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

MARKOV PROCESS SIMULATIONS OF SERVICE SYSTEMS WITH CONCURRENT HAWKES SERVICE INTERACTIONS

Andrew Daw

Marshall School of Business
University of Southern California
Bridge Hall 401B
Los Angeles, CA 90089, USA

Galit B. Yom-Tov

Faculty of Data and Decision Sciences
Technion – Israel Institute of Technology
Bloomfield Hall 519
Technion City, Haifa 3200003, ISRAEL

ABSTRACT

In multi-tasked services such as in messaging-based contact centers, parallel service interactions share a mutual dependence through the agent's concurrency. Here, we introduce Markov process simulation methods for bivariate Hawkes cluster service models that are not Markovian by default due to their concurrency dependence. To do so, we propose an alternate construction that maintains extra "shadow" variables for how the process would be under other concurrency levels. We prove that this construction yields an equivalent Markov process, and we show through numerical experiments that its corresponding simulation algorithm is significantly more efficient than the non-Markovian alternatives.

1 INTRODUCTION

Text-based communication has become widely popular in services, where many companies now serve their customers via text channels such as chat, messaging apps, and social media. These new service channels are steadily replacing call (voice-based) channels as the preferred form of customer-business communication. Indeed, a survey conducted by a cloud-based communications provider found that 78% of respondents preferred to text with a company rather than call them (RingCentral 2012). Furthermore, businesses are also showing a preference for conducting service through text. In addition to offering possible operational advantages, text-based contact centers have been recognized as important platforms for reaching potential customers and promoting sales (Tan et al. 2019; Yom-Tov et al. 2020).

Contact centers are complex stochastic systems, and this complexity stems from a combination of behavioral and operational factors. One such factor is concurrency, meaning that the agent can serve more than one customer simultaneously (Tezcan and Zhang 2014; Daw et al. 2023). This capability also distinguishes text-based contact centers from the well-studied call center systems, in which each agent serve only one customer at a time. Concurrency appears in other multi-tasked service environments, such as emergency departments, where physician treat multiple patients in parallel (Kc 2013; Goes et al. 2018), court systems (Bray et al. 2016), where judges manage multiple cases in parallel, and social welfare agencies (Campello et al. 2017), where social workers serve multiple families at the same period. Concurrency also creates dependence among the parallel services. Serving multiple customers at once alters the pace of each individual service interaction, and this then directly affects each service duration. This altered pace of service is due to at least two different mechanisms: First, the agent may be busy serving one customer while others are waiting for a reply, inducing in-process waits. Second, the multitasking itself may add psychological stress on the agent and thus cause slowdown (Kc 2013; Delasay et al. 2019).

Other behavioral complexities are common in contact centers, such as endogenous and reciprocal influences within the service exchange, and these are compounded by its history dependence (Daw 2022). For example, the customer's response time has been shown to be impacted by the agent's response time

(Ilk and Shang 2022). Moreover, recent papers have shown that customers and agents impact one other's behavior simultaneously. For example, Altman et al. (2021) also showed that the customer's expressed sentiments (quantified on a negative-to-positive spectrum) influence agent response times, and vice versa.

Daw et al. (2023) recently proposed to statistically model these asynchronous service encounters using Hawkes processes and found great fit to messaging data from a contact center in industry. Specifically, they introduced a bivariate, marked Hawkes cluster model, in which each conversation is a two-dimensional Hawkes process (one dimension for each party in the service) with random jumps that depend on the operational environment (i.e., concurrency level) in which the process evolves. Hawkes processes are self-exciting point processes, in which each event "excites" the event rate of future events, meaning it increases the probability of another event occurring soon afterwards (Hawkes 1971). Therefore, the future evolution of the process depends on its entire history. There is also a rich stream of research that uses Hawkes processes to model various forms of communication (e.g., Malmgren et al. (2008), Halpin and De Boeck (2013), Fox et al. (2016), Salehi et al. (2019)), as well as other applications. While the Hawkes process does have many nice properties that can support analysis, the introduction of concurrency dependence creates significant hurdles towards doing so. In particular, when the agent's concurrency changes stochastically and discontinuously over time, the necessary state updates require re-calculations across the full history of the service interaction, which violates the Markov property and prevents the use of many tractable methods for simulation and analysis. This paper aims to close that gap.

Operations research views analytical models and methods as essential to the evaluating accuracy and optimizing managerial decision making, such as in developing good routing, staffing, and concurrency policies within contact centers (e.g., Tezcan and Zhang (2014)). Yet, the lack of Markovian properties limits the use of analytical methods. Monte Carlo simulation has been known for many years as a promising companion or approachable alternative to analytical methods (cf. Yom-Tov and Zeitler (2018)). Hence, it is important to offer good techniques for simulating such history dependent processes. There exist many intriguing techniques for simulating Hawkes models (e.g., Hawkes and Oakes (1974), Ogata (1981), Møller and Rasmussen (2005), Dassios and Zhao (2013), Chen and Wang (2020), Chen (2021), Daw (2023)), but these are at best complicated, if not outright invalidated, by the concurrency dependence. For example, only the Ogata (1981) method can handle (deterministic) non-stationarity of model parameters off-the-shelf, but its thinning-based technique cannot handle the almost surely finite cluster setting of our model.

In this paper, we consider both the single conversation model, as proposed by Daw et al. (2023), and the agent-level $M/Hw/\kappa$ queueing system composed of these individual service conversations. That is, the agent-level system features up to κ stochastically evolving service interaction models that are dependent through the shared value of the agent's concurrency. We propose a novel methodology for simulating such systems efficiently. Our algorithm is based upon an alternate construction of the agent-level system that is distributionally equivalent yet able to reinstate the Markov property. We achieve this by keeping "shadow" copies of each conversation's Hawkes correspondence rate under each possible concurrency level (meaning the one true level and the $\kappa - 1$ others). While this procedure may look demanding at first glance due to its increased state space, we show that it actually produces a more efficient simulation algorithm in comparison to the straightforward non-Markovian approach. We introduce pseudo-code for implementing our method, and show its appropriateness by proving that it satisfies the Markov property with the same marginal distributions as the natural construction. We demonstrate the practical performance of this algorithm through several numerical experiments, showing both the superior speed of this procedure and, as a side consequence, the subtleties of the concurrency-dependent service.

2 THE HAWKES CONVERSATIONAL AND AGENT-LEVEL SERVICE MODELS

2.1 Modeling at the Level of the Customer-Agent Service Interaction

Building from Daw et al. (2023), in Definition 1, we begin by modeling each conversation as a concurrency-dependent, bivariate Hawkes process cluster.

Definition 1 (System-Dependent Bivariate Hawkes Service Model) Assuming that every service is initiated by a customer message at time A_0^{c}, let N_t^{c} and N_t^{a} be the respective point processes for the *number of customer and agent messages* sent up to time $t \geq A_0^{\mathsf{c}}$ (excluding the initial), where this pair of point processes is driven by a corresponding pair of stochastic *correspondence rate* intensities, defined with the customer and agent correspondence rates, respectively, given by

$$\begin{aligned}
\mu_t^{\mathsf{c}} &= \sum_{i=0}^{N_t^{\mathsf{c}}} \alpha^{\mathsf{c},\mathsf{c}} e^{-\beta^{\mathsf{c},\mathsf{c}}(t-A_i^{\mathsf{c}})} + \sum_{j=1}^{N_t^{\mathsf{a}}} \alpha^{\mathsf{c},\mathsf{a}} e^{-\beta^{\mathsf{c},\mathsf{a}}(t-A_j^{\mathsf{a}})}, \\
\mu_t^{\mathsf{a}} &= \sum_{i=0}^{N_t^{\mathsf{c}}} \frac{\alpha^{\mathsf{a},\mathsf{c}}}{K_t} e^{-\beta^{\mathsf{a},\mathsf{c}}(t-A_j^{\mathsf{c}})/K_t} + \sum_{j=1}^{N_t^{\mathsf{a}}} \frac{\alpha^{\mathsf{a},\mathsf{a}}}{K_t} e^{-\beta^{\mathsf{a},\mathsf{a}}(t-A_j^{\mathsf{a}})/K_t},
\end{aligned} \tag{1}$$

where A_ℓ^x is the epoch for the ℓth message sent by party x for all $\ell \in \mathbb{Z}_+$ and $x \in \{\mathsf{c},\mathsf{a}\}$ and K_t is the concurrency level at time t. That is,

$$\mathrm{P}\left(N_{t+\delta}^x - N_t^x = n \mid \mathscr{F}_t\right) = \begin{cases} \mu_t^x \delta + o(\delta) & n = 1 \\ 1 - \mu_t^x \delta + o(\delta) & n = 0 \\ o(\delta) & n > 1 \end{cases}, \tag{2}$$

for each $x \in \{\mathsf{c},\mathsf{a}\}$, where $\mathscr{F}_t$ is the natural filtration of the bivariate stochastic process. Here, $\alpha^{x,y} > 0$ for each $x,y \in \{\mathsf{c},\mathsf{a}\}$ is the *instantaneous impact* on the correspondence rate of party x upon a new message sent by party y, and $\beta^{x,y} > 0$ is the analogous *decay rate* of that impact.

The process defined above captures behavioral dependencies between customer and agent response times by connecting the correspondence rate of each party to the other party. Hence, this Hawkes process is not only self-exciting but also mutually-exciting. In this way, the bivariate structure of the model captures the customer and agent relationship. Moreover, this process captures the changes in agent's behavior that stems from the their concurrency level. This is done by slowing the correspondence rate of the agent in two ways. We can see in Definition 1 that both the jumps and the decay rates are divided by K_t. Hence, the magnitude of each instantaneous impact is diminished in the agent's correspondence rate, but the rate at which these impacts fade is also reduced in kind. In fact, one can show through the Hawkes and Oakes (1974) decomposition that, although the pace changes, this form of concurrency-dependence preserves the *mean* number of responses to each message. This mimics the slowdown of the agent due to multitasking, and also models the added in-process wait.

Definition 1 also allows us to contextualize the history dependence of the service. By Equation (2), for each successive event, the Hawkes process is conditionally equivalent to a non-stationary Poisson process when given the history up to and including the most recent event. This endows the model with many valuable features, such as the following thinning-like property. Suppose an event occurred at time t. Letting t^- denote the process values immediately before the event occurs, and letting p_t^{c} be the probability that the event was spurred by side 1, we have

$$p_t^{\mathsf{c}} = \frac{\mu_{t^-}^{\mathsf{c}}}{\mu_{t^-}^{\mathsf{c}} + \mu_{t^-}^{\mathsf{a}}}. \tag{3}$$

If one is analyzing many parallel Hawkes processes, this analogous ratio of intensities can be used to find which Hawkes process created the present activity. We will use this in constructing our simulation procedure in Section 3. Before we can build up to our agent-level service model and consider parallel and dependent services in earnest, we must first formalize the manner in which services end.

2.2 Service Process Stability and Systematic Conversation Closure Rules

Because in practice all conversations end at some point, either by one of the parties involved in the communication (customer or agent) or by the system itself (using some automated closure rule), we want to be sure that our service model is indeed a *cluster* and not a ceaseless point process. Formally, Daw et al. (2023) showed that this is assured by the following closed-form stability condition.

Assumption 1 (Hawkes Service Stability Condition) The instantaneous impact parameters, $\alpha^{x,y} > 0$, and decay rates, $\beta^{x,y} > 0$, hold to the following ratio: $\alpha^{\mathsf{c},\mathsf{a}}/\beta^{\mathsf{c},\mathsf{a}}\alpha^{\mathsf{a},\mathsf{c}}/\beta^{\mathsf{a},\mathsf{c}} < (1-\alpha^{\mathsf{c},\mathsf{c}}/\beta^{\mathsf{c},\mathsf{c}})(1-\alpha^{\mathsf{a},\mathsf{a}}/\beta^{\mathsf{a},\mathsf{a}})$.

Following the common practice used by companies, we will apply a systematic conversation closure rule. Given the filtration of the history-dependent stochastic process up to the current time ($\mathscr{F}_t$), a systematic closure rule is a stopping time that ends the conversation once some observable condition is met. We will assume that the system's closure policy satisfies the following general assumption.

Assumption 2 (Conversation Closure Rule) Let the *closure rule* be a stopping time that is precisely described by a deterministic function of the current concurrency and correspondence rates, i.e. K_t, $\mu_t^{\mathsf{c},\mathsf{c}}$, $\mu_t^{\mathsf{c},\mathsf{a}}$, $\mu_t^{\mathsf{a},\mathsf{c}}$, and $\mu_t^{\mathsf{a},\mathsf{a}}$, which yields a candidate closure time. If no further activity occurs in the conversation after t and before this candidate time, the conversation will be systematically closed at that time.

The intuition of this systematic closure assumption is that at any point in the service, one can express a rule of the style "if there is no activity in the next δ minutes, close the conversation." Assumption 2 says that this duration δ should be a deterministic function of the current state. As an example of such a policy, Daw et al. (2023) proposed a family of level-hitting stopping times that satisfy Assumption 2 and showed the operational value of this assumption from three perspectives: (a) together with Assumption 1, it ensures that the number of messages written in the conversation, $N_\infty^{\mathsf{c}} + N_\infty^{\mathsf{a}}$, is almost surely finite and that every service will end in finite time almost surely; (b) it bounds the probability of closing conversation prematurely (meaning before the customer has truly been served); and (c) it guarantees the asymptotic optimality of the concurrency-based routing procedure suggested by Tezcan and Zhang (2014).

2.3 Collecting the Conversational Service Models into an Agent-Level System

Following Definition 1, we will now propose an agent-level system comprised of several parallel conversation-level interaction models, and these processes will be mutually dependent through K_t. Such a model is similar in spirit to the process-sharing contact-center model proposed by Tezcan and Zhang (2014). There, too, service rate diminishes with the number of concurrent customers.

Definition 2 (Agent-Level Concurrent Service System Model) Assuming that customers arrive according to a homogeneous Poisson process at rate $\lambda > 0$ and that the agent serves no more than $\kappa \in \mathbb{Z}_+$ customers at once, let K_t and Q_t be the number of customers *in service* and *in queue*, respectively, where each service interaction follows the Hawkes service model in Definition 1 with Assumptions 1 and 2. That is, for $k \in \{1,\dots,\kappa\}$, let $\mu_{k:t}^x$, $N_{k:t}^x$, and $A_{k:\ell}^x$ for $x \in \{\mathsf{c},\mathsf{a}\}$ be the correspondence rates, number of messages, and message time-stamps of the kth oldest customer in service at time t, with $A_{k:0}^{\mathsf{c}}$ being the time that each customer began service. Conditioned on the present value of the concurrency K_t, the stochastic dynamics of the K_t services in Definition 1 are mutually independent, and otherwise are mutually dependent.

At the risk of abusing notation, we will hereforward let $\mathscr{F}_t$ denote the natural filtration of the full agent-level model, rather than the interaction level model. Because "queue" can hold multiple connotations across the literature and may be used by some authors to refer to the total number in system, let us clarify here that Q_t is strictly the number waiting. From Definition 2, the agent system can be viewed in Kendall notation as an $M/Hw/\kappa$ queueing model, where conversations are assigned to the agent at times given by an exogenous Poisson process, the service duration distribution is governed by the system-dependent Hawkes processes from Definition 1, and κ is the maximal number of assigned customers.

With the agent-level model now fully defined, let us begin to reason about how it would be simulated. Naturally, this must start with a Hawkes process simulation. Dassios and Zhao (2013) offers a fast and light version of the Hawkes and Oakes (1974) cluster-based approach tailored to exponential kernels, and thus appears particularly attractive here. However, because the switch in concurrency creates discontinuous change in the correspondence rates, this technique must be modified, and, moreover, this will invalidate the Markov property that is typically enjoyed by exponential kernel Hawkes processes (Oakes 1975).

Let us illustrate this. Upon a change of concurrency, updating every ongoing agent correspondence rate, i.e. each $\mu_{k:t}^{\mathsf{a},y} = \sum_{\ell:A_{k:\ell}^y \leq t} \alpha^{\mathsf{a},y}/K_t \exp(-\beta^{\mathsf{a},y}(t - A_{k:\ell}^y)/K_t)$ for each $y \in \{\mathsf{c},\mathsf{a}\}$ and every $k \leq K_t$, would require the full history of every conversation's epochs. Because this is not simply a function of the previous value of the rates, the stochastic process with state vector given by all 4κ correspondence rates (one for each conversation and each $(x,y) \in \{\mathsf{c},\mathsf{a}\}^2$ direction of the conversation) will *not* be a Markov process.

Hence, if we were to simulate the agent-level queueing system using such a state vector, we would also need to carry all of the timestamps for every ongoing conversation to compute the state transitions when the concurrency changes. Naturally, these stochastic storage allocation requirements can be inefficient in implementation. Furthermore, if the model carries all timestamps for every active conversation, then the tractability (both computational and analytical) that is typically associated with exponential decay will be lost. That is, the model could instead have a decay function that is more general than exponential, because Equation (2) implies that general kernels would require such a state space to compute the transitions. This prompts us to consider how to construct this model more efficiently.

3 A MARKOVIAN CONSTRUCTION THROUGH INCLUSION OF "SHADOW" VARIABLES

Let us start this new construction approach with an observation. Because Oakes (1975) implies that multi-dimensional Hawkes processes with exponential kernels can be Markov processes if the state vector tracks each unique intensity, we can immediately see that if the concurrency never changed, the agent-level system would be Markovian without the trailing history of timestamps. Likewise, on any interval where the concurrency is held fixed, the process *is* Markovian.

This suggests the following idea: rather than both carry and dynamically maintain the full sequence of timestamps and then recompute the correspondence rate upon every concurrency change, let us instead carry the "shadows" of what the correspondence rates would be under different concurrencies. That is, even though of course only one concurrency value holds at any one time, we can actually recognize that there are $\kappa - 1$ alternatives hiding in the background in each conversation. Equation (1) implies that when the concurrency switches values, the new agent correspondence rates will be computed using the new concurrency but also with the timestamps generated under all the previous values of the concurrency.

Specifically, this an agent-side phenomenon, as it is this correspondence rate that depends on the concurrency directly. Therefore, let us construct the shadow variables just for the agent correspondence rate. Let us introduce an additional index, $b \in \{1, \dots, \kappa\}$, that adds a dimension to the subscript to record the background shadow variables: $\mu_{k,t,b}^{\mathsf{a},y}$ for $k,b \in \{1,\dots,\kappa\}$ and $y \in \{\mathsf{c},\mathsf{a}\}$. We will refer to the variable with $b = K_t$ as *active*, since we will define this as equal to the true correspondence rate ($\mu_{k,t}^{\mathsf{a},y} = \mu_{k,t,K_t}^{\mathsf{a},y}$), and *inactive* if $b \neq K_t$. To precisely define these variables, let us describe their dynamics. All shadows, both active and inactive, decay between events, as do the customer correspondence rates:

$$\mu_{k:t+s}^{\mathsf{c},\mathsf{c}} = \mu_{k:t}^{\mathsf{c},\mathsf{c}} e^{-\beta^{\mathsf{c},\mathsf{c}} s}, \quad \mu_{k:t+s}^{\mathsf{c},\mathsf{a}} = \mu_{k:t}^{\mathsf{c},\mathsf{a}} e^{-\beta^{\mathsf{c},\mathsf{a}} s}, \quad \mu_{k:t+s,b}^{\mathsf{a},\mathsf{c}} = \mu_{k:t,b}^{\mathsf{a},\mathsf{c}} e^{-\beta^{\mathsf{a},\mathsf{c}} s/b}, \quad \mu_{k:t+s,b}^{\mathsf{a},\mathsf{a}} = \mu_{k:t,b}^{\mathsf{a},\mathsf{a}} e^{-\beta^{\mathsf{a},\mathsf{a}} s/b}, \tag{4}$$

for all $k,b \in \{1,\dots,\kappa\}$ and $s \geq 0$. Additionally, all shadows jump upon a new message in the service conversation. That is, upon a customer message contribution in service k at time t, letting t^- be the moment just before the jump and t^+ be just after, we have that every k-conservation shadow variable jumps, as does the customer correspondence rate:

$$\mu_{k:t^+}^{\mathsf{c},\mathsf{c}} = \mu_{k:t^-}^{\mathsf{c},\mathsf{c}} + \alpha^{\mathsf{c},\mathsf{c}} \quad \text{and} \quad \mu_{k:t^+,b}^{\mathsf{a},\mathsf{c}} = \mu_{k:t^-,b}^{\mathsf{a},\mathsf{c}} + \frac{1}{b} \cdot \alpha^{\mathsf{a},\mathsf{c}}. \tag{5}$$

Likewise, if an agent message occurs in service k at time t, we have

$$\mu^{\mathsf{c,a}}_{k:t^+} = \mu^{\mathsf{c,a}}_{k:t^-} + \alpha^{\mathsf{c,a}} \quad \text{and} \quad \mu^{\mathsf{a,a}}_{k:t^+,b} = \mu^{\mathsf{a,a}}_{k:t^-,b} + \frac{1}{b} \cdot \alpha^{\mathsf{a,a}}. \tag{6}$$

Through this definition, there is no need to re-compute: when the concurrency switches, we can simply switch which shadow is active. When a conversation ends and there is no customer waiting, every shadow variable (and customer correspondence rate) at the departing index will be set to 0.

Let the stochastic process $\{X_t \mid t \geq 0\}$ with state space $\mathbb{Z}_+ \times \mathbb{R}_+^{2\kappa+2\kappa^2}$ be the single-agent service system process, with state at time t given by $X_t = \{Q_t, \mu^{\mathsf{c,c}}_{k:t}, \mu^{\mathsf{c,a}}_{k:t}, \mu^{\mathsf{a,c}}_{k:t,b}, \mu^{\mathsf{a,a}}_{k:t,b} \mid 1 \leq k, b \leq \kappa\}$. Note that the process X_t tracks both K_t active conversations as well as $\kappa - K_t$ inactive conversations with no events and therefore zero correspondence rate, hence, a total of κ conversation processes.

Proposition 1 The stochastic process $\{X_t \mid t \geq 0\}$ is a Markov process with marginal distributions equivalent to the process defined in Definition 2.

Proof. Because the Hawkes agent-level model has deterministic behavior between jump epochs, it can be viewed as a point process. That is, its full sample path can be captured by the sequences of event epochs, or, equivalently, the sequence of times between epochs, where these epochs may be the timestamps within each service interaction, the closure of a service, or the external arrival of a customer. Furthermore, because the probability of any future event is a function of these underlying inter-epoch times, we can simply consider the distribution of the time to the next event when proving that the process X_t is Markov.

Let S be the random variable for the time until the next epoch after the present time t. By definition, S is either the first new customer arrival, intra-service timestamp, or service closure to occur after t. Denoting these as S_{I}, S_{II}, and S_{III}, respectively, we have $S = \min\{S_{\mathrm{I}}, S_{\mathrm{II}}, S_{\mathrm{III}}\}$. For the exogenous arrival time, S_{I}, this is simply an independently drawn exponential random variable. Turning next to S_{II}, let us consider the time to the next contribution within each service interaction.

Let us decompose S_{II} into a collection of underlying possible next points for each service within the agents concurrency, i.e. $S_{\mathrm{II}} = \min_{1 \leq k \leq \kappa} S_{k,\mathrm{II}}$ (where $\mathrm{P}(S_{k,\mathrm{II}} = \infty) = 1$ if $K_t < k$). Recalling the expressions for $\mu^{\mathsf{c}}_{k:t}$ and $\mu^{\mathsf{a}}_{k:t}$, the probability that, given the full history up to time t of service interaction k, no activity occurs in this interaction within the next s units of time is given by

$$\begin{aligned}
\mathrm{P}(S_{k,\mathrm{II}} > s \mid \mathscr{F}_t) &= e^{-\int_t^{t+s}\left(\mu^{\mathsf{c}}_{k,u} + \mu^{\mathsf{a}}_{k,u}\right)\mathrm{d}u} \\
&= e^{-\sum_{i=0}^{N^{\mathsf{c}}_{k:t}} \frac{\alpha^{\mathsf{c,c}}}{\beta^{\mathsf{c,c}}}\left(e^{-\beta^{\mathsf{c,c}}(t+s-A^{\mathsf{c}}_{k:i})} - e^{-\beta^{\mathsf{c,c}}(t-A^{\mathsf{c}}_{k:i})}\right) - \sum_{j=1}^{N^{\mathsf{a}}_{k:t}} \frac{\alpha^{\mathsf{c,a}}}{\beta^{\mathsf{c,a}}}\left(e^{-\beta^{\mathsf{c,a}}(t-A^{\mathsf{a}}_{k:j})} - e^{-\beta^{\mathsf{c,a}}(t-A^{\mathsf{a}}_{k:j})}\right)} \\
&\quad \cdot e^{-\sum_{i=0}^{N^{\mathsf{c}}_{k:t}} \frac{\alpha^{\mathsf{a,c}}}{\beta^{\mathsf{a,c}}}\left(e^{-\beta^{\mathsf{a,c}}(t+s-A^{\mathsf{c}}_{k:i})/K_t} - e^{-\beta^{\mathsf{a,c}}(t-A^{\mathsf{c}}_{k:i})/K_t}\right) - \sum_{j=1}^{N^{\mathsf{a}}_{k:t}} \frac{\alpha^{\mathsf{a,a}}}{\beta^{\mathsf{a,a}}}\left(e^{-\beta^{\mathsf{a,a}}(t-A^{\mathsf{a}}_{k:j})/K_t} - e^{-\beta^{\mathsf{a,a}}(t-A^{\mathsf{a}}_{k:j})/K_t}\right)}.
\end{aligned}$$

We can recognize correspondence rate variables within this expression. That is, we can express the preceding probability as

$$\mathrm{P}(S_{k,\mathrm{II}} > s \mid \mathscr{F}_t) = e^{-\frac{\mu^{\mathsf{c,c}}_{k:t,K_t}}{\beta^{\mathsf{c,c}}}\left(1-e^{-\beta^{\mathsf{c,c}}s}\right) - \frac{\mu^{\mathsf{c,a}}_{k:t,K_t}}{\beta^{\mathsf{c,a}}}\left(1-e^{-\beta^{\mathsf{c,a}}s}\right) - \frac{\mu^{\mathsf{a,c}}_{k:t,K_t}}{\beta^{\mathsf{a,c}}/K_t}\left(1-e^{-\beta^{\mathsf{a,c}}s/K_t}\right) - \frac{\mu^{\mathsf{a,a}}_{k:t,K_t}}{\beta^{\mathsf{a,a}}/K_t}\left(1-e^{-\beta^{\mathsf{a,a}}s/K_t}\right)},$$

where here we now use the active shadow variables to provide the current rates of the intensity, and s only appears in the exponential functions.

Then, turning to S_{III}, we have by Assumption 2 that in the absence of any other activity (i.e., if $S_{\mathrm{III}} < \min\{S_{\mathrm{I}}, S_{\mathrm{II}}\}$), the service closure time can be computed deterministically given the current system state. Naturally, there will be one such value for each conversation, and thus we will let S_{III} refer to the minimum among these deterministic values.

Hence, together, we see that for all $s \in (0, S_{\mathrm{III}})$,

$$\mathrm{P}(S > s \mid \mathscr{F}_t) = e^{-\lambda s} \prod_{k=1}^{K_t} e^{-\mu^{\mathsf{c,c}}_{k:t,K_t}\left(1-e^{-\beta^{\mathsf{c,c}}s}\right) - \mu^{\mathsf{c,a}}_{k:t,K_t}\left(1-e^{-\beta^{\mathsf{c,a}}s}\right) - \mu^{\mathsf{a,c}}_{k:t,K_t}\left(1-e^{-\beta^{\mathsf{a,c}}s/K_t}\right) - \mu^{\mathsf{a,a}}_{k:t,K_t}\left(1-e^{-\beta^{\mathsf{a,a}}s/K_t}\right)},$$

and thus $\mathrm{P}(S > s \mid \mathscr{F}_t) = \mathrm{P}(S > s \mid X_t)$. Therefore, to complete the proof, we are left to show that the state transitions at each epoch also only depend on the present state.

For any intra-service contribution (i.e., S_{II}), this is straightforward, as each shadow variable receives its corresponding jump size. Similarly, between all epochs, each shadow variable decreases according to its own decay rate. In both cases, the queue length remains constant. For newly arriving customers, this either activates an empty slot within the concurrency (i.e., if there exists a k such that $\sum_{s=1}^{\kappa} \mu_{k,t,s}^{x,y} = 0$ for all $x, y \in \{\mathsf{c}, \mathsf{a}\}$, then this service initializes), or otherwise increments Q_t by one. Then, at service closure, Q_t decreases by one if it is not already at 0, and a new conversation initializes if there was one waiting. Finally, if the concurrency changes, whether due to an arrival or a closure, the active shadow variable switches accordingly, and by construction this is available in the present state of X_t. □

Let us note that the shadow and straightforward constructions share a filtration, so it is immediate from the proof of Proposition 1 that the two models are equivalent in distribution. Let us also emphasize what may be a subtle point: the need for more than just the active shadow variables (or, equivalently, the true agent correspondence rates) is not in the event probabilities, but rather in the state updates upon concurrency-changing transitions.

A similar observation can be seen in the simulation procedure for the agent-level system that results from this Markovian construction, as shown in Algorithm 1. When updating the correspondence rates (lines 8, 15, 17, 19, and 26), *all* shadow variables are updated, but when generating new intra-service activity (line 4), only *active* concurrency level variables are used. As we can see in this pseudocode, the proof of Proposition 1 is instructive for the development of this shadow-based simulation. That is, the random elements at each iteration of the loop lie exclusively in generating the S_{I}, S_{II}, and S_{III} variables, and the rest of the steps are concerned with the corresponding updates to the shadow variables, queue length, and concurrency based on which of those times occurs first.

Let us note that this technique employs the aforementioned Dassios and Zhao (2013) exponential-kernel method in line 4 as a sub-routine for generating the next message time given the active correspondence rates. We provide the relevant pseudocode in Algorithm 2 in Appendix A for completeness.

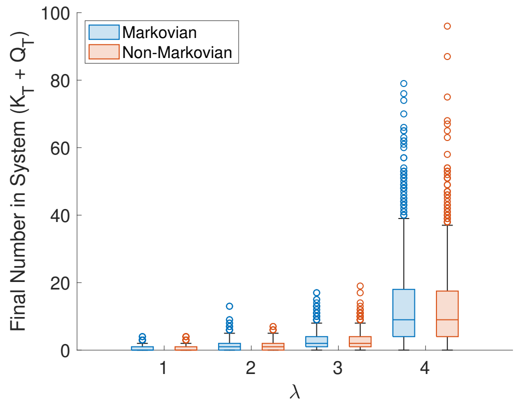
4 COMPUTATIONAL PERFORMANCE AND NUMERICAL EXPERIMENTS

To demonstrate the practical value of the shadow-based simulation, let us now conduct a series of numerical experiments that compare the computational performance of Algorithm 1 with the non-Markovian simulation. For the sake of conserving space, let us simply describe how and where Algorithm 1 would change, rather than providing the complete alternate pseudocode. In essence, the generation of S_{I}, S_{II}, and S_{III} is the same. However, any time the concurrency changes (namely, lines 11 and 24), the correspondence rates will have to be recomputed. To do this, anytime there is a new message in the conversation (lines 17 through 19), we must also append another timestamp to the given conversation's list. On the other hand, at any intra-service update of the correspondence rates (lines 8, 15, 17, and 19), only the active correspondence rate will require computation, which is an advantage of the non-Markovian approach.

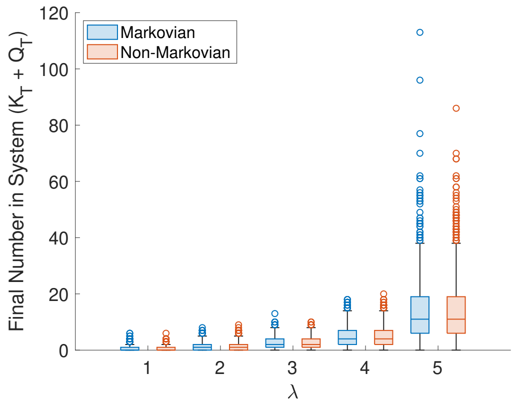
We compare these approaches over 1,024 simulation replications, each with a $T = 500$ time horizon. We use α and β parameter estimates from Daw et al. (2023) (specifically, $\alpha^{\mathsf{c},\mathsf{c}} = 0.843, \alpha^{\mathsf{c},\mathsf{a}} = 14.08, \alpha^{\mathsf{a},\mathsf{c}} = 17.10, \alpha^{\mathsf{a},\mathsf{a}} = 113.7$ and $\beta^{\mathsf{c},\mathsf{c}} = 3.640, \beta^{\mathsf{c},\mathsf{a}} = 38.39, \beta^{\mathsf{a},\mathsf{c}} = 20.37, \beta^{\mathsf{a},\mathsf{a}} = 260.1$, all per hour), we close according to a level-hitting policy, and we vary the customer arrival rate and the maximal concurrency. We have also validated across many different settings of Hawkes and agent-level parameters, and also across varying simulation horizons and process-level comparisons (evaluations at multiple timestamps), but these are not shown due to space limitations. Also omitted for brevity are experiments in simpler settings (like $\kappa = 1$), in which we also validated against other methodology from the literature.

Algorithm 1 Markov Process Simulation of the Agent-Level Service System

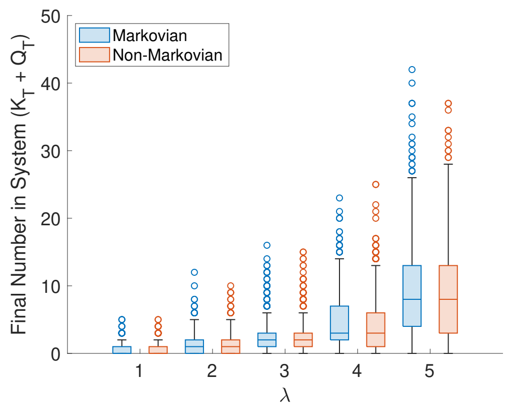
```
1: input: initial interaction rates $\boldsymbol{\mu}^{\mathsf{c,c}} \in \mathbb{R}_+^{\kappa}$, $\boldsymbol{\mu}^{\mathsf{c,a}} \in \mathbb{R}_+^{\kappa}$, $\boldsymbol{\mu}^{\mathsf{a,c}} \in \mathbb{R}_+^{\kappa\times\kappa}$, and $\boldsymbol{\mu}^{\mathsf{a,a}} \in \mathbb{R}_+^{\kappa\times\kappa}$, initial number in service $K \in \mathbb{Z}_+$, initial waiting length $Q \in \mathbb{Z}_+$, simulation horizon $T > 0$ (with $t = 0$ initially), and model parameters $\lambda > 0$, $\boldsymbol{\alpha} \in \mathbb{R}_+^{2\times 2}$, $\boldsymbol{\beta} \in \mathbb{R}_+^{2\times 2}$, $\kappa \in \mathbb{Z}_+$.
2: while $t < T$ do
3:     Generate $S_{\mathrm{I}} \sim \mathsf{Exp}(\lambda)$.
4:     Sample each $S_{\mathrm{II}}^{x,y}$ using $\sum_{k=1}^{K} \mu_k^{\mathsf{c},y}$, $\sum_{k=1}^{K} \mu_{k,K}^{\mathsf{a},y}$, and $K$.    ▷ See Algorithm 2.
5:     Set $S_{\mathrm{II}} = \min_{x,y\in\{\mathsf{c,a}\}} S_{\mathrm{II}}^{x,y}$.
6:     Set $S_{\mathrm{III}}$ as the deterministic first stopping time given $Q$, $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$.
7:     if $t + \min\{S_{\mathrm{I}}, S_{\mathrm{II}}, S_{\mathrm{III}}\} > T$ then
8:         Set $t = T$ and decay $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$ according to Equation (4).
9:     else if $S_{\mathrm{I}} < S_{\mathrm{II}}$ and $S_{\mathrm{I}} < S_{\mathrm{III}}$ then
10:        if $K < \kappa$ then
11:            Initialize a new service in $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$, and increment $K$.
12:        else
13:            Increment $Q$.
14:        end if
15:        Update $t$ to $t + S_{\mathrm{I}}$, and decay $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$ according to Equation (4).
16:    else if $S_{\mathrm{II}} < S_{\mathrm{I}}$ and $S_{\mathrm{II}} < S_{\mathrm{III}}$ then
17:        Decay $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$ according to Equation (4).
18:        Using each $S_{\mathrm{II}}^{x,y}$ and $\boldsymbol{\mu}^{x,y}$, determine the source and type of the new activity via Equation (3).
19:        Update $t$ to $t + S_{\mathrm{II}}$, and apply jump to $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$ via Equations (5) and (6).
20:    else if $S_{\mathrm{III}} < S_{\mathrm{I}}$ and $S_{\mathrm{III}} < S_{\mathrm{II}}$ then
21:        if $Q > 0$ then
22:            Initialize new service and decrement $Q$.
23:        else
24:            Decrement $K$, and set corresponding variables in $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$ to 0.
25:        end if
26:        Update $t$ to $t + S_{\mathrm{III}}$, and decay $\boldsymbol{\mu}^{\mathsf{c,c}}$, $\boldsymbol{\mu}^{\mathsf{c,a}}$, $\boldsymbol{\mu}^{\mathsf{a,c}}$, and $\boldsymbol{\mu}^{\mathsf{a,a}}$ according to Equation (4).
27:    end if
28: end while
```



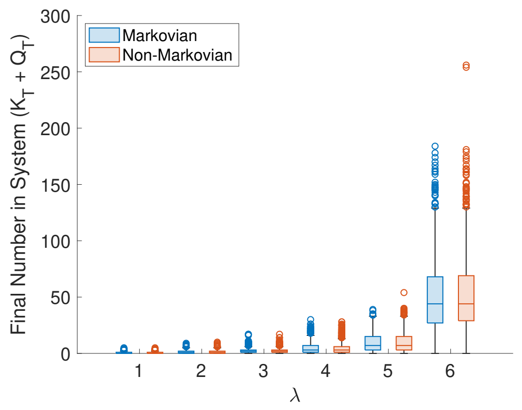
(a) $\kappa = 3$



(b) $\kappa = 6$



(c) $\kappa = 12$



(d) $\kappa = 24$

Figure 1: Validation of the distributions of the number in system at the end of the simulation horizon.

4.1 Comparing Markovian Simulation Accuracy and Performance

Let us begin by discussing the algorithm validation. In Figure 1, we can see that the two simulation procedures (Algorithm 1 and the straightforward re-computation approach) produce indistinguishable distributions across all scenarios. The fact that these algorithms match in their output distribution is assured as a consequence of Proposition 1, but nevertheless it is valuable to verify this empirically.

Next, in Figure 2, we investigate the algorithm performance in terms of its computational complexity. Specifically, Figure 2 shows the experiment run-times for both algorithms in each simulation scenario. While each could have conceivable advantages – the Markov simulation does not re-compute, but the non-Markov has fewer states – we can see that Algorithm 1 outperforms in every setting. Moreover, the gap between run-times grows as the load on the system increases, and this is consistent across the levels of κ. Increases in κ lead to slight increases in run-time at each possible value of λ and in each algorithm, but this change is much smaller than the difference in algorithms or the difference between λ values.

This leads to a main takeaway of this paper. Maintaining the extra shadow variables in the Markovian simulation shows its value through computational savings. Figure 2 shows that switching concurrency is quite costly in the non-Markovian approach, and thus we see the wider gap between the methodologies when the system load is higher. When the system load is lower, such as at $\lambda = 1$, the run-time difference is more likely due to the dynamically allocating storage for the conversation timestamps, because in these

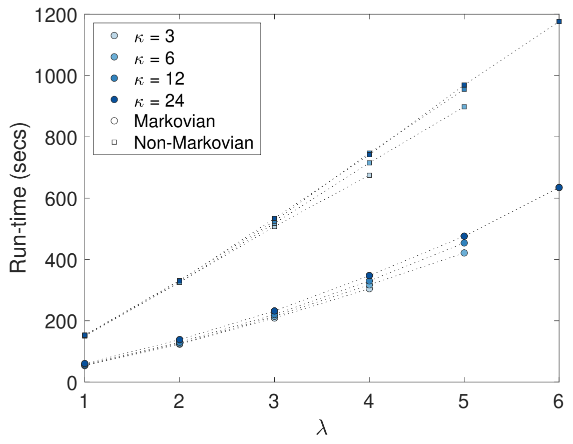


Figure 2: Simulation run-time comparisons across λ and κ.

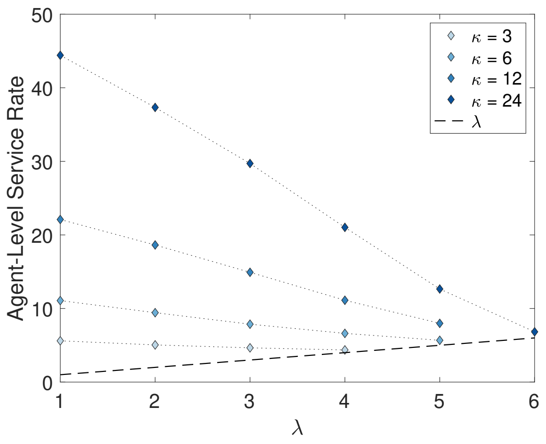


Figure 3: Concurrency-dependent total service rate as a function of λ and κ.

settings the concurrency switches infrequently. On the other hand, at larger customer arrival volumes, the concurrency will change more often and thus the non-Markovian simulation will bear this additional computational burden. For these reasons, as λ grows the gap in computation times grows, too.

4.2 Identifying System Complexities Through the Simulation Experiment

As a second takeaway, let us observe that this experiment has also offered fundamental insight into the concurrent service system as well. In Figure 3, we see can see that the service rate actually *decreases* when the arrival rate *increases*. We typically think of these quantities as unrelated ingredients that together may be used to compute utilization or offered load, but, here, because of the concurrency, the service rate actually depends on the arrival rate directly. For intuition on this phenomenon, consider that when the arrival rate increases, this means that the agent is more frequently pushed into higher levels of concurrency, and this slows down all ongoing conversations.

Similarly, when κ increases with λ unchanged, more customers are served overall each hour, but each individual service is slower on average. This is particularly apparent if the customer arrival volumes are large. For example, we can see this in Figure 3 at $\lambda = 5$. Each κ value at this point constitutes double the maximal concurrency from the point below it, but the agent-level service rates do not double from one maximal concurrency setting to the next.

Table 1: Mean final numbers in system ($K_T + Q_T$) with 95% confidence intervals in parenthesis.

κ	$\lambda = 1$	$\lambda = 2$	$\lambda = 3$	$\lambda = 4$	$\lambda = 5$	$\lambda = 6$
3	0.51 (*0.46, 1.07*)	1.37 (*1.27, 2.83*)	2.90 (*2.74, 5.96*)	13.1 (*12.3, 26.9*)		
6	0.56 (*0.51, 0.61*)	1.29 (*1.20, 1.37*)	2.34 (*2.21, 2.48*)	4.59 (*4.37, 4.81*)	14.0 (*13.3, 14.8*)	
12	0.57 (*0.52, 0.62*)	1.22 (*1.13, 1.31*)	2.36 (*2.21, 2.52*)	4.71 (*4.46, 4.96*)	9.13 (*8.71, 9.56*)	
24	0.53 (*0.48, 0.58*)	1.25 (*1.16, 1.34*)	2.44 (*2.29, 2.58*)	4.89 (*4.60, 5.18*)	10.3 (*9.74, 10.8*)	49.7 (*47.7, 51.7*)

To further elaborate on this point, let us introduce Table 1, which displays the sample mean and 95% confidence intervals for the same final number in system distributions as in Figure 1. In this format, we can again observe interesting phenomenon. Specifically, let us note that at $\lambda = 4$, the mean number in system *increases* with statistical significance when the maximum concurrency is increased from $\kappa = 6$ to $\kappa = 24$. The same can be said at $\lambda = 5$ from $\kappa = 12$ to $\kappa = 24$. So, while the agent-level system appears to have more capacity, there are actually more customers in the system, and Figure 3 would suggest that this is because the service rates have slowed.

5 CONCLUDING DISCUSSION AND FUTURE DIRECTIONS

While the focus of this paper has been restricted to customer-agent service interactions in text-based services, the modeling and simulation philosophy can apply to other concurrency-dependent settings. For example, analogous models of team work (with higher dimensional multivariate models) may be natural future applications of this approach. Furthermore, state-dependent discontinuous service rates may have implications for nuanced interpretations of stability (e.g., Dong et al. (2015), Dong (2022)). We anticipate that the shadow-based construction would hold similar value in simulating such cases.

There are also opportunities for future work in expanding the simulation. For example, Chen (2021) and Chen and Wang (2020) provide steady-state simulation algorithms for the Hawkes process, and Algorithm 1 is only transient. Similarly, Daw (2023) provides a combinatorially inspired decomposition for sampling Hawkes clusters in which the closure is not necessarily a stopping time, and this may cover service settings not applicable here. Finally, we are also interested in a full enterprise-level model, where can incorporate this agent-level model and its nested interaction-level model to further study system-wide operational problems like routing or staffing.

In closing, let us summarize what we feel to be the true value of this simulation algorithm and Markovian modeling approach. This methodology provides a faster way to identify insights for the stochastic service model, such as studied in Daw et al. (2023) or as seen here in the endogenous service rate observations we have found in the numerical experiments. Hence, we hope that these techniques leads to further discovery of valuable managerial insights, as we intend to pursue.

A DASSIOS-ZHAO SAMPLING PROCEDURE FOR GENERATING INTRA-MESSAGE TIMES

Algorithm 2 Dassios-Zhao Sampling Procedure for Exponential Kernel Hawkes Processes

```
 1: Generate $U^{\mathsf{c,c}}, U^{\mathsf{c,a}}, U^{\mathsf{a,c}}, U^{\mathsf{a,a}} \overset{\mathsf{iid}}{\sim} U(0,1)$.
 2: Set $D^{\mathsf{c},y} = 1 + \beta^{\mathsf{c},y} \log(U^{\mathsf{c},y}) / \sum_{k=1}^{K_t} \mu_{k:t}^{\mathsf{c},y}$ and $D^{\mathsf{a},y} = 1 + \frac{\beta^{\mathsf{a},y}}{K_t} \log(U^{\mathsf{a},y}) / \sum_{k=1}^{K_t} \mu_{k:t,K_t}^{\mathsf{a},y}$ for each $y \in \{\mathsf{c},\mathsf{a}\}$.
 3: **for** $x, y \in \{\mathsf{c},\mathsf{a}\}$ **do**
 4:     **if** $D^{x,y} > 0$ **then**
 5:         Set $S_{\mathrm{II}}^{x,y} = -K_t^{\mathbf{1}\{x=\mathsf{a}\}} \log(D^{x,y}) / \beta^{x,y}$.
 6:     **else**
 7:         Set $S_{\mathrm{II}}^{x,y} = \infty$.
 8:     **end if**
 9: **end for**
10: **return** $S_{\mathrm{II}}^{\mathsf{c,c}}$, $S_{\mathrm{II}}^{\mathsf{c,a}}$, $S_{\mathrm{II}}^{\mathsf{a,c}}$, and $S_{\mathrm{II}}^{\mathsf{a,a}}$.
```

REFERENCES

Altman, D., G. B. Yom-Tov, M. Olivares, S. Ashtar, and A. Rafaeli. 2021. "Do Customer Emotions Affect Agent Speed? An Empirical Study of Emotional Load in Online Customer Contact Centers". *Manufacturing & Service Operations Management* 23(4):854–875.

Bray, R. L., D. Coviello, A. Ichino, and N. Persico. 2016. "Multitasking, Multiarmed Bandits, and the Italian Judiciary". *Manufacturing & Service Operations Management* 18(4):545–558.

Campello, F., A. Ingolfsson, and R. A. Shumsky. 2017. "Queueing Models of Case Managers". *Management Science* 63(3):882–900.

Chen, X. 2021. "Perfect Sampling of Hawkes Processes and Queues with Hawkes Arrivals". *Stochastic Systems* 11(13):264–283.

Chen, X., and X. Wang. 2020. "Perfect Sampling of Multivariate Hawkes Processes". In *Proceedings of the 2020 Winter Simulation Conference*, edited by K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 469–480. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Dassios, A., and H. Zhao. 2013. "Exact Simulation of Hawkes Process with Exponentially Decaying Intensity". *Electronic Communications in Probability* 18:1–13.

Daw, A. 2022. “Services Shaped by History”. *Queueing Systems* 100(3-4):409–411.

Daw, A. 2023. “Conditional Uniformity and Hawkes Processes”. *Mathematics of Operations Research*.

Daw, A., A. Castellanos, G. Yom-Tov, J. Pender, and L. Gruendlinger. 2023. “The Co-Production of Service: Modeling Services in Contact Centers Using Hawkes Processes”. *Management Science*.

Delasay, M., A. Ingolfsson, B. Kolfal, and K. Schultz. 2019. “Load Effect on Service Times”. *European Journal of Operational Research* 279(3):673–686.

Dong, J. 2022. “Metastability in Queues”. *Queueing Systems* 100(3-4):413–415.

Dong, J., P. Feldman, and G. B. Yom-Tov. 2015. “Service Systems with Slowdowns: Potential Failures and Proposed Solutions”. *Operations Research* 63(2):305–324.

Fox, E. W., M. B. Short, F. P. Schoenberg, K. D. Coronges, and A. L. Bertozzi. 2016. “Modeling E-mail Networks and Inferring Leadership Using Self-Exciting Point Processes”. *Journal of the American Statistical Association* 111(514):564–584.

Goes, P. B., N. Ilk, M. Lin, and J. L. Zhao. 2018. “When More is Less: Field Evidence on Unintended Consequences of Multitasking”. *Management Science* 64(7):2973–3468.

Halpin, P. F., and P. De Boeck. 2013. “Modelling Dyadic Interaction with Hawkes Processes”. *Psychometrika* 78(4):793–814.

Hawkes, A. G. 1971. “Spectra of Some Self-Exciting and Mutually Exciting Point Processes”. *Biometrika* 58(1):83–90.

Hawkes, A. G., and D. Oakes. 1974. “A Cluster Process Representation of a Self-Exciting Process”. *Journal of Applied Probability* 11(3):493–503.

Ilk, N., and G. Shang. 2022. “The Impact of Waiting on Customer-Instigated Service Time: Field Evidence from a Live-Chat Contact Center”. *Journal of Operations Management* 68(5):487–514.

Kc, D. S. 2013. “Does Multitasking Improve Performance? Evidence from the Emergency Department”. *Manufacturing & Service Operations Management* 16(2):168–183.

Malmgren, R. D., D. B. Stouffer, A. E. Motter, and L. A. Amaral. 2008. “A Poissonian Explanation for Heavy Tails in E-mail Communication”. *Proceedings of the National Academy of Sciences* 105(47):18153–18158.

Møller, J., and J. G. Rasmussen. 2005. “Perfect Simulation of Hawkes Processes”. *Advances in Applied Probability* 37(3):629–646.

Oakes, D. 1975. “The Markovian Self-Exciting Process”. *Journal of Applied Probability* 12(1):69–77.

Ogata, Y. 1981. “On Lewis’ Simulation Method for Point Processes”. *IEEE Transactions on Information Theory* 27(1):23–31.

RingCentral 2012, Dec. “Texting for Work on the Rise Per RingCentral Survey”. Press Release.

Salehi, F., W. Trouleau, M. Grossglauser, and P. Thiran. 2019. “Learning Hawkes Processes from a Handful of Events”. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, 12694–12704. Vancouver, Canada.

Tan, X. J., Y. Wang, and Y. Tan. 2019. “Impact of Live Chat on Purchase in Electronic Markets: The Moderating Role of Information Cues”. *Information Systems Research* 30(4):1248–1271.

Tezcan, T., and J. Zhang. 2014. “Routing and Staffing in Customer Service Chat Systems with Impatient Customers”. *Operations Research* 62(4):943–956.

Yom-Tov, G. B., L. Yedidsion, and Y. Xie. 2020. “An Invitation Control Policy for Proactive Service Systems: Balancing Efficiency, Value and Service Level”. *Manufacturing & Service Operations Management* 23(5):1077–1095.

Yom-Tov, G. B., and T. Zeitler. 2018. “Delay Guarantee Planning of Call-back Options in Time-varying Service Systems”. In *Proceedings of the 2018 Winter Simulation Conference*, edited by M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, 2084–2094. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

ANDREW DAW is an Assistant Professor of Data Sciences and Operations in the Marshall School of Business at the University of Southern California. He is interested in applied probability and stochastic modeling for service operations, particularly so at the intersection of behavioral operations. Much of his recent work has involved the self-exciting Hawkes process, where he has used the history-dependent stochastic process to model behavior that depends on interactions, influences, and impulses. Prior to joining USC Marshall, he received his doctorate from the School of Operations Research and Information Engineering at Cornell University, where he was supported as a National Science Foundation Graduate Research Fellow. His email address is andrew.daw@usc.edu, and his website is https://faculty.marshall.usc.edu/Andrew-Daw/.

GALIT B. YOM-TOV is an Associate Professor in the Faculty of Data and Decision Sciences at the Technion - Israel Institute of Technology, and co-director of the SEELab (https://seelab.net.technion.ac.il). Her research interests include service science and behavioral operations, in particular, in healthcare and contact centers environments. Her research aims to build models for understanding the impact of customer and agent behavior on service systems and to incorporate these behaviors into operational models of such systems. She leads a multidisciplinary research approach that applies a combination of Data Science and Stochastic Modeling to archives of digital traces from service systems. She currently serves as an associate editor for Manufacturing and Service Operations Management and Operations Research journals. Her email address is gality@technion.ac.il, and her website is https://gality.net.technion.ac.il.