z}_I(0))$ and

$$\lim_{\lambda\to\infty} \bar{Q}^\lambda(0) = 0, \quad \text{a.s.} \tag{27}$$

Proposition 1. *Consider a sequence of chat service systems that satisfies* (9), (26), *and* (27). *Under* π_P

$$\lim_{T\to\infty} \lim_{\lambda\to\infty} \bar{Z}^\lambda(T) \to z^*(1, N) \quad \textit{a.s.} \tag{28}$$

See Appendix EC5 for a proof.

Nonpreemptive systems with unknown basic levels: Although the policy π_P we proposed for the preemptive systems identifies the basic levels automatically, in CSC systems preemption and handing off are rarely used. This is mainly because handing off a customer to another agent may lead to inefficiencies since the agent who receives the customer does not know the history of the chat session with the customer. Hence that agent is likely to spend extra time to go through the conversation of the customer with the previous agent or ask the same questions again, which in addition is likely to frustrate the customer. Therefore, we propose asymptotically optimal policies that do not require preemption (or handing off customers).

As mentioned above, we use a virtual system to identify basic levels. In this virtual system, the number of (virtual) agents is the same as the actual system and so is the arrival process, but service times are different. At each customer arrival we generate a random service time for the arriving customer in the virtual system using the same service time distribution in the actual system. The virtual system runs in the same manner as the preemptive system we described in the previous section. The customers are rearranged in the virtual system at time zero as described above, and an arriving customer is routed to one of the virtual agents following π_P.

Next we describe the routing policy for the actual system that uses the virtual system. We basically use the policy π^* proposed in §4.1, but we estimate i_{j_1} and i_{j_2} using the virtual system as follows. Fix $\epsilon > 0$. If there are at least two efficient levels in the virtual system with at least $N\epsilon$ agents at time t, we use the efficient level with the lowest index, denoted by $\hat{\imath}_{j_1}(t)$ as an estimator for i_{j_1} and we use $\hat{\imath}_{j_1+1}(t)$, as an estimator for i_{j_2}. If there is only one such level, denoted by $\hat{\imath}_{j_1}(t)$, we use π^* with basic levels $\hat{\imath}_{j_1}(t)$ and $j_1(t)$, where $j_1(t)$ is the smallest indexed level greater than $\hat{\imath}_{j_1}(t)$ that is not in $\mathcal{N}$, if $\hat{\imath}_{j_1}(t) \neq I$. If $\hat{\imath}_{j_1}(t) = I$, we use π^* with a single level. Aside from this estimation procedure, π^* is used as described in §4.1. That is, if a customer arrives at time t, we use $\hat{\imath}_{j_1}(t)$ and $\hat{\imath}_{j_1+1}(t)$ (or $j_1(t)$) instead of i_{j_1} and i_{j_2}, respectively, in §4.1. The virtual system is run simultaneously with the actual system. We denote this policy by $\hat{\pi}^*$ or by $\hat{\pi}^*(\epsilon)$ when we would like to make the dependence on ϵ explicit.

The policy $\hat{\pi}^*$ is not an asymptotically optimal routing policy in the sense similar to (20) because it turns out that it does not satisfy (28) in general. This is because in our estimation procedure, a level is deemed basic if there are more than $N\epsilon$ agents in that level in the virtual system. On the other hand, it is possible that in the optimal solution of (10), a level is basic with less than $N\epsilon$ agents. However, since $\epsilon > 0$ is arbitrary, the performance of $\hat{\pi}^*(\epsilon)$ can be made arbitrarily close to the performance of π^*. We state this result next.

Theorem 4. *Consider a sequence of chat service systems that satisfies* (9). *For* $\delta > 0$, *there exists* $\epsilon > 0$ *such that the sequence of chat service systems under the policy* $\hat{\pi}^*(\epsilon)$ *satisfies*

$$\limsup_{\lambda\to\infty} P^{Ab,\,\hat{\pi}^*}(\lambda, N^\lambda) \leqslant P^{Ab}(1, N) + \delta, \tag{29}$$

for any $\lambda \geqslant 0$ (*i.e., the choice of* $\epsilon > 0$ *does not depend on* λ). *Also, if* $z_i^*(\lambda, N) > \epsilon$ *for* $i = i_j^*$ *and* i_{j+1}^*,

$$\limsup_{\lambda\to\infty} P^{Ab,\,\hat{\pi}^*}(\lambda, N^\lambda) = P^{Ab}(1, N). \tag{30}$$

See Appendix EC6 for a proof. We expect that policy π^* (which requires the knowledge of the basic levels) performs slightly better than $\hat{\pi}^*$ since it does not have to estimate

the basic levels. However, the difference should be small in the long run since $\hat{\pi}^*$ eventually estimates the basic levels accurately (except those that have very few agents). We compare the performances of these policies in several numerical experiments in §6.

5. Staffing Problem

In this section we turn our attention to the staffing problem (8). Recall that our objective is to identify the minimum staffing level that satisfies a service level constraint in terms of the abandonment probability for a fixed arrival rate. We already identified policies to effectively route customers to available agents; hence, we restrict our attention to these policies. Our solution procedure therefore is based on an LP, referred to as the staffing LP, which is similar to the routing LP.

Staffing LP: Given the arrival rate λ and the service level constraint p^{Ab}, consider the following staffing LP.

$$\min_{\{N,\,\lambda_i:\,i=1,\ldots,I+1\}} \quad N \tag{31}$$

$$\text{s.t.} \quad \sum_{i=1}^{I} P_i^{Ab}\lambda_i + \lambda_{I+1} \leqslant p^{Ab}\lambda, \tag{32}$$

$$\sum_{i=1}^{I} \frac{\lambda_i}{\hat{d}_i} \leqslant N, \tag{33}$$

$$\sum_{i=1}^{I} \lambda_i + \lambda_{I+1} \geqslant \lambda, \tag{34}$$

$$\lambda_i \geqslant 0, \quad i = 1, 2, \ldots, I+1$$

The objective of the staffing LP is to minimize the number of agents. The decision variables (λ_i's) have the same interpretation as in the routing LP with λ_{I+1} denoting the rate customers abandon the queue. The left hand side of the constraint (32) can be understood as the total abandonment rate for the system. The constraints (33) and (34) are identical to (12) and (13), respectively. The staffing and the routing LPs are closely related as described in the following result. We denote the optimal solution of the staffing LP (31) by $N(\lambda, p_{Ab})$. If $p^{Ab} < P_1^{Ab}$, the staffing LP is infeasible, so we assume that $p^{Ab} \geqslant P_1^{Ab}$. Let i^*_{j+1} denote the index of the lowest indexed efficient level with

$$P^{Ab}_{i^*_{j+1}} \geqslant p^{Ab}. \tag{35}$$

Lemma 2. *If $P_1^{Ab} \leqslant p^{Ab} < P_I^{Ab}$, an optimal solution (λ^*, N^*), with $\lambda^* = (\lambda_i^*:\ i = 1, \ldots, I+1)$, of (31) is given by $\lambda_i^* = 0$, if $i \neq i^*_j, i^*_{j+1}$,*

$$\lambda^*_{i^*_j} = \frac{P^{Ab}_{i^*_{j+1}} - p^{Ab}}{P^{Ab}_{i^*_{j+1}} - P^{Ab}_{i^*_j}}\lambda,$$
$$\lambda^*_{i^*_{j+1}} = \lambda - \lambda^*_{i^*_j} = \frac{p^{Ab} - P^{Ab}_{i^*_j}}{P^{Ab}_{i^*_{j+1}} - P^{Ab}_{i^*_j}}\lambda, \tag{36}$$

and

$$N^* = \frac{\lambda^*_{i^*_j}}{\hat{d}_{i^*_j}} + \frac{\lambda^*_{i^*_{j+1}}}{\hat{d}_{i^*_{j+1}}}.$$

If $p^{Ab} \geqslant P_I^{Ab}$, an optimal solution of (31) is given by $\lambda_i^* = 0$, $i = 1, 2, \ldots I-1$,

$$\lambda_I^* = \frac{\lambda(1-p^{Ab})}{1-P_I^{Ab}}, \quad \lambda^*_{I+1} = \lambda - \lambda_I^*, \quad \text{and} \quad N^* = \lambda_I^*/\hat{d}_I. \tag{37}$$

The proof is very similar to that of Lemma 1; hence, it is omitted. We note again that $P^{Ab}_{i^*_{j+1}} > P^{Ab}_{i^*_j}$ by Lemma EC1 hence (36) is well defined.

Asymptotic optimality: Now we show that the optimal solution of the staffing LP is asymptotically optimal when the arrival rate is large. For this purpose we use a slightly different asymptotic regime from the one we described in §2.3;we consider a sequence of systems where the arrival rate becomes large but the service level $p^{Ab} > 0$ and other system parameters are held constant. Consider a sequence of systems indexed by the sequence arrival rates $\{\lambda^n\}$ such that

$$\frac{\lambda^n}{n} \to \lambda \quad \text{as } n \to \infty. \tag{38}$$

A sequence of staffing levels $\{N^n\}$ is said to be asymptotically feasible if there exists a sequence of policies $\{\pi^n\}$ such that

$$\limsup_{n\to\infty} P^{Ab,\,\pi^n}(\lambda^n, N^n) \leqslant p^{Ab}. \tag{39}$$

Also, a sequence of asymptotically feasible staffing levels $\{N^{*,n}\}$ is asymptotically optimal if for any sequence of asymptotically feasible staffing levels $\{N^n\}$,

$$\limsup_{n\to\infty} \frac{N^{*,n}}{N^n} \leqslant 1. \tag{40}$$

Given the arrival rate λ^n and service level constraint p^{Ab}, the staffing LP not only determines the optimal staffing level $N^*(\lambda^n, p^{Ab})$ but also the basic levels, those levels with $\lambda_i^* > 0$. It is easy to show that these basic levels are those given by $i^*(\lambda^n, N^*(\lambda^n, p^{Ab}))$ (recall the definition given in §3). Hence the proposed policy can be used in this setting as well with these basic levels. (In the implementation of our policy we assume that the basic levels are known. We consider the case when they are not below.) We denote this policy in the nth system by $\pi^{n,*}$ for notational simplicity. We next show the staffing LP (31) gives an asymptotically optimal staffing level when used with the proposed dynamic priority policy $\pi^{n,*}$.

Theorem 5. *Consider a sequence of chat service systems that satisfies (38). The sequence of staffing levels $\{N^*(\lambda^n, p^{Ab})\}$ is asymptotically feasible when used with the sequence of proposed policies $\{\pi^{n,*}\}$. Moreover, $\{N^*(\lambda^n, p^{Ab})\}$ is asymptotically optimal.*

See Appendix EC7 for a proof.

Unknown arrival rate: Finally, we consider the case when the basic levels are not known. This case is especially important when the exact arrival rate is not known and staffing decisions must be made based on the distribution of the arrival rate. In this case, there is a small inefficiency (relative to the case when the basic levels are known) in the system introduced by the estimation procedure. However, the magnitude of the inefficiency can be made arbitrarily small, as we explain next.

Based on Theorem 4, the proposed policy $\hat{\pi}^*(\epsilon)$ can be off from the optimal solution by ϵ. This happens only when one of the basic levels should have fewer than ϵN agents in steady state. Otherwise, $\hat{\pi}^*(\epsilon)$ attains the same steady state as $\pi^{n,*}$. Therefore, it is possible to inflate the staffing level $N^*(\lambda^n, p^{Ab})$ by a small amount δ to obtain asymptotically feasible staffing solution. Next we formalize these results.

THEOREM 6. *Consider a sequence of chat service systems that satisfies* (38)*. Given $\delta > 0$, there exists $\epsilon > 0$ such that the sequence of staffing levels $\{(1+\delta)N^*(\lambda^n, p^{Ab})\}$ is asymptotically feasible when used with the routing policy $\hat{\pi}^*(\epsilon)$.*

See Appendix EC7 for a proof.

6. Numerical Experiments

In this section we present the results of numerical experiments to illustrate the effectiveness of the routing policies we proposed and our solution procedure for the staffing problem. Throughout the experiments in this section we consider a CSC system with six levels. We use the following service rates for each level;

$$\mu = \{2.8, 2, 1.6, 1.5, 1.15, 1.15\};$$

therefore,

$$d = \{2.8, 4, 4.8, 6, 5.75, 6.9\}.$$

It is easily checked that only levels 3 and 5 are inefficient ($\mathcal{F} = \{1, 2, 4, 6\}$, $\mathcal{F}^c = \{3, 5\}$) and $\mathcal{N} = \{5\}$. We also set the abandonment rates $\nu = \gamma = 0.2$. We first focus on the routing policies and assume that the staffing level is fixed. We run experiments in six different settings with various arrival rates and staffing levels. In Table 1, we display the associated data with each setting along with the optimal solution $z^* = (z_1^*, \ldots, z_6^*)$ to the routing LP (10); see (18). (The sizes of the systems in the third and sixth experiments are not very realistic and we use them to verify the asymptotic accuracy of our results). We present the optimal abandonment probability estimated by LP (10) under the column P_{ab} for each setting as well. We present the simulation results under three policies π^* (we drop λ from the notation for simplicity), $\hat{\pi}^*$, and the policy that gives priority to the lowest indexed nonempty level, denoted by π. (Recall that the policy π is asymptotically optimal if $\mathcal{F}^c = \varnothing$.) Observe that under the policy π^*, the priorities are given by $(0, 1, 3, 2, 5, 4)$ in the first three settings in Table 1 and by $(0, 1, 2, 3, 5, 4)$ in Settings 4–6. For the policy $\hat{\pi}^*$, we set $\epsilon = 0.1$. In each simulation experiment, we run the simulation to allow 1.5 million arrivals and we use the first 20% of the simulation time as the warmup period. The results of the simulations are presented in Table 2 for Settings 1–3 and in Table 3 for Settings 4–6. In these tables, under column Z_i we present the average number of agents at level i and under column P_{ab} we present the percentage of customers that abandoned the system. The confidence intervals in all the experiments are less than 0.5% of the mean values; therefore, we do not include them in our tables.

Table 1. Data for experiments and optimal solutions.

Setting	λ	N	z_1^*	z_2^*	z_3^*	z_4^*	z_5^*	z_6^*	P_{ab} (%)
1	140	25	0	12.5	0	12.5	0	0	10.4
2	280	50	0	25	0	25	0	0	10.4
3	1,400	250	0	125	0	125	0	0	10.4
4	180	25	0	0	0	17.30	0	7.70	12.8
5	320	50	0	0	0	34.60	0	15.40	12.8
6	1,800	250	0	0	0	173	0	77	12.8

As expected from our asymptotic results, the performances of the systems under the proposed policies π^* and $\hat{\pi}^*$ converge to the values estimated by the optimal solution of the routing LP (10), presented in Table 1, as the system size gets large. In addition, the proposed policies perform remarkably well even when there are only 25 agents. Also, the policy $\hat{\pi}^*$, which uses the virtual system to estimate basic levels, performs slightly worse than the policy π^* in all cases, but the difference is very small. When there are 25 agents, the average difference between the performance of π^* and the optimal solution of LP (10) is less than 4%. This difference goes below 2% when there are 250 agents. This shows that the static planning LP (10) is quite accurate in estimating the system performance.

The differences between the performance of π^* and the optimal solution of the routing LP (10) are because the occupancy in some levels are 0 in the optimal solution, but this cannot be achieved in reality without preemption. For example, in Setting 2, there are 25 expected agents at level 4 and zero agents at level 3 in the optimal solution of (10). However, when an agent completes service at level 4, that agent will have to wait for an arrival to go back to level 4 or complete another service to go to level 2, the other basic level. In either case, the agent will have to wait for a certain amount of time because the arrivals (or service completions) are not instantaneous and there may be other agents awaiting arrivals in nonbasic levels 1 or 3. As the system size gets larger, the wait will be shorter. This is also observed in our numerical experiments. In Setting 1, where the arrival rate is 140, the average number of agents at level 3 is around 10% of all agents, and when the arrival rate is increased to 1,400 in Setting 3, it decreases to 3.4%, under π^*.

We present the results under π in order to compare our policies to another policy that does not require the solution of the LP (10). In terms of the abandonment probability, it

Table 2. Results of simulation experiments: Settings 1–3.

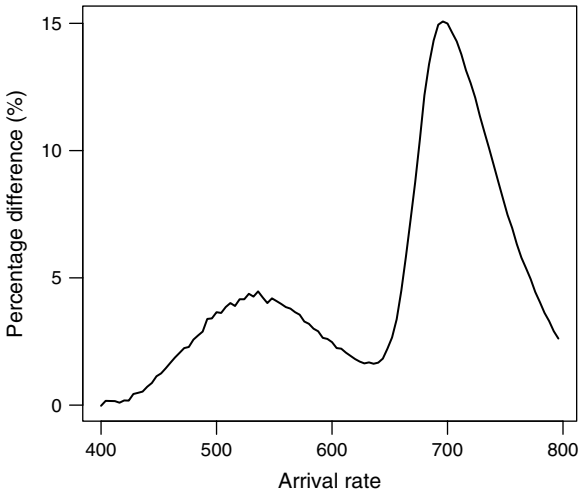
Setting	Policy	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	P_{ab} (%)
1	π^*	0.012	0.554	10.096	2.507	11.668	0.16	0.002	10.80
1	$\hat{\pi}^*$	0.012	0.529	9.693	3.278	11.328	0.154	0.006	10.82
1	π	0.004	0.179	3.581	14.729	6.297	0.207	0.002	11.08
2	π^*	0.006	0.573	21.859	3.707	23.831	0.024	0	10.75
2	$\hat{\pi}^*$	0.006	0.571	21.767	3.874	23.755	0.026	0	10.76
2	π	0.001	0.101	4.758	34.323	10.78	0.038	0	11.12
3	π^*	0.001	0.607	118.697	8.614	122.08	0	0	10.72
3	$\hat{\pi}^*$	0.001	0.604	118.68	8.622	122.092	0	0	10.73
3	π	0	0.022	6.466	202.582	40.93	0	0	11.18

Table 3. Results of simulation experiments: Settings 4–6.

Setting	Policy	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	P_{ab} (%)
4	π^*	0	0.002	0.061	1.529	11.683	2.11	9.615	13.30
4	$\hat{\pi}^*$	0	0.003	0.12	1.171	10.316	3.978	9.412	13.49
4	π	0	0	0.02	0.49	3.939	11.796	8.755	14.34
5	π^*	0	0	0.03	1.751	27.251	3.29	17.678	13.14
5	$\hat{\pi}^*$	0	0.001	0.054	1.462	25.12	5.829	17.534	13.27
5	π	0	0	0.003	0.206	4.6	28.258	16.932	14.53
6	π^*	0	0	0.005	1.745	159.571	8.278	80.401	12.94
6	$\hat{\pi}^*$	0	0	0.005	1.681	157.447	10.441	80.426	12.97
6	π	0	0	0	0.009	2.364	163.093	84.534	14.77

can be perform as much as 13% worse than the proposed policies and its average performance in all experiments is around 6% worse than π^*. In order to have a better idea about the differences between these two policies (π^* and π), we next simulate the system with $N = 100$ and arrival rate ranging from 400 to 800. We present the increase in the abandonment probability versus the arrival rate when policy π is used instead of π^* in Figure 1. We note that when the arrival rate is between 440 and 680, the basic levels are 2 and 4, and between 680 and 800, they are 4 and 6. Note that π^* tries to avoid having agents at levels 3 and 5 as much as possible. Also, level 5 is more "inefficient" than level 3, in the sense that using levels 4 and 6 instead of level 5 is more beneficial than using levels 2 and 4 instead of level 3. Hence, the difference between the performances of these two policies is significantly larger for higher values of the arrival rate. In general, the difference between the policy π^* and π depends on how inefficient is a level that is avoided by π^*.

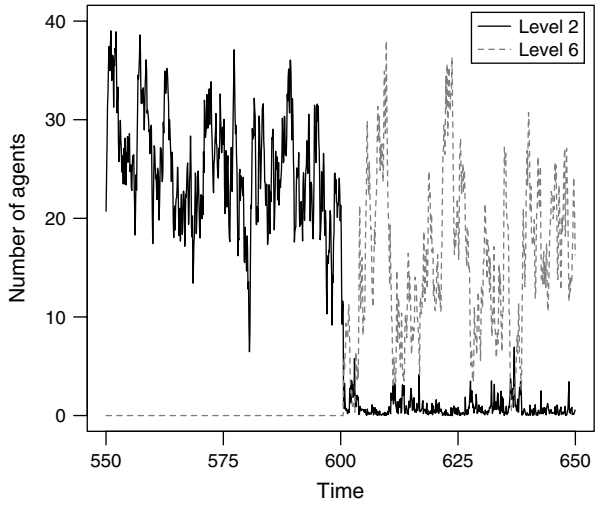
Figure 1. Percentage difference in abandonment probabilities between π^* and π.



We next illustrate how the virtual system adapts to changes in arrival rates. We consider a system in Setting 2 where the arrival rate is 280, and we change the arrival rate to 360 at time 600. Therefore, until time 600, levels 2 and 4 are basic and after time 600, levels 4 and 6 are basic. We simulate the system until time 650 and we plot the average number of agents at levels 2 and 6 in each five-time unit interval in Figure 2. As can be seen in Figure 1, there is a quick change in the number of agents at each level at time 600, and they reach levels close to the "steady state" quantities as anticipated by LP (10).

Remark 1. To test the effect of the number of levels on the performance of our approximations, we also run simulation experiments with $I = 4$ in Settings 1–3. The abandonment probabilities in these experiments were very close to those when $I = 6$; the average difference was less than 1%. We only observed significant differences between the systems with $I = 4$ and $I = 6$ when the average queue length in steady state is nonnegligible and the abandonment rate from queue is large. For example, when we set $\lambda = 160$, $N = 25$ and $\gamma = 10$ (all the other parameters are set equal to their corresponding values in Setting 1), the difference between

Figure 2. Number of agents at levels 2 and 6 in the virtual system.



the abandonment rates in systems with $I = 4$ and $I = 6$ is close to 15%.

6.1. Staffing Levels

We next test the quality of the staffing levels found by the staffing LP in simulation experiments. We consider two service level targets $p^{Ab} = 0.10$ and 0.14 with three different arrival rates presented in Table 4. We use the staffing LP (31) to compute the required staffing levels with these parameters.

To assess the quality of our proposed solution, we also find effective staffing levels using a numerical procedure along with simulations. Specifically, we use a simple search algorithm where we start with a small value for the staffing level and simulate the system under the proposed policy to estimate the performance. We keep on increasing the staffing level until the service level requirement is satisfied and stop at the smallest feasible staffing level. In Table 4, we present the solutions from the numerical search algorithm under column $\bar{N}$ and the solutions of the staffing LP (31) under column N^*. Obviously the staffing LP's solutions are very close to those found by the numerical method: they are off by at most one agent. We emphasize that the proposed staffing solutions are very accurate even in small systems where the optimal staffing level is as low as seven agents.

Table 4. Comparison of staffing solutions.

p^{Ab}	λ	$\hat{N}$	N^*
0.10	50	11	10
	100	21	20
	250	51	50
0.14	50	7	7
	100	14	13
	250	34	33

7. Conclusions and Directions for Further Research

In this paper we considered the staffing and routing in CSC systems. The fact that agents can serve multiple customers simultaneously makes the analysis of these systems more complicated than that of traditional service systems. We first propose a routing LP to identify effective routing policies. Initially we propose a policy for the case when the arrival rate is observable. We also consider the case when the arrival rate is unobservable and we propose a policy based on a virtual system that automatically and seamlessly determines basic levels. We prove that these policies asymptotically minimize the abandonment probability in steady state in large systems. We also present a staffing LP and prove that the staffing solution found by the staffing LP is asymptotically optimal. Our numerical experiments suggest that the proposed solution procedure and the routing policies work remarkably well even with systems that have fewer than 10 agents.

There are several interesting future research directions. First, we only focus on the case when the service requirements of the customers are exponentially distributed. When the service requirements have a general distribution, measure-valued fluid limits (see Whitt 2006) can be used to study these systems. It could also be possible to establish the distribution of time in CSC systems, instead of just the expected value, using such fluid limits. In addition, especially when $\lambda \approx \hat{d}_i N$, for some i, the diffusion limits may be more suitable to determine effective staffing levels. For the overloaded case, the analysis is similar to the (traditional) case when agents only help one customer at a time. Finally, agents usually have different capabilities and customers have different needs in most chat systems. How to route arriving customers in different classes to agents with different skills effectively needs to be studied as well. It may be possible to cross-train agents to provide service in multiple service channels, such as email and chat. In such a setting it is interesting to study the staffing and the routing problems by considering the different service level measures in different channels.

Supplemental Material

Supplemental material to this paper is available at http://dx.doi.org/10.1287/opre.2014.1284.

Acknowledgments

Tolga Tezcan's research is supported in part by National Science Foundation [Grants CMMI-0954126 and CMMI-1130346]. Jiheng Zhang's research is supported in part by Hong Kong Research Grants Council [Grant GRF-622411]. The authors thank the anonymous referees for their comments and suggestions that improved the paper significantly.

References

Aksin Z, Armony M, Mehrotra V (2007) The modern call center: A multi-disciplinary perspective on operations management research. *Production Oper. Management* 16(6):655–688.

Andrews DC, Haworth KN (2002) Online customer service chat: Usability and sociability issues. *J. Internat. Marketing* 2(1):1–20.

Atar R, Giat C, Shimkin N (2010) The $c\mu/\theta$ rule for many-server queues with abandonment. *Oper. Res.* 58(5):1427–1439.

Atar R, Giat C, Shimkin N (2011) On the asymptotic optimality of the $c\mu/\theta$ rule under ergodic cost. *Queueing Systems: Theory Appl.* 67(2):127–144.

Bannan K (2000) Chatting up a sale. *Wall Street Journal* (October 25), http://www.liveassistance.com/n001023a.html.

Bassamboo A, Randhawa RS (2010) On the accuracy of fluid models for capacity sizing in queueing systems with impatient customers. *Oper. Res.* 58(5):1398–1413.

Bassamboo A, Randhawa RS, Zeevi A (2010) Capacity sizing under parameter uncertainty: Safety staffing principles revisited. *Management Sci.* 56(10):1668–1686.

Behzad B, Tezcan T (2012) Robust design and control of call centers with flexible IVR systems. *Manufacturing Service Oper. Management* 14(3):386–401.

Bob B, Terwiesch C (2013) The behavior of doctors under load: An empirical analysis of reputation in online service market places. Technical report, The Wharton School, University of Pennsylvania, Philadelphia.

Broughton K (2001) Our experiment in online, real-time reference. *Comput. Libraries* 21(4):26–31.

Francoeur S (2001) An analytical survey of chat reference services. *Reference Services Rev.* 29(3):189–204.

Gans N, Koole G, Mandelbaum A (2003) Telephone call centers: Tutorial, review and research prospects. *Manufacturing Service Oper. Management* 5(2):79–141.

Gladstones WH, Regan MA, Lee RB (1989) Division of attention: The single-channel hypothesis revisited. *Quart. J. Experiment. Psych.* 41(1):1–17.

Harrison JM (2000) Brownian models of open processing networks: Canonical representation of workload. *Ann. Appl. Probab.* 10(1): 75–103.

Hasija S, Pinker E, Shumsky RA (2010) OM practice—Work expands to fill the time available: Capacity estimation and staffing under Parkinson's law. *Manufacturing Service Oper. Management* 12(1):1–18.

KC DS, Terwiesch C (2009) Impact of workload on service time and patient safety: An econometric analysis of hospital operations. *Management Sci.* 55(9):1486–1498.

Kleinrock L (1976) *Queueing Systems, Volume 2: Computer Applications*. (John Wiley & Sons, New York).

Koyama JY (1998) http://digiref.scenarios.issues. *Reference User Services Quart.* 38(1):51–53.

Luo J, Zhang J (2013) Staffing and control of instant messaging based customer service centers. *Oper. Res.* 61(2):328–343.

Oder N (2001) The shape of e-reference. *Library J.* 126(2):46–60.

Pashler H (1994) Dual-task interference in simple tasks: Data and theory. *Psych. Bull.* 116(2):220–244.

Shae Z, Garg D, Bhose R, Mukherjee R, Guven S (2007) Efficient Internet chat services for help desk agents. *IEEE Internat. Conf. Services Comput.* (IEEE Computer Society, Los Alamitos, CA), 589–596.

Staats BR, Gino F (2012) Specialization and variety in repetitive tasks: Evidence from a Japanese bank. *Management Sci.* 58(6):1141–1159.

Tan T, Netessine S (2014) When does the devil make work? An empirical study of the impact of workload on worker productivity. *Management Sci.* ePub ahead of print April 21, http://dx.doi.org/10.1287/mnsc.2014.1950.

TELUS International (2011) Best practices: Online chat sales. Benchmarking study white paper.

Tezcan T, Zhang J (2013) Chat service systems with general service and abandonment times. Working paper, University of Rochester, Rochester, NY.

Whitt W (2006) Fluid models for multiserver queues with abandonments. *Oper. Res.* 54(1):37–54.

Williams RJ (1998) Diffusion approximations for open multiclass queueing networks: Sufficient conditions involving state space collapse. *Queueing Systems: Theory Appl.* 30(1–2):27–88.

Zhang J, Dai JG, Zwart B (2009) Law of large number limits of limited processor-sharing queues. *Math. Oper. Res.* 34(4):937–970.

Tolga Tezcan is an assistant professor of operations management at the Simon School of Business, University of Rochester. His research interests are in asymptotic analysis and optimal control of large queuing systems with applications in service systems.

Jiheng Zhang is an assistant professor in the Department of Industrial Engineering and Logistics Management at the Hong Kong University of Science and Technology. His research interests are in performance evaluation and optimal control via asymptotic analysis of queueing systems arising from applications in manufacturing and services.