Proceedings of the 2023 Winter Simulation Conference
C. G. Corlu, S. R. Hunter, H. Lam, B. S. Onggo, J. Shortle, and B. Biller, eds.

ACHIEVING STABLE SERVICE-LEVEL TARGETS IN TIME-VARYING QUEUEING SYSTEMS: A SIMULATION-BASED OFFLINE LEARNING STAFFING ALGORITHM

Kurtis Konrad
Yunan Liu

Edward P. Fitts Department of Industrial and Systems Engineering
North Carolina State University
Raleigh, NC 27607, USA

ABSTRACT

In this paper, we develop a new staffing algorithm for achieving stable service-level targets in queues with time-varying arrivals. Specifically, we aim to stabilize the tail probability of delay, which is the probability that the waiting time exceeds a designated target $\tau > 0$. We integrate reinforcement learning into the decision making in queueing models; our new method recursively evolves the staffing decision by alternating between two phases: (i) we generate simulated queueing data by operating the system under the present staffing function (exploitation), and (ii) we utilize the newly generated data to devise improved staffing decision (exploration). We demonstrate the effectiveness of our new method using various numerical examples.

1 INTRODUCTION

In service systems, an important *service-level* (SL) goal is to keep the *fraction of customers who are served within a delay target* τ above a probability target $1-\alpha$, or equivalently, contain the *probability that the waiting time exceeds* τ below α. With $\tau = 20$ seconds and $1-\alpha = 0.8$, this goal reduces to the famous 80/20 rule, a long-standing industrial standard in call center management. Let W_t be the waiting time at time t; such an SL constraint can be formally described using the *tail probability of delay* (TPoD). Namely,

$$\mathbb{P}(W_t > \tau) \leq \alpha, \qquad \text{for} \quad \tau > 0, \quad \alpha \in (0,1). \tag{1}$$

TPoD-related metrics have many applications in practical service systems. One notable example is today's multimedia contact centers, wherein customers are served across different channels including phone calls, email, live chat, and social media channels. According to Preece et al. (2018), a collection of insights from call-center practitioners suggest that, besides the 80/20 rule for call contacts, 80% of live chats be answered within 40 seconds, 95% of emails within four hours, and 80% of social media posts within 20 minutes. The TPoD is also widely used in health care. For instance, inpatient wards of Singapore hospitals strive to manage the probability that delay stays below six hours (Shi et al. 2016)). Another relevant example is the Canadian triage and acuity scale (CTAS) guideline that classifies patients in the emergency department into five acuity levels. According to Murray (2003), "CTAS level i patients need to be seen by a physician within w_i minutes $100\alpha_i$% of the time," with $(w_1, w_2, w_3, w_4, w_5) = (0, 15, 30, 60, 120)$ and $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5) = (0.98, 0.95, 0.9, 0.85, 0.8)$. The SL requirements in these examples can all be described by (1) with specific parameters τ and α.

Of course, controlling the TPoD is not the only goal and there do exist other SL metrics in service systems, such as the probability of delay, mean waiting time and probability of abandonment. However, the TPoD is a more general function and it can transform into most of these other metrics. For example, setting $\tau = 0$ reduces to the probability of delay, and integrating the TPoD over τ gives the mean waiting time.

Despite its practical significance, the TPoD is not widely studied in the queueing theory literature, especially in the setting of more realistic queueing models having time-varying arrival rates, multiple servers, and customer abandonment. The reason is two-fold: First, a TPoD constraint in form of (1) involves the careful analysis of the waiting time distribution beyond its mean, exact solutions of which are hardly amenable to analytic results

(except for approximations arising from large-scale limits; see Liu (2018), Liu et al. (2021) for examples). Second, nonstationary queueing models are much more complex than stationary models; the former requires the tracking of the transient queueing trajectory while the latter can be characterized by steady-state values. For example, according to Shi et al. (2016), the six-hour service level in Singapore hospitals is found to be highly unstable and vary between 4% and 37% over time in the course of a day.

In this paper, we attempt to tackle this problem by integrating machine-learning methodologies into the decision making in queueing systems. Specifically, we develop a new recursive algorithm that can be used to determine the required staffing function that achieves time-stable performance for a time-varying queue within a finite time. Following the main idea of reinforcement learning (Sutton and Barto 2018), our algorithm organizes the overall learning process into successive cycles each of which consists of two phases: (i) **Exploitation**: The decision maker (the "agent") generates relevant queueing data via a decision-aware simulator under a candidate solution (the best staffing plan by far), (ii) **Exploration**: Using the newly collected data, improved decisions (i.e., staffing plan) are prescribed and to be used to configure the simulator (the "environment") in the next cycle. This process continues until our staffing decision evolves to optimality. Because we learn from offline (simulated) data, we call this method the *simulation-based offline learning staffing algorithm* (SOLSA). See Figure 1 for an illustration of the recursive learning structure for SOLSA.

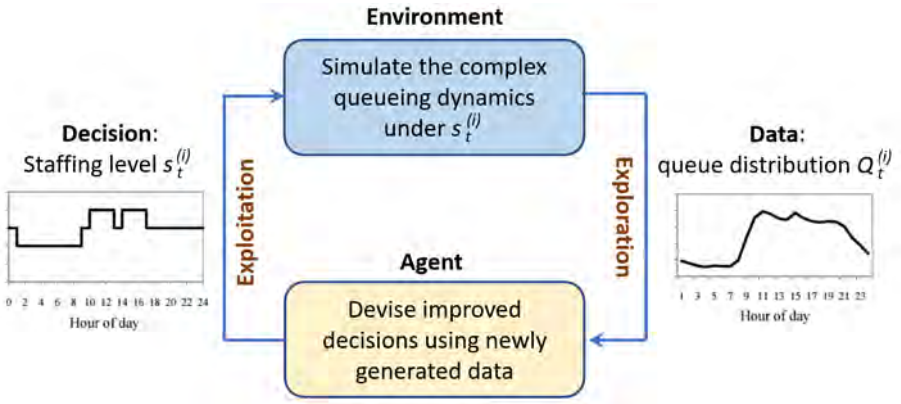

Figure 1: Simulation-based offline learning staffing algorithm (SOLSA).

Our contributions

- We develop a new staffing algorithm, dubbed SOLSA, to achieve time-stable performance in a time-varying Markovian queueing system. Our new method recursively learns the staffing solution using offline queueing data generated by simulation.
- We conduct extensive numerical experiments to evaluate the performance of SOLSA. We also consider practical constraints on staffing. All experiments confirm the efficiency and effectiveness of SOLSA.
- Supplementing the engineering confirmation, we also give the theoretical proof for the convergence of SOLSA.

In the remainder of this paper, we review the relevant literature in Section 1.1. Next, we introduce some preliminary results in Section 2 and formally present our SOLSA algorithm in Section 2.1. In Section 3 we provide engineering confirmations for the effectiveness of SOLSA by conducting numerical studies. In Section 4 we give theoretical proof for the convergence of SOLSA. Finally, we give concluding remarks in Section 5.

1.1 Literature Review

Queues with time-varying conponents have been studied for many years and have applications to many service systems such as call centers, police patrols, emergency departments, etc. See Whitt (2018) for a comprehensive review of time-varying queues. We next only review results on staffing to cope with the time-varying demand, which is closely related to the topic of present paper. Green and Kolesar (1991) introduced the pointwise stationary approximation (PSA) method which approximates a time-varying queue by a sequence of stationary

queues indexed by time and prescribes the required staffing using the steady-state results. PSA works well only when the service times are relatively short and the quality of service is high. To control the probability of delay, Jennings et al. (1996) proposed a modified-offered-load (MOL) approach that determines the staffing level based on the number of busy servers in an infinite-server queue. The MOL idea has been later extended to stabilize the mean waiting time and probability of abandonment in queues with customer abandonment (Liu and Whitt 2012; Liu and Whitt 2014b; Liu and Whitt 2017), and the blocking probability in loss models (Li et al. 2016). Another direction is to revise the staffing decision to account for the non-Poisson property in the time-varyiing arrival process; see He et al. (2016) and Sun and Liu (2021) for the staffing of nonstationary queues with arrival processes modeled by the dispersion ratio and auto-correlations. The present paper draws distinction from the above literature by treating the TPoD, which is much more challenging SL metric than those studied in these papers.

We are not the first to work with the TPoD in time-varying queues. Previously, the primary tools to analyze time-varying queues are approximation models arsing from large-scale limits. See Liu (2018), Liu et al. (2021) for TPoD-oriented staffing formulas developed based on many-server heavy-traffic fluid and diffusion limits. The effectiveness of these formulas depend on the scale of the system: They are shown to be asymptotically "correct" when the scale grows large, but their performance degrades in small-scale models. Another issue is that these fluid and diffusion limits, although much more tractable than their corresponding root queue models, are still not easy to analyze. Defraeye and Van Nieuwenhuyse (2013) developed a simulation-based method to treat TPoD using a searching method, which comes with several costs. First, they cannot provide convergence guarantees. Second, it is computationally expensive in that it can take many iterations to reach satisfactory solutions and in each iteration they need to simulate the wait time distribution which leads to additional implementation complexity and increased computational cost.

Our paper is most closely related to Feldman et al. (2008), which developed an iterative staffing algorithm (ISA) to control the probability of delay (PoD), i.e., the probability the waiting time W_t is greater than 0. The heart of ISA lies in the immediate transformation from the total queue-length Q_t to the PoD; specifically, $\mathbb{P}(W_t > 0) = \mathbb{P}(Q_t \geq s_t)$ where s_t is the number of servers at t. This simple relationship enables ISA to adjust and evaluate the PoD based on the (observable) queue distribution. However, this does not work for the TPoD in form of (1) because the TPoD is not immediately computable using the queue length at t; instead, it is a performance metric drawing from the full distribution of W_t (realized only at future times). In the present paper, we draw inspiration from (Feldman et al. 2008) to use the queue length as the data to inform our decisions. However, we develop a new way to transform the queue length (present "reward") to the waiting time and TPoD (future information), in order to generate improved TPoD-driven staffing policies (future "action").

Our work is related to the large body of literature on simulation optimization (Amaran et al. 2014) and simulation analytics (Nelson 2019). The major distinction is that we focus on solving a constraint satisfaction problem rather than optimizing certain objective values. Our work is also somewhat related to the offline-simulation-online-application method (Hong and Jiang 2019) because our learning process draws heavily from the offline data simulated under previous decisions. However, our staffing policies are not decisioned in real time. Because of this connection, there is potential for our work to leverage the power of green simulation techniques (Feng and Staum 2017) to produce improved solution efficiency. We leave this as an important direction for future work.

2 THE SOLSA ALGORITHM

We consider the $M_t/M/s_t+M$ queueing system having Poisson arrivals with time-varying rate $\lambda(t)$ (the M_t), *independent and identically distributed* (IID) service times following an exponential distribution with rate μ (the M), customer abandonment following exponential abandonment times with rate θ (the $+M$), and a time-varying staffing function (the s_t). Our goal is to seek the right staffing function $s(t)$ in order to achieve a desired TPoD target for some designated τ and α as in (1). Before we introduce the SOLSA algorithm, we first introduce some preliminary results which will serve as useful building blocks.

Queue-based waiting time prediction. SOLSA is an automated procedure for seeking the proper staffing level to control the TPoD SL target. A critical step in SOLSA is to efficiently exploit the simulated queueing data and convert them to the desired SL target. Since our SL target is based on the waiting time

W_t, it may seem reasonable to directly simulate the waiting time data. However, doing so potentially has the following issues: first, the waiting time W_t is not observable at t and becomes available only at some future times. In addition, a customer's potential waiting time data is unavailable after all he/she opts to abandon from the queue before his/her service begins. This gives rise to significant challenges for the data collection in real queueing models or computer simulations. Second, in a computer simulation model, the simulated waiting times are much more challenging to generate than other queue states such as the queue length. For example, the queue length is an integer valued process which increases and decreases by 1 at a time; it can be simulated using simple methods such as *uniformization*. While tracking the waiting times requires tracking all customers' timestamps; it demands more complex implementation and bigger data storage.

Following Feldman et al. (2008), our simulator generates queue length data. Then we convert the queue length to the TPoD; we establish a one-to-one relationship between the distribution of Q_t and that of $(W_t|Q_t)$, the waiting time at t conditional on the queue length distribution Q_t. Specifically, we write

$$\mathbb{P}(W_t > \tau) = \sum_q \underbrace{\mathbb{P}(W_t > \tau | Q_t = q)}_{\equiv \beta(\tau|q)} \cdot \mathbb{P}(Q_t = q), \tag{2}$$

where $\beta(\tau|q)$ is the queue-based TPoD predictor. To compute $\beta(\tau|q)$, we invoke the matrix geometric methods in Latouche and Ramaswami (1999) and model a customer's waiting time as the first passage time until absorption in a pure-death Markov chain, beginning in a transition state q. Specifically, a transition occurs from state q to state $q-1$ if another customer either abandons or enters service with rate $s_t\mu + q\theta$. We remark that the idea of conditioning on the queue length $Q_t = q$ goes beyond the TPoD case in (2) and can work for other SL metrics. For example, if the goal is to control the mean waiting time instead of TPoD, we can again simulate the queue length and then work with the conditional mean waiting time given Q_t, that is $\mathbb{E}[W_t|Q_t = q]$.

In real-world service systems, the staffing function cannot be changed in continuous time and remains a constant within a fixed staffing interval (e.g., 30 minutes in call centers). Our algorithm take into account this practical constraint. Let ΔT_D be the length of time between potential staffing changes. Then, let ℓ represent the ℓ^{th} staffing epoch, where $\ell = 0$ is the initially selected level. This means that the time of the ℓ^{th} potential staffing change is $t_\ell = \ell \cdot \Delta T_D$. Further, if our time horizon is T, then there are $\bar{\ell} = \left\lceil \frac{T}{\Delta T_D} \right\rceil$ total staffing epochs.

2.1 SOLSA

We present the complete algorithm for SOLSA that also incorporates longer staffing durations in Algorithm 1. We begin with an initial staffing function $s_t^{(0)}$ (we assume $s_t^{(0)}$ is sufficiently large to ensure we overly achieve our TPoD goal). Then, we simulate the system and evaluate the queue length distribution at each time t. We note that t is pseudo-continuous, in the sense that it is intended to represent a continuous measurement of time, but is in fact discretized for the purposes of numeric calculation.

Based on the queue length distribution for each time t, we calculate the smallest value of staffing such that our TPoD target can be met in Line 6. This becomes our new candidate staffing function. In Theorem 1, we prove that the TPoD is a monotonically decreasing function with respect to the staffing level, so this optimization can be solved using any root-finding method. We note that Line 8 in Algorithm 1 says that all staffing from time 0 to τ should be set to the selected staffing level s_τ, since there is no queue length distribution before time 0 that we can use. In practice, this warm-up period does not affect the algorithm's performance. Next, in Lines 9–14, we set the true staffing level in each epoch to be the maximum of the optimal candidate staffing levels in that epoch. This ensures that all arriving customers will see the desired service level, regardless of when they arrive during a staffing interval. These lines can be omitted if we wish to allow staffing to vary continuously.

If the maximum difference in the old and new staffing functions is less than a threshold ε, then the algorithm has converged and we have found our optimal staffing function. If we have not yet converged, we repeat the process with the new staffing function. In Section 4 we prove that SOLSA converges in finite time. We also remark that SOLSA is a self-validating algorithm: when SOLSA terminates in some iteration i^*, the latest staffing recommendation $s_t^{(i^*)}$ is automatically the staffing function to achieve the desired TPoD target.

Algorithm 1 SOLSA

Inputs: τ, α, T, ΔT_D, ε
Initialize: $i \leftarrow 0$, $\bar{\ell} \leftarrow \left\lceil \frac{T}{\Delta T_D} \right\rceil$, $\delta_s \leftarrow \infty$, $s_t^{(0)} \leftarrow \infty, \forall t$
while $\delta_s > \varepsilon$ **do**
  Use simulation to evaluate the distribution of $Q_t^{(i)}$ under $s_t^{(i)}$, $\forall t$.
  for each t **do**
    $\hat{s}_{t+\tau}^{(i+1)} \leftarrow \arg\min_s \left\{ s | \sum_q \mathbb{P}(W(s,q) > \tau) \cdot \mathbb{P}(Q_t^{(i)} = q) < \alpha \right\}$
  end for
  $\hat{s}_{\{t|t<\tau\}}^{(i+1)} \leftarrow \hat{s}_{\tau}^{(i+1)}$
  for $\ell = 0$ To $\bar{\ell}$ **do**
    $\tilde{t} \leftarrow [\ell \cdot \Delta T_D, (\ell+1) \cdot \Delta T_D)$
    for each $t \in \tilde{t}$ **do**
      $s_t^{(i+1)} \leftarrow \max_{t \in [\tilde{t}]} \left\{ \hat{s}_t^{(i+1)} \right\}$
    end for
  end for
  $\delta_s \leftarrow \max_t \left| s_t^{(i+1)} - s_t^{(i)} \right|$
  $i \leftarrow i+1$
end while

As soon will be seen in Section 3 that SOLSA usually converges quickly. Nevertheless, there is potential to further reduce the computational complexity by leveraging the power of green simulation methods (Feng and Staum 2017). Specifically, the simulated queue-length data are retrieved from (i) the present operational policy (i.e., staffing rule) and (ii) all random sources (i.e., arrival process, service times and abandonment times) that are generated from simulation. Because all of these random elements are decision blind (independent of the staffing rule), we can reuse them in the next iteration under a different staffing rule.

3 SIMULATION EXPERIMENTS

3.1 A Base Example with Sinusoidal Arrival Rate

In our base model, we consider a sinusoidal arrival rate in form of

$$\lambda(t) = n(a + b \cdot \sin(ct)), \tag{3}$$

with $a = 1$, $b = 0.2$, $c = 1$, and the system scale $n = 100$. We also let $\mu = \theta = 1$, and we set the fixed-staffing interval $\Delta T_D = 0.01$. We let $\varepsilon = 1$, meaning that we stop once the maximum difference in staffing levels between iterations is 1. We utilize the preemptive work-releasing policy when we dismiss a busy server, meaning that if an active server departs, the customer is returned to the head of the queue. This secondary wait is ignored for the purposes of measuring wait time. We let our waiting time target be $\tau = 0.5$ (half of the mean service time). The results are shown in Figure 2 for settings where α ranges from 0.1 to 0.9, utilizing 5000 simulation replications in each iteration. We begin with an initial candidate staffing with 200 servers. We see that our algorithm performs remarkably well at achieving the target TPoD. In addition, to generate the correct staffing functions (as shown in the middle plot), SOLSA converges in only 2 iterations.

We have conducted several other numerical examples, all of which exhibit similar stabilized TPoD values and confirm the good performance of SOLSA. We also observed convergence behavior which primarily depended on the ratio r of the abandonment rate to the service rate $r = \theta/\mu$, which is consistent with the findings in Feldman et al. (2008). Specifically, if $r = 1$, then the algorithm converged in 2 iterations. If $r > 1$, the algorithm typically converged in 3–5 iterations, and if $r < 1$, the it converged in 7–10 iterations.

3.2 Comparing SOLSA with Staffing Methods drawn from Many-Server Heavy-Traffic Limits

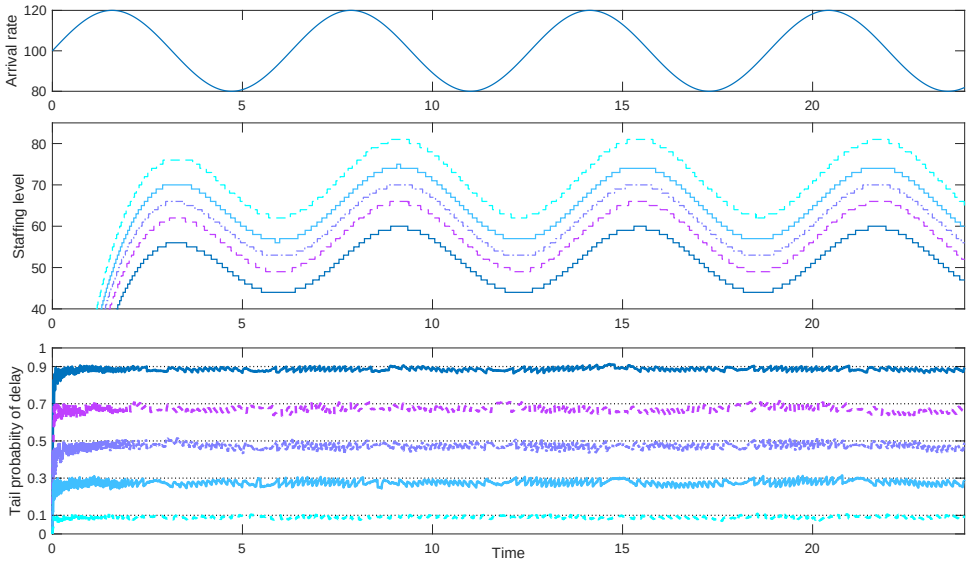

Figure 2: Stabilizing the TPoD at targets $\alpha = 0.1, 0.3, 0.5, 0.7, 0.9$.

An alternative way of setting the staffing level to cope with time-varying demand is by exploring the analysis of the *many-server heavy-traffic* (MSHT) limits of the relevant queueing system; see for example Aras et al. (2018) for recent developments. These limits are derived by allowing the system's scale (i.e., the arrival rate and the number of servers) to grow large. Indeed, staffing functions resulting from these MSHT limits are shown to be asymptotically correct as the scale approaches infinity.

The most relevant result on staffing to stabilize the TPoD is the *two-term Gaussian approximation* (TTGA) proposed in Liu (2018), of which the development draws heavily from the MSHT limits in the efficiency-driven (overloaded) regime Liu and Whitt (2014a). Although TTGA is proven to be effective in general, it exhibits significant performance degradation when the system scale is small or the system is not sufficiently overloaded (with the delay target τ much smaller than the mean service time). We next conduct performance analysis of SOLSA with TTGA (we refer to the specific TTGA staffing formula in Corollary 3 of Liu (2018) under the sinusoidal arrival rate (3)). To substantiate that SOLSA is a more effective and robust method, we hereby specifically focus on the two TTGA-unfavored cases (a) a small delay target, and (b) a small system scale.

In Panel (a) of Figure 3, we continue to use our base example in Section 3.1, but we reduce the delay target to 0.01. We observe that the TPoD results under TTGA begin to deviate from their target values because the smaller delay target is pushing the model out of the efficiency-driven regime. On the other hand, SOLSA continues to work effectively because it is independent of the value of the delay target. Next, we consider a small-scale queueing model having the sinusoidal arrival function in (3) with the scale n reduced from 100 to 10. According to Panel (b) of Figure 3, we conclude that TTGA exhibits somewhat poor performance with the TPoD constantly violating the target, while SOLSA is able to consistently keep the TPoD right below the target. This is not surprising because TTGA requires the system's scale to be large while SOLSA is robust to the system's scale.

3.3 A Practical Constraint: Fixed Staffing Intervals

In previous examples, we allows the staffing function to vary in almost continuous time (with $\Delta T_D = 0$) so that the staffing changes can be sufficient flexible to achieve near constant SL targets. Next, we consider the practical case with $\Delta T_D > 0$; that is, the staffing level cannot be adjusted in continuous time except at discrete time points $t_1, t_2, t_3, \ldots$ with step size T_D (i.e., with $t_i = i \cdot \Delta T_D$).

We again consider the sinusoidal arrival rate in form of (3), with $\mu = \theta = 1$ and step size $\Delta T_D = 0.5$ (which is consistent with call center models wherein the staffing is adjusted once every 30 minutes). In Figure 4, we consider three TPoD targets $\alpha = 0.1, 0.5, 0.9$ and delay target $\tau = 0.1$ (i.e., 10% of the expected service time).

In Figure 4, we see that SOLSA continues to exhibit good performance with TPoD values controlled right below the targets. However, unlike the previous examples, the TPoD curves are much less smooth, showing

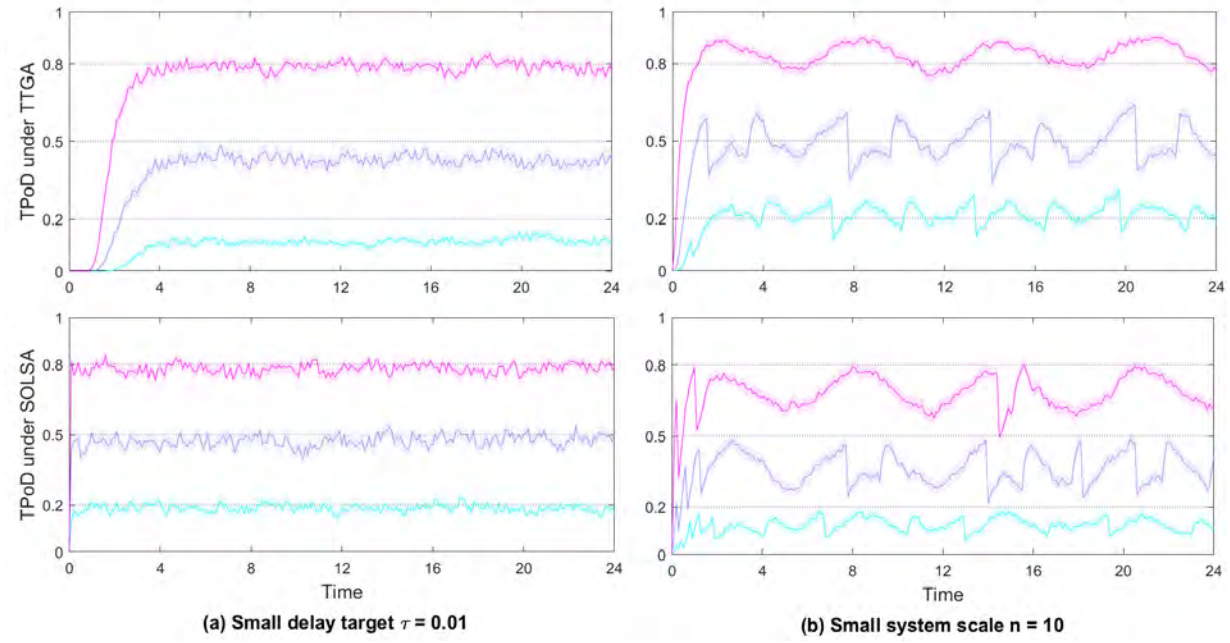

Figure 3: Performance comparison between TTGA (top) and SOLSA (bottom): (a) a smaller delay target $\tau = 0.01$ and (b) a smaller system scale $n = 10$.

the effect of the discretization of the staffing function under the fixed interval constrains. Also, we note that the staffing function now requires more servers than the previous case with $\Delta T_D = 0$ (as in Figure 2). This is because our goal is to ensure that the maximum TPoD be controlled below the target in all intervals, that is

$$\max_{t_i \le t \le t_{i+1}} \mathbb{P}(W_t > \tau) \le \alpha \qquad \text{for all} \quad t_i = i \cdot \Delta T_D, \quad i = 1, 2, \ldots.$$

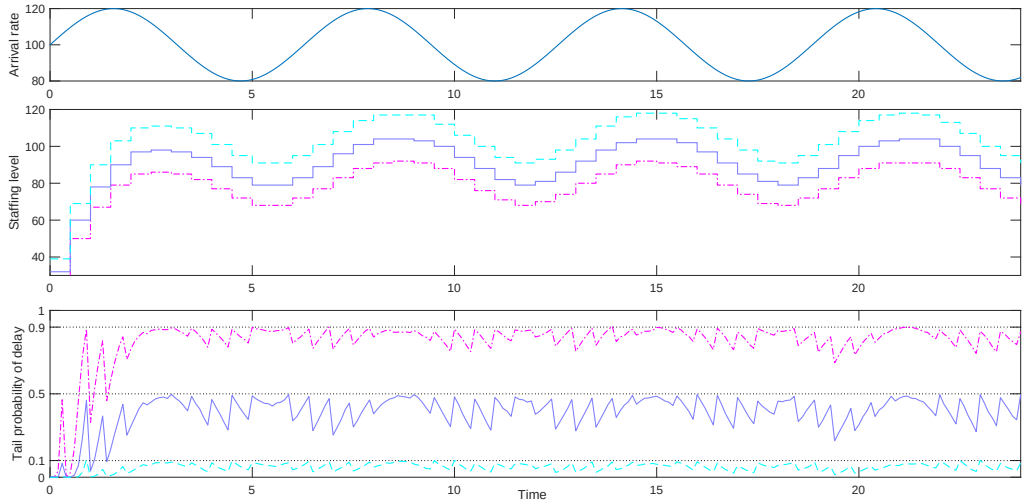

Figure 4: Stabilizing the TPoD at targets $\alpha = 0.1, 0.5, 0.9$ under fixed staffing intervals with $\Delta T_D = 1/2$.

3.4 Realistic Call Center Arrival

Finally, we apply SOLSA to a queueing model informed by a realistic arrival rate function estimated from call center demand data (Green et al. 2001; Feldman et al. 2008). This arrival rate is for a medium-sized financial services call center, where the average service time is 6 minutes, leading to a service rate of $\mu = 10$

per hour. We follow Feldman et al. (2008) to set the abandonment rate, $\theta = 10$. First, we present the arrival rate in the top panel of Figure 5. Notice the incredibly low volume in the early and late hours of the day, as well as the rapid increase in the arrival rate during the morning.

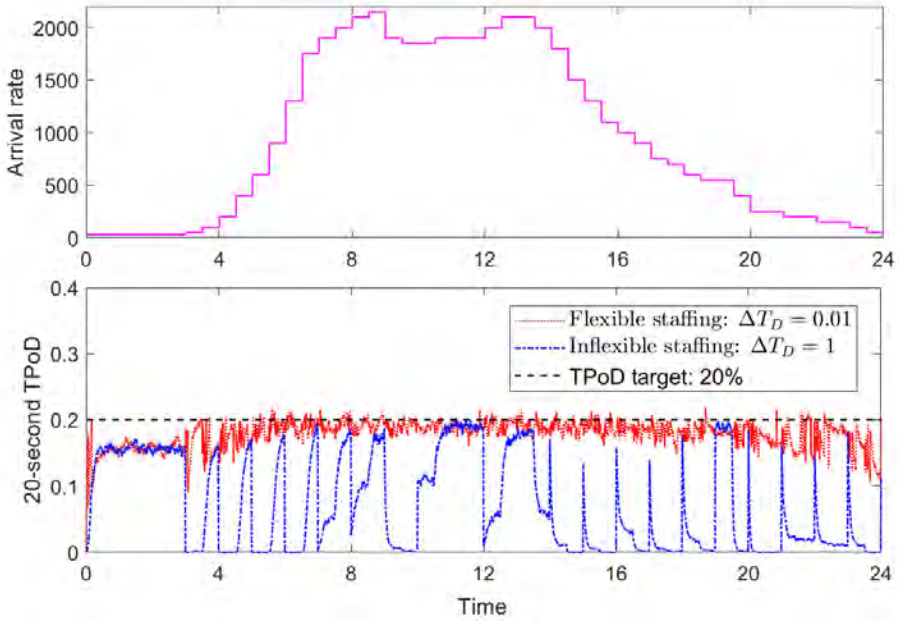

Figure 5: Achieving the 80/20 goal with a call center arrival rate under (a) flexible staffing (dotted line) and (b) inflexible staffing with $\Delta T_D = 1$ (dashed-and-dotted line).

Then, in an effort to meet the common "80/20" rule in call centers, where 80% of the calls need to be answered within 20 seconds, we run SOLSA with the service level goal of $\mathbb{P}(W > 20 \text{ sec}) < 0.2$. The simulated TPoD performance under the SOLSA staffing functions (obtained in 2 iterations) are presented in the bottom panel of Figure 5. First, we show the case of near-continuous staffing, where $\Delta T_D = 0.01$ (flexible staffing, the dotted line), then we show the case where staffing changes once per hour ($\Delta T_D = 1$, the dashed-and-dotted line). As we can see in the flexible-staffing case, the algorithm works well by keeping the TPoD right below the 20% mark (except for the slight overstaffing when the demand rate is close to 0).

In the inflexible-staffing scenario, the target is always achieved, but is usually overachieved due to the practical constraints on the fixed staffing interval. Comparing to the flexible-staffing result, the inflexible staffing with $\Delta T_D = 1$ requires a nearly 10.7% increase in staffing, which may be a cost prohibitive solution. (We also considered an intermediate case with $\Delta T_D = 0.5$ which leads to a 5.1% increase in staffing). We see that as the variability in the arrival rate decreases, our performance function is also less variable.

4 PROVING THE EFFECTIVENESS OF SOLSA

In this section, we prove two important aspects of SOLSA for the special case of a $M_t/M/s_t + M$ system. We will currently assume that staffing is allowed to fluctuate continuously, but we will show that the proofs are easily extended to the case where it changes at discrete time points. First, we prove that the TPoD is monotonic with respect to the staffing level. We will use this result to prove that the algorithm converges.

4.1 Proof of Monotonicity

Theorem 1 Wait Time TPoD Monotonicity. For a $G_t/M/s_t + M$ system where s is held constant over the analysis window, $\mathbb{P}(W(s_t, Q_t) > \tau)$ is a monotonic function that decreases w.r.t s_t for a given distribution Q_t.

Proof of Theorem 1. Using conditioning, we can say that $\mathbb{P}(W(s_t, Q_t) > \tau) = \sum_q \mathbb{P}(W(s_t, q) > \tau) \cdot \mathbb{P}(Q_t = q)$. we observe that $\mathbb{P}(Q_t = q)$ is a fixed quantity in our analysis. Therefore, if $\mathbb{P}(W(s_t, q) > \tau)$ is monotone w.r.t. s_t for all values of q, then we know that $\mathbb{P}(W(s_t, Q_t) > \tau)$, which is a convex combination of monotone functions,

is also a monotone function. Thus, it is sufficient to prove that $\mathbb{P}(W(s_t,q) > \tau)$ is monotone decreasing w.r.t. s_t for all values of q. In order to prove that $\mathbb{P}(W(s+1,q) > \tau) \leq \mathbb{P}(W(s,q) > \tau)$, we will utilize the definition of stochastic ordering. Namely, a random variable A is stochastically less than a random variable B if $\mathbb{P}(A > x) \leq \mathbb{P}(B > x)$. We will denote this relation as $A \leq_{st} B$.

We first observe that for two exponential random variables X and Y, with rates λ_X and λ_Y respectively, $X \leq_{st} Y$ if and only if $\lambda_X \geq \lambda_Y$. This can be easily seen by verifying the definition of the exponential distribution.

We note that the potential waiting time consists of $\kappa = q - s + 1$ independent exponential random variables, and present a visual of the situation in Figure 6 for a system with s servers and q customers. N_i is the service time of the i^{th} stage, where i more service completions or abandonments must be completed before the customer enters service. The rate at which a stage finishes is $\lambda_i = s\mu + (i-1)\theta$ when there are s servers.

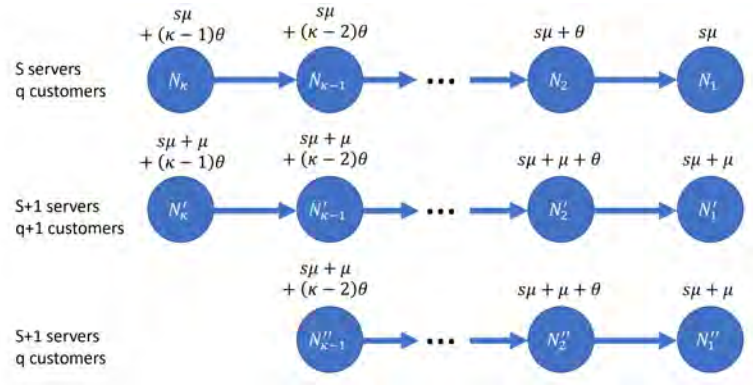

Figure 6: The waiting time of three similar systems.

Suppose we examine a system with $s+1$ servers and $q+1$ total customers, and let N_i' be the service time of the i^{th} stage in this new system. Clearly, $\kappa = \kappa'$, but as we can see in Figure 6, $\lambda_i' = \lambda_i + \mu$.

Now, we can clearly see that the random variable representing the service time in stage i is stochastically less in the larger system. Namely, $N_i' \leq_{st} N_i$ for $i \in [1,\kappa]$. We next take advantage of the following property of stochastic ordering. Namely, if A_i and B_i are independent and $A_i \leq_{st} B_i$, then $\sum_{i=1}^n A_i \leq_{st} \sum_{i=1}^n B_i$. Therefore, we can say that $\sum_{i=1}^{\kappa} N_i' \leq_{st} \sum_{i=1}^{\kappa} N_i$. Furthermore, Figure 6 shows a third system that corresponds to a situation with $s+1$ servers, but only q customers. This system has one less stage to complete, but for each stage, $\lambda_i'' = \lambda_i'$. We can now order these systems as follows.

$$\sum_{i=1}^{\kappa-1} N_i'' \leq_{st} \sum_{i=1}^{\kappa} N_i' \leq_{st} \sum_{i=1}^{\kappa} N_i \quad \text{and} \quad W(s+1,q) \leq_{st} W(s+1,q+1) \leq_{st} W(s,q)$$

$$\text{so that} \quad \mathbb{P}(W(s+1,q) > \tau) \leq \mathbb{P}(W(s,q) > \tau)$$

We have thus shown that $\mathbb{P}(W(s_t,q) > \tau)$ decreases w.r.t. s_t, which therefore means that $\mathbb{P}(W(s_t,Q_t) > \tau) = \sum_q \mathbb{P}(W(s_t,q) > \tau) \cdot \mathbb{P}(Q_t = q)$ is a monotonic function that decreases w.r.t s_t.

4.2 Proving the Convergence of SOLSA

We begin by stating two existing theorems that we will use in our proof of convergence. Our analysis builds on the stochastic comparison between two random variables X and Y; we say $X \geq_{st} Y$ if X *stochastically* dominates Y with $\mathbb{P}(X > t) \geq \mathbb{P}(Y > t)$ for all t.

Theorem 2 (Belzunce et al. (2016) Theorem 2.2.8) Let $\{X_1(\theta)|\theta \in S \subseteq \mathbb{R}\}$ and $\{X_2(\theta)|\theta \in S \subseteq \mathbb{R}\}$ be two families of random variables, and Θ_1 and Θ_2 be two random variables with common support S. If $X_1(\theta) \leq_{st} X_2(\theta), \forall \theta \in S$, $\Theta_1 \leq_{st} \Theta_2$, and $\mathbb{E}[\phi(X_1(\theta))]$ or $\mathbb{E}[\phi(X_2(\theta))]$ or both are increasing in θ, for all real valued increasing function ϕ, then $X_1(\Theta_1) \leq_{st} X_2(\Theta_2)$.

We next restate Theorem 7.1 in Feldman et al. (2008)using Q_t as the distribution of number of customers in the system at time t instead of N_t.

Theorem 3 (**Feldman et al. (2008) Theorem 7.1**) Consider the $M_t/M/s_t+M$ model on the time interval $[0,T]$, starting empty at time 0. Let $r=\frac{\theta}{\mu}$. If $r\geq 1$ and $s_t^{(A)}\leq s_t^{(B)}$ for all $t\in[0,T]$, or if $r\leq 1$ and $s_t^{(A)}\geq s_t^{(B)}$ for all $t\in[0,T]$, then $\{Q_t^{(A)}:0\leq t\leq T\}\leq_{st}\{Q_t^{(B)}:0\leq t\leq T\}$.

Our proof of convergence follows similar logic to that of Feldman et al. (2008), with the important modification of relating staffing choices to that of waiting time, rather than just queue length.

Theorem 4 (**Convergence of SOLSA**) Consider the $M_t/M/s_t+M$ model on the time interval $[0,T]$, starting empty at time 0. Suppose that we consider piecewise-constant staffing functions that can only change at multiples of $D>0$. Suppose that in each iteration n we can obtain the actual stochastic process $\{Q_t^{(n)}:0\leq t\leq T\}$ associated with the staffing function $\{s_t^{(n)}:0\leq t\leq T\}$ (without statistical error). Suppose that $s_t^{(0)}=\infty\,\forall t\in[0,T]$.

(1). If $r>1$, then $s_t^{(n)}\leq s_t^{(m)}$ for all $n>m\geq 0$ and there exists a positive integer n_0 such that

$$s_t^{(SOLSA)}=s_t^{(n_0)}=s_t^{(n)}\forall t \text{ and } n\geq n_0.$$

(2). If $r<1$, then there exist two subsequences, $\{s_t^{(2n)}\}$ and $\{s_t^{(2n+1)}\}$, such that $s_t^{(2n)}\downarrow s_t^{(even)}$ and $s_t^{(2n+1)}\uparrow s_t^{(odd)}$.

$$s_t^{(0)}\geq s_t^{(2n)}\geq s_t^{(2n+2)}\geq s_t^{(2n+3)}\geq s_t^{(2n+1)}\geq s_t^{(1)}$$

for all $t,0\leq t\leq T$, and for all $n\geq 0$. Moreover, there exists a positive integer n_0 such that

$$s_t^{(2n)}=s_t^{(2n_0)}=s_t^{(even)}\geq s_t^{(odd)}=s_t^{(2n_0+1)}=s_t^{(2n+1)}$$

for all $t,0\leq t\leq T$, and for all $n\geq n_0$.

Proof of case 1: $r>1$. First, assume that $s_t^{(0)}=\infty$. We therefore know that $s_t^{(0)}>s_t^{(1)}$. Therefore, by Thm. 7.1, we know that $Q_t^{(0)}\geq_{st}Q_t^{(1)}$. By the construction of our algorithm and the monotonicity of wait time with respect to the staffing level, we know that $s_t^{(1)}$ satisfies the following:

$$\mathbb{P}(W(s_t^{(1)},Q_t^{(0)})>\tau)\leq\alpha<\mathbb{P}(W(s_t^{(1)}-1,Q_t^{(0)})>\tau). \tag{4}$$

(4) shows that at each time t, the staffing level is chosen such that removing one server would increase the TPoD above the target level.

Next, we utilize Thm. 2.2.8 from Belzunce et al. (2016). $\{W(s,q)|q\in\mathbb{N}\}$ is a family of random variables, and it is easy to verify that $W(s,q)\leq_{st}W(s,q)$. Since $Q_t^{(1)}\leq_{st}Q_t^{(0)}$, Thm. 2.2.8 allows us to say that

$$W(s_t^{(1)},Q_t^{(1)})\leq_{st}W(s_t^{(1)},Q_t^{(0)}).$$

By the definition of stochastic comparison, this means that $\mathbb{P}(W(s_t^{(1)},Q_t^{(1)})>\tau)\leq\mathbb{P}(W(s_t^{(1)},Q_t^{(0)})>\tau)$. Combining this result with (4), we see that

$$\mathbb{P}(W(s_t^{(1)},Q_t^{(1)})>\tau)\leq\mathbb{P}(W(s_t^{(1)},Q_t^{(0)})>\tau)\leq\alpha.$$

For each time t, we now seek to find $s_t^{(2)}$, which is found using the distribution of $Q_t^{(1)}$. Let us begin by assuming $s_t^{(2)}=s_t^{(1)}$. Since the TPoD is monotonically decreasing w.r.t. s_t, one of the following must be true:

$$\mathbb{P}(W(s_t^{(2)},Q_t^{(1)})>\tau)\leq\alpha\leq\mathbb{P}(W(s_t^{(2)}-1,Q_t^{(1)})>\tau)\quad\text{or}\quad\mathbb{P}(W(s_t^{(2)},Q_t^{(1)})>\tau)\leq\mathbb{P}(W(s_t^{(2)}-1,Q_t^{(1)})>\tau)\leq\alpha.$$

Therefore, $s_t^{(2)}\leq s_t^{(1)}$. Now, we repeat the process and we see that $Q_t^{(n)}$ is stochastically decreasing with respect to n and $s_t^{(n)}$ is also decreasing in n. Since our staffing levels are non-negative integers and we only use a finite number of values of t, then our algorithm is guaranteed to converge in finitely many steps.

Proof of case 2: $r < 1$. Again, let us assume that $s_t^{(0)} = \infty$, so we know that $s_t^{(0)} > s_t^{(1)}$. Therefore, by Thm. 7.1, we know that $Q_t^{(0)} \leq_{st} Q_t^{(1)}$. By the construction of our algorithm and the monotonicity of wait time w.r.t. the staffing level, we know that $s_t^{(1)}$ satisfies the following:

$$\mathbb{P}(W(s_t^{(1)}, Q_t^{(0)}) > \tau) \leq \alpha < \mathbb{P}(W(s_t^{(1)} - 1, Q_t^{(0)}) > \tau). \tag{5}$$

We again utilize Thm. 2.2.8 from Belzunce et al. (2016) to say that $W(s_t^{(1)}, Q_t^{(1)}) \geq_{st} W(s_t^{(1)}, Q_t^{(0)})$, which means that $\mathbb{P}(W(s_t^{(1)}, Q_t^{(1)}) > \tau) \geq \mathbb{P}(W(s_t^{(0)}, Q_t^{(1)}) > \tau)$. Combining this result with (5), we see that

$$\mathbb{P}(W(s_t^{(1)}, Q_t^{(1)}) > \tau) \geq \mathbb{P}(W(s_t^{(1)}, Q_t^{(0)}) > \tau) \leq \alpha.$$

Therefore, if we again start by assuming that $s_t^{(2)} = s_t^{(1)}$, then we again have two possible situations. Either $\mathbb{P}(W(s_t^{(1)}, Q_t^{(1)}) > \tau) \leq \alpha$ is true, in which case

$$\mathbb{P}(W(s_t^{(2)}, Q_t^{(1)}) > \tau) \leq \alpha \leq \mathbb{P}(W(s_t^{(2)} - 1, Q_t^{(1)}) > \tau)$$

and $s_t^{(2)} = s_t^{(1)}$. Otherwise, $\mathbb{P}(W(s_t^{(1)}, Q_t^{(1)}) > \tau) > \alpha$, in which case, by the monotonicity of the TPoD w.r.t. s_t, we must increase $s_t^{(2)}$ until $\mathbb{P}(W(s_t^{(2)}, Q_t^{(1)}) > \tau) \leq \alpha$ is true. Thus, $s_t^{(2)} \geq s_t^{(1)}$. We can now order our staffing functions as $s_t^{(0)} \geq s_t^{(2)} \geq s_t^{(1)}$ (since $s_t^{(0)} = \infty$), which leads us to the following ordering of the distributions of number in the system: $Q_t^{(0)} \leq_{st} Q_t^{(2)} \leq_{st} Q_t^{(1)}$.

Using our previous arguments, we can see that $s_t^{(2)} \geq s_t^{(3)} \geq s_t^{(1)}$. This pattern will continue, and we will see that $s_t^{(2n)}$ will decrease in n and $s_t^{(2n+1)}$ will increase in n, maintaining the relationship that $s_t^{(2n)} \geq s_t^{(2n+1)}$. Since our staffing levels are again integers and we are using only finitely many values of t, we will again achieve convergence in finitely many steps, though it may be to two different limits, separated by at most 1.

4.3 Convergence with Discretized Staffing Changes

At its core, the modified algorithm remains nearly identical to the original, thus it follows that the algorithm should converge, but we will briefly remark on the reasoning behind the belief, which is indeed correct.

At each iteration, the algorithm selects a candidate staffing function $\hat{s}^{(i+1)}$ based on the previous iteration's staffing function $s^{(i)}$. The selection of this candidate staffing function occurs just as in the original algorithm.

Thus, if $r > 1$, $\hat{s}^{(i+1)} \leq s^{(i)}$. Now, for a given staffing interval ℓ (with its corresponding time interval $\tilde{t}$), the staffing level selected for the next iteration $s_t^{(i+1)}$, which is the max level observed during that interval, must also be less than or equal to $s_t^{(i)}$. Therefore, the original proof of convergence remains effecting and the modified algorithm will converge. By the same reasoning, if $r < 1$ and i is even, we know that $s^{(i+1)} \leq s^{(i)}$. However, if i is odd, then $\hat{s}^{(i+1)} \geq s^{(i)}$. For each staffing interval, the maximum value is certainly at least as large as the value observed in iteration i, therefore $s^{(i+1)} \geq s^{(i)}$. Once again, the original proof of convergence remains effective and the modified algorithm will converge.

5 CONCLUDING REMARKS

This work contributes to the ongoing efforts to achieve stable SL targets in time-varying queueing systems. Our SL goal builds on the TPoD which has many practical applications including customer contact centers and healthcare. We develop a new simulation-based offline reinforcement learning algorithm which recursively learns the desired staffing functions by alternating between two phases: (i) **exploitation** of the present candidate solution to produce decision-aware queueing data, and (ii) **exploration** of improved decisions using newly generated data. The effectiveness and efficiency of our new method are substantiated by (i) comprehensive simulation experiments and (ii) theoretical convergence guarantees. We also consider the practical constraint of fixed staffing intervals. One practical future direction is to expand the scope of SOLSA to include other commonly used SL metrics such as the mean waiting time and probability of abandonment. Key steps in these extensions is to develop one-to-one mapping from the queue lengths to these other metrics. Another important dimension is to generalize SOLSA from queues with Markovian structure to non-Markovian settings, with nonexponential service and abandonment times.

REFERENCES

Amaran, S., N. V. Sahinidis, B. Sharda, and S. J. Bury. 2014. "Simulation Optimization: A Review of Algorithms and Applications". *Annals of Operations Research* 240(1):351–380.

Aras, A. K., X. Chen, and Y. Liu. 2018. "Many-Server Gaussian Limits for Non-Markovian Queues with Customer Abandonment". *Queueing Systems* 89(1):81–125.

Defraeye, M., and I. Van Nieuwenhuyse. 2013. "Controlling Excessive Waiting Times in Small Service Systems with Time-Varying Demand: An Extension of the ISA Algorithm". *Decision Support Systems* 54(4):1558–1567.

Feldman, Z., A. Mandelbaum, W. A. Massey, and W. Whitt. 2008. "Staffing of Time-Varying Queues to Achieve Time-Stable Performance". *Management Science* 54(2):324–338.

Feng, M., and J. Staum. 2017. "Green Simulation: Reusing the Output of Repeated Experiments". *ACM Transactions on Modeling and Computer Simulation* 27(4):1–28.

Green, L., and P. Kolesar. 1991. "The Pointwise Stationary Approximation for Queues with Nonstationary Arrivals". *Management Science* 37(1):84–97.

Green, L. V., P. J. Kolesar, and J. Soares. 2001. "Improving the Sipp Approach for Staffing Service Systems That Have Cyclic Demands". *Operations Research* 49(4):549–564.

He, B., Y. Liu, and W. Whitt. 2016. "Staffing a Service System with Non-Poisson Nonstationary Arrivals". *Probability in the Engineering and Informational Sciences* 30(2):593–621.

Hong, L. J., and G. Jiang. 2019. "Offline Simulation Online Application: A New Framework of Simulation-Based Decision Making". *Asia-Pacific Journal of Operational Research* 36(06):1940015.

Jennings, O. B., A. Mandelbaum, W. A. Massey, and W. Whitt. 1996. "Server Staffing to Meet Time-Varying Demand". *Management Science* 42(10):1383–1394.

Latouche, G., and V. Ramaswami. 1999. *Introduction to Matrix Analytic Methods in Stochastic Modeling*. Philadelphia, Pa: Society for Industrial and Applied Mathematics.

Li, A., W. Whitt, and J. Zhao. 2016. "Staffing To Stabilize Blocking In Loss Models With Time-Varying Arrival Rates". *Probability in the Engineering and Informational Sciences* 30(2):185–211.

Liu, Y. 2018. "Staffing to Stabilize the Tail Probability of Delay in Service Systems with Time-Varying Demand". *Operations Research* 66(2):514–534.

Liu, Y., X. Sun, and K. Hovey. 2021. "Scheduling to Differentiate Service in a Multiclass Service System". *Operations Research* 70(1):527–544.

Liu, Y., and W. Whitt. 2012. "Stabilizing Customer Abandonment in Many-Server Queues with Time-Varying Arrivals". *Operations Research* 60(6):1551–1564.

Liu, Y., and W. Whitt. 2014a. "Many-Server Heavy-Traffic Limits for Queues with Time-Varying Parameters". *Annals of Applied Probability* 24(1):378–421.

Liu, Y., and W. Whitt. 2014b. "Stabilizing Performance In Networks Of Queues With Time-Varying Arrival Rates". *Probability in the Engineering and Informational Sciences* 28(4):419–449.

Liu, Y., and W. Whitt. 2017. "Stabilizing Performance in a Service System with Time-Varying Arrivals and Customer Feedback". *European Journal of Operational Research* 256(2):471–486.

Murray, M. J. 2003. "The Canadian Triage and Acuity Scale: A Canadian Perspective on Emergency Department Triage". *Emergency Medicine* 15(1):6–10.

Nelson, B. L. 2019. "'Some Tactical Problems in Digital Simulation' for the next 10 Years". *Journal of Simulation* 10(1):2–11.

Preece, D., F. Sherlock, and B. Bischoff. 2018. "What Are the Industry Standards for Call Centre Metrics?". *Call Centre Helper*. https://www.callcentrehelper.com/industry-standards-metrics-125584.htm, accessed April 17th.

Shi, P., M. C. Chou, J. Dai, D. Ding, and J. Sim. 2016. "Models and Insights for Hospital Inpatient Operations: Time-Dependent ED Boarding Time". *Management Science* 62(1):1–28.

Sun, X., and Y. Liu. 2021. "Staffing Many-Server Queues with Autoregressive Inputs". *Naval Research Logistics* 68(3):312–326.

Sutton, R., and A. Barto. 2018. *Reinforcement Learning: An Introduction*. Piscataway, NJ: Bradford Books.

Whitt, W. 2018. "Time-Varying Queues". *Queueing Models and Service Management* 1(2):79–164.

AUTHOR BIOGRAPHIES

KURTIS KONRAD is a PhD candidate in the Edward P. Fitts Department of Industrial Engineering at North Carolina State University. He holds a masters degree in Industrial and Systems Engineering from North Carolina State University. His research interest is in stochastic modeling, simulation, and online learning, with interests in applying these methodologies to health care, transportation networks, and other service systems. His email address is kekonrad@ncsu.edu.

YUNAN LIU is an associate professor in the Department of Industrial and Systems Engineering at North Carolina State University. He earned his Ph.D. in Operations Research from Columbia University. His research interests include queueing theory, stochastic modeling, simulation, applied probability, online learning, and optimal control, with applications to call centers, healthcare, and transportation. His work was awarded first place in the INFORMS Junior Faculty Interest Group Paper Competition in 2016. His email address is yliu48@ncsu.edu. His website is https://yunanliu.wordpress.ncsu.edu/.