OPERATIONS RESEARCH
Vol. 64, No. 4, July–August 2016, pp. 867–885
ISSN 0030-364X (print) | ISSN 1526-5463 (online)

http://dx.doi.org/10.1287/opre.2016.1499

Modeling Load and Overwork Effects in Queueing Systems with Adaptive Service Rates

Mohammad Delasay
Tepper School of Business, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, delasays@cmu.edu

Armann Ingolfsson, Bora Kolfal
Alberta School of Business, University of Alberta, Edmonton, Alberta T6G 2R6, Canada
{armann.ingolfsson@ualberta.ca, bkolfal@ualberta.ca}

Servers in many real queueing systems do not work at a constant speed. They adapt to the system state by speeding up when the system is highly loaded or slowing down when load has been high for an extended time period. Their speed can also be constrained by other factors, such as geography or a downstream blockage. We develop a state-dependent queueing model in which the service rate depends on the system "load" and "overwork." Overwork refers to a situation where the system has been under a heavy load for an extended time period. We quantify load as the number of users in the system, and we operationalize overwork with a state variable that is incremented with each service completion in a high-load period and decremented at a rate that is proportional to the number of idle servers during low-load periods. Our model is a quasi-birth-and-death process with a special structure that we exploit to develop efficient and easy-to-implement algorithms to compute system performance measures. We use the analytical model and simulation to demonstrate how using models that ignore adaptive server behavior can result in inconsistencies between planned and realized performance and can lead to suboptimal, unstable, or oscillatory staffing decisions.

Keywords: state-dependent queues; quasi-birth-and-death; service operations; behavioral operations; load; overwork; staffing.
Subject classifications: probability: stochastic model applications; queues: Markovian.
Area of review: Operations and Supply Chains.
History: Received August 2013; revisions received March 2015, December 2015; accepted March 2016. Published online in *Articles in Advance* June 6, 2016.

1. Introduction

Most capacity planning and queueing models are based on an assumption that servers work at a constant speed. This assumption is a simplification of reality, and researchers have documented various ways in which the assumption is violated. A typical finding is that servers speed up when the system *load*, usually measured by the system occupancy, increases (Edie 1954, KC and Terwiesch 2012, Gans et al. 2010, Kuntz et al. 2014, Chan et al. 2014, Tan and Netessine 2014). Some researchers have hypothesized, and in some cases verified, that such speedup cannot be sustained indefinitely, and therefore servers slow down when load remains high over an extended period (Sze 1984, Gans et al. 2010, Dietz 2011), a situation that is referred to as *overwork* (KC and Terwiesch 2009).

We believe it is important to study adaptive server behavior (speedup and slowdown) from three perspectives: (1) empirically, to establish fundamental knowledge about whether, how, and why servers in real systems adapt; (2) analytically, to develop tractable models that incorporate the main aspects of real server behavior; and (3) prescriptively, to investigate the impact on solution quality of accounting for adaptive server behavior in models used to generate solutions. In this paper we focus on the second perspective: developing tractable models. We also touch on the third perspective by illustrating possible negative impacts on solution quality that might result from ignoring adaptive server behavior.

We extend the commonly used fixed-server-speed Erlang C capacity-planning model to allow server speed to depend on load and overwork, we derive the performance measures of the extended model, and we investigate the "errors" that can result from using the Erlang C model to predict performance or prescribe capacity levels. Extending the Erlang C model to allow server speed to depend on load is not difficult (Jackson 1963 is an early example). It is more challenging to allow server speed to also depend on overwork while maintaining model tractability, and it appears that no one has undertaken that analysis.

The modeling challenge is to operationalize overwork through an index that does not require detailed memory of the past history of the system. The indices that we investigate are all based on keeping track of *high-load periods* through the concept of a *k-partial busy period*, which is a period during which k or more of the s servers in the system are busy serving users. Typically, we select k equal to the number of servers, but our model also

allows k to be less than s. During a k-partial busy period, one could track various cumulative overwork measures. We describe three cumulative measures: number of service completions, elapsed time, and elapsed time summed over all busy servers. We focus on the first measure, which leads to a tractable two-dimensional Markov chain, with one dimension corresponding to load and the other dimension corresponding to overwork. The Markov chain is a quasi-birth-and-death (QBD) process with a special structure that was previously identified by van Leeuwaarden and Winands (2006). We build on their work to design efficient algorithms to compute system performance measures.

2. Literature Review

We survey two streams of related work: Empirical studies that document that service speed varies when system conditions change (Section 2.1) and analytical and simulation studies that investigate the performance of state-dependent systems or develop optimal service rate control policies (Section 2.2).

2.1. Empirical Studies

Batt and Terwiesch (2016, Figure 3) categorize the effects of system load increase on service speed as either speedup or slowdown. We use the speedup and slowdown categories to organize our review of empirical papers.

Speedup effects have been observed in many contexts. Edie (1954), the earliest empirical study we know of, reports that toll booth holding times (service times) at the Lincoln Tunnel and the George Washington Bridge decrease with traffic volume and the number of open booths because (1) the collectors expedite the operation under the backed-up traffic pressure and (2) the drivers are more likely to have their payment ready before they reach the toll booth. Sze (1984) anecdotally reports that Bell System telephone operators work faster during overloaded periods to work off the queue. Tan and Netessine (2014) and Staats and Gino (2012) document speedup behavior of restaurant waiters and bank loan application processors, respectively.

There are also several reports of speedup effects in healthcare systems. KC and Terwiesch (2009) find evidence of speedup in two distinctly different operations in a hospital, patient transportation and cardiothoracic surgery, where patient transportation time and patient length of stay (LOS) decrease with the number of busy transporters and the number of occupied beds, respectively. KC and Terwiesch (2012) and Chan et al. (2014) argue that when a cardiac intensive care unit (ICU) is full and a new patient needs to be admitted, care providers are likely to discharge the most stable patient early. Batt and Terwiesch (2016) observe speedup effects for several emergency department (ED) patient care tasks. Kuntz et al. (2014) report a nonlinear relation between bed occupancy level and hospital LOS and confirm that physicians use their discretion regarding early discharge when load is very high. However, load does not appear to affect LOS when the utilization is low.

Turning to slowdown mechanisms, Sze (1984) lists the slowdown behavior of telephone operators after long high-load periods without relief, as one of the complexities of workforce management in call centers. Dietz (2011) observes a positive correlation between call volume and average service time when call volume is high and hypothesizes that "shift fatigue" leads to longer service times. Gans et al. (2010) define *run length*, the number of services an agent has performed since the last gap of longer than one hour, as a proxy for how overworked a call center agent is, and they find that higher run length is associated with longer average call times for some agents.

KC and Terwiesch (2009) show that the load effect for in-hospital patient transportation time and cardiothoracic surgery patient LOS is not permanent and overwork, measured as the excess load over a specified number of past time periods, eventually slows down transporters and medical staff. Batt and Terwiesch (2016) also find evidence of slowdown in such ED tasks as lab specimen collection and X-ray imaging. Armony et al. (2015) show that service speed decreases with load when the number of patients within an ED is high and conjecture that ED medical staff slow down when they feel overwhelmed by the system pressure. Slowdown can also be caused by customers rather than servers, as Chan et al. (2014) demonstrate, by analyzing the relationship between wait time in the ED and LOS in the ICU, for patients that were admitted to the ICU from the ED.

2.2. Analytical and Simulation Studies

Markovian analytical models typically use service *rates* rather than service *time*; therefore, we focus on service rates in the remainder of the paper, with the understanding that service rate is the inverse of service time (at least when the service rates are non-state-dependent). Analytical queueing models with load-dependent service rates have a long history, dating back to Jackson (1963), who obtains the joint probability distribution of the queue lengths in a network with Markovian routing and where each station has Markovian service rates that depend on the queue length at that station. Welch (1964) and Harris (1967) extend modeling to generally distributed service times, but their work is limited to a single server and the load dependence is limited to dependence on whether or not the system is empty when service begins (Welch 1964) or to dependence on the queue length when service begins (Harris 1967).

Several researchers (Crabill 1972, George and Harrison 2001, Hopp et al. 2007, Alizamir et al. 2013) have investigated models where service rates are assumed to be controllable, with costs assigned to customer waiting and service effort. These researchers find that speedup is optimal in certain situations—specifically, George and Harrison (2001) find that the optimal single-server service rate increases with queue length and Hopp et al. (2007) and Alizamir

et al. (2013) show that in some situations it is optimal to complete service early, in settings with discretionary task completion criteria.

Other researchers have formulated models that are meant to capture speedup effects seen in hospitals (Berk and Moinzadeh 1998, Chan et al. 2014) and in production-line experiments (Powell and Schultz 2004). All of these papers investigate conditions under which speedup improves long-term throughput. In the ICU setting investigated by Chan et al. (2014), the impact on throughput is complicated by readmission of patients that were discharged prematurely.

Turning to slowdown, Aksin and Harker (2001) and Dong et al. (2015) both model multiserver systems where service rates decrease with load. Aksin and Harker's (2001) work is motivated by a customer contact centre in which service rates decrease with load because agents require access to a shared information system with limited capacity, whereas Dong et al. (2015) extend the Erlang A model to incorporate slowdown effects. Slowdown could also occur if customers who wait longer have longer average service times, as modeled by Chan et al. (2014) and Selen et al. (2015). Selen et al. (2015) formulate their model as a QBD, as we do, but their phase variable is defined differently from ours.

We make the following contributions: (1) We introduce a tractable overwork measure that can be incorporated in a Markov chain by adding a single state variable. (2) We extend the Erlang C model to a two-dimensional Markov chain that incorporates flexible dependence of service rates on load and overwork. (3) We build on the work of van Leeuwaarden and Winands (2006) to develop efficient algorithms to compute steady-state probabilities and system performance measures. (4) We demonstrate that the Erlang C model results in significant errors, even if one uses a well-chosen "representative service rate," to predict the performance of systems with load and overwork-dependent service rates. We further show that a one-dimensional model, where load is used as a proxy for overwork, does not provide good approximations to the system performance measures, even when the load-dependent service rates are chosen to optimize the quality of the approximation. (5) We illustrate several types of unintended consequences that can result from using the Erlang C model to prescribe staffing in systems with load and overwork-dependent service rates, including oscillatory staffing, unstable staffing, and convergence to a suboptimal staffing level.

3. Operationalizing Overwork

In this section we discuss alternative ways to operationalize the concept of overwork—a situation where the system has been under a heavy load for an extended time period. We review how other researchers have operationalized overwork and related constructs to measure overwork empirically, we discuss why it is challenging to incorporate previously proposed overwork measures in a stochastic model, and we outline a family of overwork measures that can be incorporated in a Markov chain through the addition of only one state variable.

Overwork might result in human servers (for example, hospital porters or call center agents) becoming fatigued, resulting in a slowdown in service delivery. In settings where a "server" represents a bundle of human and other resources, overwork could influence service rate through more complex mechanisms. If one views a bed and the resources needed to serve a patient in a cardiothoracic surgery ward as a server, for example, then KC and Terwiesch (2009) argue that overworked physicians tend to prescribe more testing, which can delay patient discharge.

We know of two empirical studies that operationalize overwork. KC and Terwiesch (2009) measure overwork as the average excess load over the last T time periods. For example, suppose that the average load is 2 requests per server per hour, but in the previous $T = 5$ hours, each server has handled 3 requests per hour, on average. The resulting overwork is $3 - 2 = 1$ *extra* request per server per hour for a total amount of $1 \times 5 = 5$ units of overwork per server. Alternatively, the KC and Terwiesch (2009) operationalization could be applied separately for each server. In a similar vein, Gans et al. (2010) define "run length" as the number of calls an agent has answered since the last service gap of longer than an hour to measure overwork for an individual agent. This means that if the last sufficiently long service gap of an agent ended one hour ago and during the previous hour the agent handled 50 calls, then the agent overwork is measured as 50.

Although the overwork variables that KC and Terwiesch (2009) and Gans et al. (2010) defined can be measured empirically, they require historical information about service completions in previous time periods, and from a stochastic modeling perspective, using these definitions requires a high-dimensional state space. To be concrete, suppose we define the following variables:

$$\begin{aligned} I(t) &= \text{total number of users in the system at time } t,\\ B_u(t) &= 1 \text{ if server } u \text{ is busy at time } t,\ 0 \text{ otherwise},\\ &\quad u = 1, \ldots, s,\\ J_u(t) &= \text{overwork measure for server } u \text{ at time } t,\\ &\quad u = 1, \ldots, s. \end{aligned}$$

If one were to use the KC and Terwiesch (2009) operationalization at the server level, then to calculate $J_u(t)$, one would need information about not only the current value of $B_u(t)$ but the history of $B_u(t)$ from $t - T$ to t, assuming that each period is one time unit. Suppose that one approximates this history with the $T + 1$ indicator values $B_u(t - T), B_u(t - T + 1), \ldots, B_u(t)$. One could formulate a Markov chain with state variables $I(t)$ and $B_u(t - T)$, $B_u(t - T + 1), \ldots, B_u(t)$ for $u = 1, \ldots, s$, and these variables would suffice to calculate $J_u(t)$ for $u = 1, \ldots, s$. If the system capacity is N, then the cardinality of the state space is $(2^s + N - s)^{(T+1)}$. The exponential growth in the cardinality of the state space as the number of servers and T grow is known as the *curse of dimensionality* (Bellman 1961).

To mitigate the curse of dimensionality, we (1) model overload at the system level instead of the server level and (2) use cumulative overwork measures that do not require memory of the past history of the system. Modeling at the system level removes the need for state variables that keep track of whether individual servers are busy. The total number of busy servers can be obtained as $B(t) = \min(I(t), s)$. The cumulative overwork measures that we propose have the properties that they increase during high-load periods and decrease during low-load periods (which is analogous to the KC and Terwiesch 2009 measure). We formalize a high-load period as a k-partial busy period, which commences when an arrival finds $k-1$ users in the system and ends when a departure leaves the system with $k-1$ users (Artalejo and Lopez-Herrero 2001). In our computational experiments, we set $k = s$, the number of servers, but our model also allows values of $k < s$.

Here are three possibilities for the amount by which a cumulative overwork measure increases per infinitesimal time unit δ during a high-load period:

—$E(t)$: δ, that is, keep track of *E*lapsed time;

—$C(t)$: $\delta \times B(t)$, that is, keep track of *C*umulative service time;

—$J(t)$: number of service (or *J*ob) completions in $(t, t+\delta]$.

During a low-load period infinitesimal time unit, we assume that $E(t)$ decreases by δ and that $C(t)$ decreases by $\delta \times B(t)$. We assume that $J(t)$ decreases by the number of "overwork reduction events" that occur in $(t, t+\delta]$, where overwork reduction events occur at rate $(s-B(t))\gamma$, which is proportional to the number of idle servers. When all servers are idle, all three measures decrease at rate $s\gamma$.

We simulated $E(t)$, $C(t)$, and $J(t)$ in a 10-server Erlang C system with arrival rate of 8/hr, service rate of 1/hour, $k = s = 10$, and $\gamma = 3$/hr. Figure 1 shows sample paths for $E(t)$, $C(t)$ (normalized by dividing by s), and $J(t)$. The three measures behave similarly: They increase together during high-load periods and decrease together during low-load periods. Table 1, which is based on the same simulation, extended to 2,500 hours, shows high correlations between the three measures.

We focus on the count-based overwork measure $J(t)$ because it leads to a Markov chain with a discrete state space and because it behaves similarly to the time-based overwork measures $E(t)$ and $C(t)$.

Figure 1. Sample paths for $E(t)$, $C(t)$, and $J(t)$.

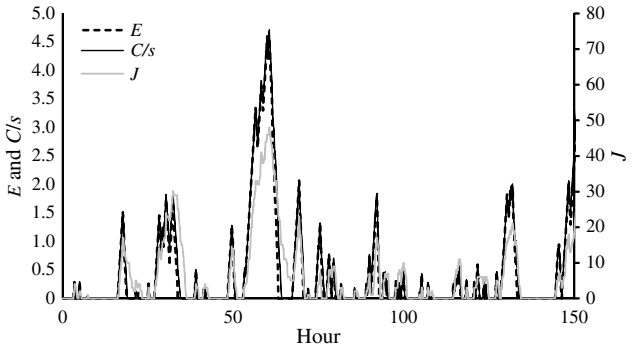


Table 1. Correlations between $E(t)$, $C(t)$, and $J(t)$.

	$E(t)$	$C(t)$	$J(t)$
$E(t)$	1		
$C(t)$	0.95	1	
$J(t)$	0.86	0.95	1

4. Model Formulation

In our generalization of the Erlang C model, users arrive according to a Poisson process with rate λ and wait in an infinite-capacity first-come-first-served queue for the first available of s parallel and identical servers. Servers are never idle when users are waiting. The rate $\mu_{i,j}$ at which every busy server completes its current service depends on the state variables $I(t)$ (number of users in the system, which measures load) and $J(t)$, formally defined as,

$$J(t) = \text{overwork up to time } t, \text{ which increases by one with every service completion in a } k\text{-partial busy period that does not end the } k\text{-partial busy period } (I(t^-) \geqslant k+1), \text{ and decreases by one unit at a time at rate } (s-B(t))\gamma_{i,j} \text{ outside a } k\text{-partial busy period } (0 \leqslant I(t^-) \leqslant k). \quad (1)$$

The resulting Model M_1 is a continuous time Markov chain with state space $\Omega_1 = \{(i,j)\colon i = 0, 1, \ldots; j = 0, 1, \ldots\}$. We define I^1_∞ and J^1_∞ to be the steady-state random variables for $I(t)$ and $J(t)$ in Model M_1.

We do not specify service time distributions but the assumption that the holding time in each system state is exponential with a state-dependent mean, $\mu_{i,j}$, implies that service time distributions are phase-type. Khudyakov et al. (2010, as reported in Gans et al. 2010) comment that modeling service times with phase-type distributions facilitates a useful shift in perspective, from viewing service times as static entities to viewing them as the outcome of a dynamic process of user-server interaction.

We represent particular values of $B(t)$, $I(t)$, and $J(t)$ by b, i, and j. The following four transitions from a general state (i,j) characterize Model M_1:

User arrival, with rate λ, resulting in a transition to state $(i+1, j)$.

User departure within a k-partial busy period, with rate $b\mu_{i,j}$, resulting in a transition to state $(i-1, j+1)$, when $i \geqslant k+1$.

User departure outside or ending a k-partial busy period, with rate $b\mu_{i,j}$, resulting in a transition to state $(i-1, j)$, when $1 \leqslant i \leqslant k$.

Overwork decrease outside k-partial busy period, with rate $(s-b)\gamma_{i,j}$, resulting in a transition to state $(i, j-1)$, when $0 \leqslant i \leqslant k-1$ and $j \geqslant 1$.

If all service rates are equal ($\mu_{i,j} = \mu, \forall (i,j) \in \Omega_1$) and $\lambda < s\mu$, then the marginal steady-state distribution of I^1_∞

matches the system size distribution of the Erlang C model. In the remainder of the paper, we make five assumptions.

ASSUMPTION 1. *Service rates are independent of the load if the load is sufficiently high*: *There exists an* $s' \geqslant s$ *such that* $\mu_{s'+l,j} = \mu_{s',j}$ *for* $j = 0, 1, \ldots$ *and* $l = 1, 2, \ldots$.

The appropriate value for s' could depend on the information that servers have about queue length and on how they react to that information. If the queue is invisible to the servers, then $s' = s$ might be appropriate. If the queue is visible but servers cannot differentiate among queue lengths that are longer than q, then $s' = s + q$ might be appropriate.

ASSUMPTION 2. *Overwork accumulates up to a finite threshold* m.

Typically, we expect service rates to decrease with j and if they do, then $\mu_{i,m}$ represents the smallest possible service rate for any given i. Even if this expectation is judged not to be realistic, if one selects the threshold m to be sufficiently large, then we expect $\Pr(J_\infty^1 > m)$ to be negligible.

Assumption 2 restricts the state space, Ω_1, for Model M_1 to $\Omega_2 = \{(i, j)\colon i = 0, 1, \ldots; j = 0, \ldots, m\}$ and we refer to the resulting model as M_2 and show its transition diagram in Figure 2.

ASSUMPTION 3. *All service rates are positive*: $\mu_{i,j} > 0$, $\forall (i, j) \in \Omega_2$.

ASSUMPTION 4. $k \leqslant s$.

Setting $k = s$ has the desirable consequence that overwork reduction events can occur whenever one or more servers are idle, but we also allow $k < s$.

ASSUMPTION 5. $s\mu_{s',m} > \lambda$.

In Section 5.2 we show that under Assumption 3, Model M_2 is stable if and only if Assumption 5 holds.

5. Model Analysis and Algorithms

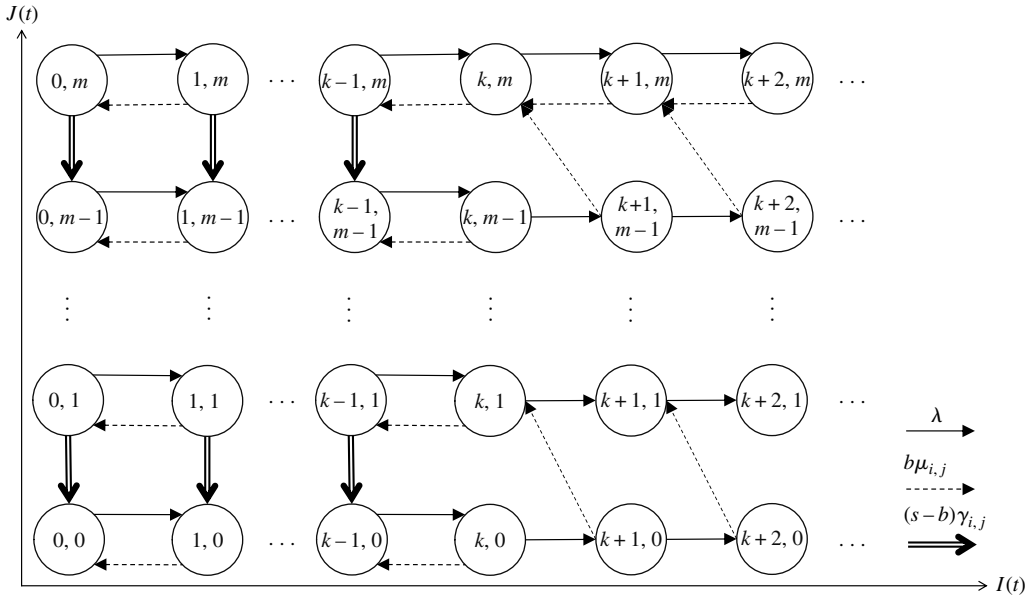
In this section we establish that Model M_2 is a QBD process, whose special structure for levels above k (first recognized by van Leeuwaarden and Winands 2006), together with general matrix-geometric results, allows tractable calculation of steady-state probabilities and other important system performance measures.

We formulate Model M_2 as a level-dependent QBD process with I as the level and J as the phase, that is, we order the system states lexicographically as $\{L(0), L(1), \ldots\}$, where the set $L(i) = \{(i, 0), \ldots, (i, m)\}$ contains the $m + 1$ states in Level $i \geqslant 0$. Model M_2 is a QBD process because the transitions for the level are skip free. The process has the following block-tridiagonal transition matrix:

$$\begin{pmatrix}
\mathbf{B} & \mathbf{A}_0 & & & & & & \\
\mathbf{A}_2^{(1)} & \mathbf{A}_1^{(1)} & \mathbf{A}_0 & & & & & \\
\ddots & \ddots & \ddots & & & & & \\
 & \mathbf{A}_2^{(k)} & \mathbf{A}_1^{(k)} & \mathbf{A}_0 & & & & \\
 & & \mathbf{A}_2^{(k+1)} & \mathbf{A}_1^{(k+1)} & \mathbf{A}_0 & & & \\
 & & & \ddots & \ddots & \ddots & & \\
 & & & & \mathbf{A}_2^{(s')} & \mathbf{A}_1^{(s')} & \mathbf{A}_0 & \\
 & & & & & \mathbf{A}_2^{(s')} & \mathbf{A}_1^{(s')} & \mathbf{A}_0 \\
 & & & & & & \ddots & \ddots & \ddots
\end{pmatrix}.$$

We show expressions for the matrix blocks in the online appendix (available as supplemental material at http://dx.doi.org/10.1287/opre.2016.1499). The matrix block $\mathbf{A}_0$ for arrivals (which increase the level by one) is constant for

Figure 2. State transition diagram for Model M_2.



all levels. The matrix blocks $\mathbf{A}_2^{(i)}$ for service completions (which decrease the level by one) become constant at level s'. The matrix blocks $\mathbf{A}_1^{(i)}$ correspond, in general, to transitions within a level. In Model M_2, only Levels 0 to $k-1$ have such transitions. The diagonal entries of $\mathbf{A}_1^{(i)}$ are chosen to ensure that $\mathbf{A}_0+\mathbf{A}_1^{(i)}+\mathbf{A}_2^{(i)}$ has zero row sums.

Given that $\mathbf{A}_0$, $\mathbf{A}_1^{(i)}$, and $\mathbf{A}_2^{(i)}$ are all constant for $i \geqslant s'$, we can also formulate Model M_2 as a level-independent QBD process with larger boundary blocks by considering all states in Levels 0 to $s'-1$ to be boundary states. Although the level-independent QBD representation is more compact, the level-dependent one is more computationally efficient and it is structurally similar to the fixed-service-rate Erlang C model, as we illustrate later in this section.

Let the row vector $\boldsymbol{\pi}_i = (\pi_{i,0}, \ldots, \pi_{i,m})$ denote the steady-state probabilities of the states in $L(i)$, $i \geqslant 0$. In a level-dependent QBD process, the steady-state probabilities satisfy

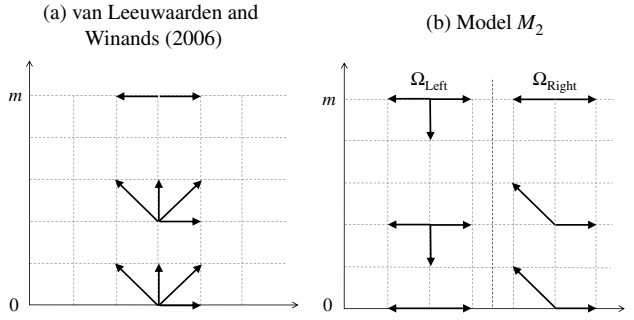
$$\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i \mathbf{R}^{(i)}, \quad i \geqslant 0, \tag{2}$$

where the rate matrix for $L(i)$, $\mathbf{R}^{(i)}$, records the expected time spent in $L(i+1)$ between two consecutive visits to $L(i)$, expressed in the expected time spent in $L(i)$ (Neuts 1981).

In a level-independent QBD process, the constant rate matrix $\mathbf{R}$ is the minimal nonnegative solution to $\mathbf{R}^2\mathbf{A}_2 + \mathbf{R}\mathbf{A}_1 + \mathbf{A}_0 = \mathbf{0}$ (Neuts 1981), and it must be computed numerically in most cases. van Leeuwaarden and Winands (2006) introduce a class of *level-independent* QBD processes with transitions limited to the ones shown in Figure 3(a), for which the entries in $\mathbf{R}$ are explicitly obtainable by counting lattice paths between two particular states multiplied by a constant path probability. The useful property that facilitates the computation of $\mathbf{R}$ in these QBD processes is that if the process leaves $L(i)$, then it will return back to $L(i)$ in a finite number of transitions.

We partition Ω_2 into $\Omega_{\text{Right}} = \Omega_2 \cap \{L(k+1), L(k+2), \ldots\}$ (states in levels $k+1$ onward) and $\Omega_{\text{Left}} = \Omega_2 \cap \{L(0), L(1), \ldots, L(k)\}$ (states in levels 0 to k). Figure 3(b) shows the transition types in Model M_2. The transitions in Ω_{Right} are a subset of the transitions in Figure 3(a) (the class identified by van Leeuwaarden and Winands 2006), but the transitions in Ω_{Left} are not. We use these observations to separately compute the rate matrices in Ω_{Right} and Ω_{Left}. For Ω_{Left}, we propose an efficient algorithm to compute the rate matrices. For Ω_{Right}, we exploit the special structure in Figure 3(a) and we extend the work of van Leeuwaarden and Winands (2006) by allowing the transition rates to be level and phase-dependent.

Figure 3. Allowed transitions.



We begin the analysis at Level k (the border between Ω_{Left} and Ω_{Right}). We define $\overleftarrow{\mathbf{R}}^{(i)}$, $i = k, k-1, \ldots, 1$, for rate matrices that correspond to the level decreasing by one unit, towards Level 0 and $\overrightarrow{\mathbf{R}}^{(i)}$, $i = k, k+1, \ldots, s'$, for rate matrices that correspond to the level increasing by one unit, starting at Level k. Note that $\overrightarrow{\mathbf{R}}^{(s'+l)} = \overrightarrow{\mathbf{R}}^{(s')}$, for $l = 1, 2, \ldots$, because of Assumption 1.

We define a left (right) excursion as a time interval that begins when the process moves one step to the left (right) from $L(i)$, $i = k, \ldots, 1$ ($i = k+1, k+2, \ldots$) and ends when the process returns to $L(i)$. The rate matrices $\overleftarrow{\mathbf{R}}^{(i)}$ are lower-triangular square matrices of order $m+1$ (because visiting an upper phase is not possible during a left excursion) and the rate matrices $\overrightarrow{\mathbf{R}}^{(i)}$ are upper-triangular square matrices of order $m+1$ (because visiting a lower phase is not possible during a right excursion):

$$\overleftarrow{\mathbf{R}}^{(i)} = \begin{pmatrix} \overleftarrow{R}^{(i)}_{0,0} & & \\ \vdots & \ddots & \\ \overleftarrow{R}^{(i)}_{m,0} & \cdots & \overleftarrow{R}^{(i)}_{m,m} \end{pmatrix}, \quad i = k, \ldots, 1, \quad \text{and,}$$

$$\overrightarrow{\mathbf{R}}^{(i)} = \begin{pmatrix} \overrightarrow{R}^{(i)}_{0,0} & \cdots & \overrightarrow{R}^{(i)}_{0,m} \\ & \ddots & \vdots \\ & & \overrightarrow{R}^{(i)}_{m,m} \end{pmatrix}, \quad i = k, \ldots, s'.$$

The entries in the rate matrices are:

$\overleftarrow{R}^{(i)}_{j,h}$ = expected time spent in state $(i-1, h)$, $h = j, \ldots, 0$, during a left excursion with the initial state (i, j), expressed in the expected sojourn time in state (i, j), and

$\overrightarrow{R}^{(i)}_{j,h}$ = expected time spent in state $(i+1, h)$, $h = j, \ldots, m$, during a right excursion with the initial state (i, j), expressed in the expected sojourn time in state (i, j).

We replace the forward matrix-geometric recursion in (2) with a backward recursion for Ω_{Left} and a forward recursion for Ω_{Right}:

$$\boldsymbol{\pi}_{i-1} = \boldsymbol{\pi}_i \overleftarrow{\mathbf{R}}^{(i)}, \quad i = k, k-1, \ldots, 1, \tag{3}$$

$$\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i \overrightarrow{\mathbf{R}}^{(i)}, \quad i = k, k+1, \ldots \tag{4}$$

After computing the rate matrices, as explained in Section 5.1, the linear equation set (5) along with the normalization Equation (6), written at Level k, characterizes the steady-state probabilities completely:

$$\boldsymbol{\pi}_k (\overleftarrow{\mathbf{R}}^{(k)}\mathbf{A}_0 + \mathbf{A}_1^{(k)} + \overrightarrow{\mathbf{R}}^{(k)}\mathbf{A}_2^{(k+1)}) = \mathbf{0}, \tag{5}$$

$$\Pr(I_\infty^2 \leqslant k-1) + \Pr(k \leqslant I_\infty^2 < s') + \Pr(I_\infty^2 \geqslant s') = 1, \quad (6)$$

where I_∞^2 is the steady-state random variable for $I(t)$ in Model M_2 and

$$\Pr(I_\infty^2 \leqslant k-1) = \boldsymbol{\pi}_k \left(\sum_{i=1}^{k} \prod_{j=0}^{k-i} \overleftarrow{\mathbf{R}}^{(k-j)} \right) \mathbf{1}, \quad (7)$$

$$\Pr(k \leqslant I_\infty^2 < s') = \boldsymbol{\pi}_k \left(\mathbf{I} + \sum_{i=k}^{s'-2} \prod_{j=k}^{i} \overrightarrow{\mathbf{R}}^{(j)} \right) \mathbf{1}, \quad (8)$$

$$\Pr(I_\infty^2 \geqslant s') = \boldsymbol{\pi}_{s'} \left(\sum_{i=0}^{\infty} \overrightarrow{\mathbf{R}}^{(s')^i} \right) \mathbf{1} = \boldsymbol{\pi}_k \prod_{i=k}^{s'-1} \overrightarrow{\mathbf{R}}^{(i)} (\mathbf{I} - \overrightarrow{\mathbf{R}}^{(s')})^{-1} \mathbf{1}. \quad (9)$$

5.1. Computing the Rate Matrices

We extend Property 3.1 in van Leeuwaarden and Winands (2006) to capture the level-dependency of the rate matrices in Ω_{Left} and Ω_{Right} and restate the property as follows:

PROPERTY 1. *Element $\overrightarrow{R}^{(i)}_{j,h}$, corresponding to a right excursion starting from state (i,j) to state $(i+1,h)$, and element $\overleftarrow{R}^{(i)}_{j,h}$, corresponding to a left excursion starting from state (i,j) to state $(i-1,h)$, can be decomposed as*

$$\overrightarrow{R}^{(i)}_{j,h} = \overrightarrow{q}^{(i)}_{j,h} E(\overrightarrow{X}^{(i)}_h) \frac{[\mathbf{A}_1^{(i)}]_{j,j}}{[\mathbf{A}_1^{(i+1)}]_{h,h}}, \quad i=k,\dots,s';\ j=0,\dots,m;\ h=j,\dots,m, \quad (10)$$

$$\overleftarrow{R}^{(i)}_{j,h} = \overleftarrow{q}^{(i)}_{j,h} E(\overleftarrow{X}^{(i)}_h) \frac{[\mathbf{A}_1^{(i)}]_{j,j}}{[\mathbf{A}_1^{(i-1)}]_{h,h}}, \quad i=k,\dots,1;\ j=0,\dots,m;\ h=0,\dots,j, \quad (11)$$

where $\overrightarrow{q}^{(i)}_{j,h}$ ($\overleftarrow{q}^{(i)}_{j,h}$) is the combined probability of all paths from state (i,j) to state $(i+1,h)$ ($(i-1,h)$), $E(\overrightarrow{X}^{(i)}_h)$ ($E(\overleftarrow{X}^{(i)}_h)$) is the expected number of visits to state $(i+1,h)$ ($(i-1,h)$) given that the excursion visits state $(i+1,h)$ ($(i-1,h)$) at least once, and $[\mathbf{A}_1^{(i)}]_{j,j}/[\mathbf{A}_1^{(i+1)}]_{h,h}$ ($[\mathbf{A}_1^{(i)}]_{j,j}/[\mathbf{A}_1^{(i-1)}]_{h,h}$) is the ratio of the expected time spent in state $(i+1,h)$ ($(i-1,h)$) to the expected time spent in state (i,j).

Property 1 decomposes the task of computing the matrix elements $\overrightarrow{R}^{(i)}_{j,h}$ and $\overleftarrow{R}^{(i)}_{j,h}$ into computation of three terms. Obtaining the last term in (10) and (11) is straightforward. For the computation of the first two elements in (10), we exploit the transition structure of Model M_2 in Ω_{Right}, as shown in Figure 3(b). The structural property that facilitates the efficient computation of the first two terms in (10) is that a right excursion will finish in a finite number of transitions; Model M_2 has no loops in Ω_{Right}, except in Phase m. In contrast to the level-independent QBD processes in van Leeuwaarden and Winands (2006), the entries in $\overrightarrow{\mathbf{R}}^{(i)}$ cannot be computed explicitly for the level-dependent Model M_2, because of the complexity of computing $\overrightarrow{q}^{(i)}_{j,h}$. We illustrate the computation of $\overrightarrow{q}^{(i)}_{j,h}$ in Subsection 5.1.1 and provide the details in EC.2.1–EC.2.2. We outline the computation of the elements in $\overleftarrow{R}^{(i)}_{j,h}$ in Subsection 5.1.2 and provide details in EC.3.1–EC.3.2.

5.1.1. Computing $\overrightarrow{q}^{(i)}_{j,h}$. Let the upper-triangular matrix $\overrightarrow{\mathbf{q}}^{(i)}$ contain elements $\overrightarrow{q}^{(i)}_{j,h}$. We explain how to compute the $\overrightarrow{q}^{(i)}_{j,h}$ probabilities, first, for $h=0,\dots,m-1$ and second, for $h=m$.

To calculate $\overrightarrow{q}^{(i)}_{j,h}$ for $h=0,\dots,m-1$, one needs to add the probabilities of all paths from state (i,j) to state $(i+1,h)$ that are part of a right excursion. All such paths are acyclic. In the special case of $s=1$ and constant service rates where all paths have equal probabilities, van Leeuwaarden and Winands (2006, Theorem 3.1) provide a closed-form solution for $q_{j,h}$ obtained by multiplying the number of paths with the fixed path probability.

The state-dependent service rates in Model M_2, however, result in unequal path probabilities. As an example, Figure 4 shows the five possible acyclic paths from state $(4,0)$ to state $(5,3)$, assuming $k \geqslant 4$. All paths include 4 arrivals and 3 service completions. Denote the probability that the next transition from state $(i,j) \in \Omega_{\text{Right}}$ is an arrival as $\overrightarrow{\phi}_{i,j} = \lambda/(\lambda + b\mu_{i,j})$ and the probability that the next transition is a service completion as $\overrightarrow{\psi}_{i,j} = b\mu_{i,j}/(\lambda + b\mu_{i,j})$. If the service rates are fixed and $s=1$, then the arrival and service completion probabilities are constants $\overrightarrow{\phi} = \lambda/(\lambda+\mu)$ and $\overrightarrow{\psi} = \mu/(\lambda+\mu)$, resulting in $q_{0,3} = 5\,\overrightarrow{\phi}^4\overrightarrow{\psi}^3$. When service rates are state-dependent, however, each path has a different probability, resulting in

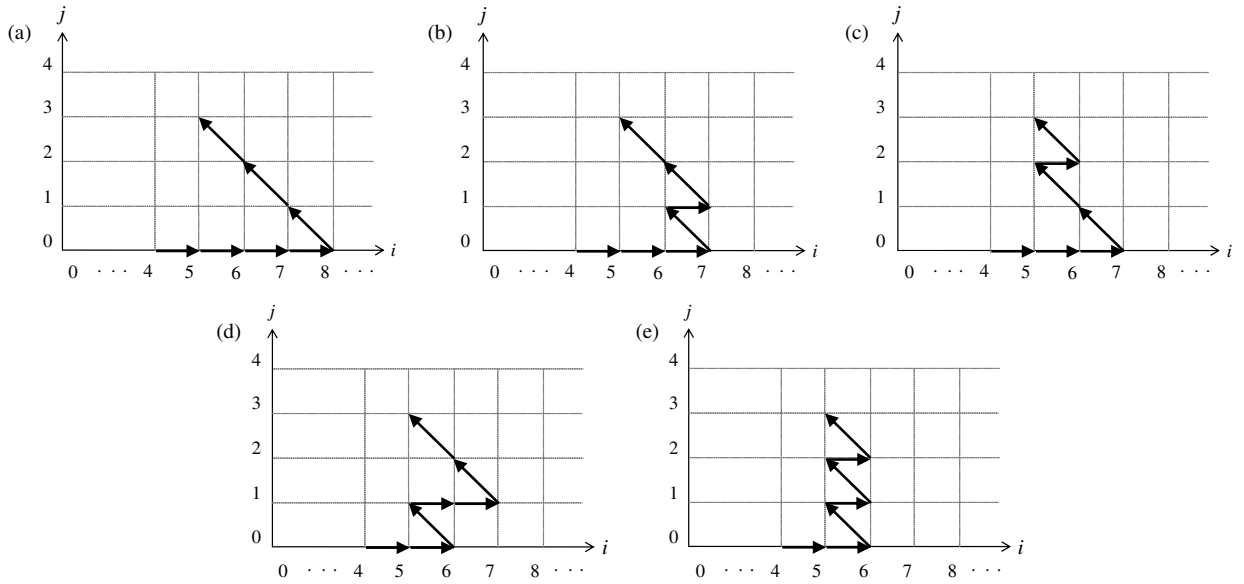
$$\begin{aligned}\overrightarrow{q}^{(4)}_{0,3} = {} & \overrightarrow{\phi}_{4,0}\overrightarrow{\phi}_{5,0}\overrightarrow{\psi}_{6,2} \cdot (\overrightarrow{\phi}_{6,0}\overrightarrow{\phi}_{7,0}\overrightarrow{\psi}_{8,0}\overrightarrow{\psi}_{7,1} + \overrightarrow{\phi}_{6,0}\overrightarrow{\psi}_{7,0}\overrightarrow{\phi}_{6,1}\overrightarrow{\psi}_{7,1} \\ & + \overrightarrow{\phi}_{6,0}\overrightarrow{\psi}_{7,0}\overrightarrow{\psi}_{6,1}\overrightarrow{\phi}_{5,2} + \overrightarrow{\psi}_{6,0}\overrightarrow{\phi}_{5,1}\overrightarrow{\phi}_{6,1}\overrightarrow{\psi}_{7,1} \\ & + \overrightarrow{\psi}_{6,0}\overrightarrow{\phi}_{5,1}\overrightarrow{\psi}_{6,1}\overrightarrow{\phi}_{5,2}).\end{aligned}$$

We calculate $\overrightarrow{q}^{(i)}_{j,h}$ for $h=0,\dots,m-1$ by solving a set of linear equations that resemble the equations used to calculate probabilities of absorption in a Markov chain (for example, see Bertsekas and Tsitsiklis (2008, Section 7.4)). Continuing with the Figure 4 example, $\overrightarrow{q}^{(4)}_{0,3}$ is obtained by solving the following equations:

$$\begin{aligned}
&\overrightarrow{q}^{(4)}_{0,3} = \overrightarrow{\phi}_{4,0}\delta^{5,3}_{5,0}, && \delta^{5,3}_{5,1} = \overrightarrow{\phi}_{5,1}\delta^{5,3}_{6,1},\\
&\delta^{5,3}_{5,0} = \overrightarrow{\phi}_{5,0}\delta^{5,3}_{6,0}, && \delta^{5,3}_{6,1} = \overrightarrow{\phi}_{6,1}\delta^{5,3}_{7,1} + \overrightarrow{\psi}_{6,1}\delta^{5,3}_{5,2},\\
&\delta^{5,3}_{6,0} = \overrightarrow{\phi}_{6,0}\delta^{5,3}_{7,0} + \overrightarrow{\psi}_{6,0}\delta^{5,3}_{5,1}, && \delta^{5,3}_{7,1} = \overrightarrow{\psi}_{7,1}\delta^{5,3}_{6,2},\\
&\delta^{5,3}_{7,0} = \overrightarrow{\phi}_{7,0}\delta^{5,3}_{8,0} + \overrightarrow{\psi}_{7,0}\delta^{5,3}_{6,1}, && \delta^{5,3}_{5,2} = \overrightarrow{\phi}_{5,2}\delta^{5,3}_{6,2},\\
&\delta^{5,3}_{8,0} = \overrightarrow{\psi}_{8,0}\delta^{5,3}_{7,1}, && \delta^{5,3}_{6,2} = \overrightarrow{\psi}_{6,2}\delta^{5,3}_{5,3},
\end{aligned}$$

where the variable $\delta^{5,3}_{a,b}$, $a=5,\dots,8$ and $b=0,\dots,8-i$, is the total probability of all acyclic paths from state (a,b)

Figure 4. Paths from state $(4, 0)$ to state $(5, 3)$, $k \geqslant 4$.



to state $(5, 3)$, for a state (a, b) that is on an acyclic path between state $(4, 0)$ and state $(5, 3)$. The equation with $\delta_{a,b}^{5,3}$ on the left-hand side decomposes the sum of the path probabilities based on the possible transitions out of state (a, b) and their probabilities. The equations can be solved recursively, starting with the initial condition $\delta_{5,3}^{5,3} = 1$.

We detail the calculations for all terms in (10) in EC.2.1–EC.2.2, and we analyze the computational complexity of computing the $\overrightarrow{\mathbf{R}}^{(i)}$ matrices in EC.2.3. The total number of operations needed to compute each of the rate matrices $\overrightarrow{\mathbf{R}}^{(i)}$ is $(m^3 + 4m^2 + 11m + 4)/2$. We argue in EC.2.3 that our algorithm is more efficient and easier to implement than the most efficient published algorithm, in Van Houdt and van Leeuwaarden (2011).

5.1.2. Computing $\overleftarrow{R}_{j,h}^{(i)}$. The transitions in Ω_{Left} do not have the same special structure as in Ω_{Right}, because loops are possible in all phases. Instead, we exploit the following property to compute the first two terms in (11): During a left excursion, one never returns to a phase after leaving it. Furthermore, even though the number of transitions in a left excursion could be infinite, the number of states visited must be finite. We present a set of linear equations to compute $\overleftarrow{q}_{j,h}^{(i)}$ in EC.3.1 and an efficient recursive equation to compute $E(\overleftarrow{X}_h^{(i)})$ in EC.3.2.

5.2. Stability Condition

Intuitively, one expects that the stability condition for Model M_2 should involve the arrival rate λ and the services rates $\mu_{s',j}$ for Level s', above which the service rates stabilize. Theorem 1 provides the precise stability condition.

THEOREM 1. *Model M_2 is stable if and only if $s\mu_{s',m} > \lambda$.*

The proof of Theorem 1 (in EC.4) is based on a general stability condition for level-independent QBD processes. Assumption 3 (that all service rates are positive) is sufficient to ensure that Model M_2 is irreducible. To understand why the stability condition does not involve the service rates in phases below m but only involves the service rate $\mu_{s',m}$ for Phase m, note that for any k-partial busy period that is sufficiently long for at least m service completions to occur, the phase will reach m and remain there until the k-partial busy period ends.

5.3. Performance Measures

Model M_2 is structurally similar to the Erlang C model, particularly for the case when $s' = s$, and we focus on that case in this section. Define the scalars $r_i = \lambda/((i+1)\mu)$, $i = 1, \ldots, s-1$, for the Erlang C model. In Model M_2, the rate matrices $\overleftarrow{\mathbf{R}}^{(i)}$ play the role of the scalars r_{i-1}^{-1}, $i = 1, \ldots, k$, and the rate matrices $\overrightarrow{\mathbf{R}}^{(i)}$ play the role of the scalars r_i, $i = k, \ldots, s-2$. The rate matrix $\overrightarrow{\mathbf{R}}^{(s)}$ plays the role of the Erlang C utilization $r_{s-1} = \lambda/(s\mu)$. Table 2 compares formulas for the steady-state probabilities, delay probabilities, and average delays for the Erlang C and M_2 models.

To show the structural similarities more clearly in Table 2, we decompose the Erlang C formulas into the same three regions as used for Model M_2 in (7)–(9). Erlang C formulas often involve expressing the probabilities of all states in terms of π_0, but one can equivalently express the probabilities of all states in terms of π_k, where $k > 0$, as in Table 2. This approach also provides numerical advantages for Erlang C calculations, as discussed in Ingolfsson and Tang (2012).

Table 2. Structural similarities between the Erlang C and model M_2 when $s' = s$.

Model	Steady-state probabilities and performance measures
Erlang C	$\pi_{i-1} = \pi_i r_{i-1}^{-1}, \quad i = k, \ldots, 1$ $\pi_{i+1} = \pi_i r_i, \quad i = k, \ldots, s-1$ $\pi_{i+1} = \pi_i r_{s-1}, \quad i = s, s+1, \ldots$ $\pi_k = \left(1 + \sum_{i=1}^{k} \prod_{j=0}^{k-i} r_{k-j+1}^{-1} + \sum_{i=k}^{s-2} \prod_{j=k}^{i} r_j + \prod_{i=k}^{s-1} r_i (1 - r_{s-1})^{-1}\right)^{-1}$ $\Pr(\text{Delay}) = \pi_k \prod_{i=k}^{s-1} r_i (1 - r_{s-1})^{-1}$ $W_q = \frac{1}{\lambda} \pi_k \prod_{i=k}^{s-1} r_i r_{s-1} (1 - r_{s-1})^{-2}$
M_2	$\boldsymbol{\pi}_{i-1} = \boldsymbol{\pi}_i \overleftarrow{\mathbf{R}}^{(i)}, \quad i = k, \ldots, 1$ $\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i \overrightarrow{\mathbf{R}}^{(i)}, \quad i = k, \ldots, s-1$ $\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i \overrightarrow{\mathbf{R}}^{(s)}, \quad i = s, s+1, \ldots$ $\boldsymbol{\pi}_k = \left[\left(\mathbf{I} + \sum_{i=1}^{k} \prod_{j=0}^{k-i} \overleftarrow{\mathbf{R}}^{(k-j)} + \sum_{i=k}^{s-2} \prod_{j=k}^{i} \overrightarrow{\mathbf{R}}^{(j)} + \prod_{i=k}^{s-1} \overrightarrow{\mathbf{R}}^{(i)} (\mathbf{I} - \overrightarrow{\mathbf{R}}^{(s)})^{-1}\right)\mathbf{1}\right]^{-1}$ $\Pr(I_\infty^2 \geqslant s) = \Pr(\text{Delay}) = \boldsymbol{\pi}_k \prod_{i=k}^{s-1} \overrightarrow{\mathbf{R}}^{(i)} (\mathbf{I} - \overrightarrow{\mathbf{R}}^{(s)})^{-1} \mathbf{1}$ $W_q = \frac{1}{\lambda} \boldsymbol{\pi}_k \prod_{i=k}^{s} \overrightarrow{\mathbf{R}}^{(i)} (\mathbf{I} - \overrightarrow{\mathbf{R}}^{(s)})^{-2} \mathbf{1}$

To obtain the expected queue delay, W_q, expression for Model M_2 in Table 2, we use Little's Law after deriving the average queue length, L_q, as below:

$$L_q = \sum_{i=s}^{\infty} (i-s)\boldsymbol{\pi}_i \mathbf{1} = \boldsymbol{\pi}_s \overrightarrow{\mathbf{R}}^{(s)} (\mathbf{I} - \overrightarrow{\mathbf{R}}^{(s)})^{-2} \mathbf{1}$$
$$= \boldsymbol{\pi}_k \prod_{i=k}^{s} \overrightarrow{\mathbf{R}}^{(i)} (\mathbf{I} - \overrightarrow{\mathbf{R}}^{(s)})^{-2} \mathbf{1}.$$

The virtual waiting time for Model M_2 corresponds to the time to absorption in a modified version of the model, where all arrival transitions are removed, all states with one or more free servers are aggregated into a single absorbing state, and the probability distribution for the initial state is the steady-state distribution for Model M_2, conditional on all servers being busy. The details of this standard approach are discussed, for example, in Ramaswami and Lucantoni (1985). We provide details of computing the virtual waiting time distribution for Model M_2 in EC.5.

6. Effects of Ignoring Load and Overwork

In this section we illustrate the magnitude of the errors that can result from using the fixed-service-rate Erlang C model, instead of the state-dependent Model M_2, to predict the performance of a system with load and overwork effects. Model M_2 is general in that it can accommodate any functional dependence of service rates on load and overwork as long as Assumptions 1–5 are satisfied. In this and the next section, however, we focus on simplified situations to develop insights and for ease of exposition. In this section we report on experiments in which the service rate varies with load only, with overwork only, and jointly with load and overwork. Our base case is a 35-server system in which load and overwork begin to impact the service rates when all servers are busy ($k = s = 35$). Where relevant, we set the rate at which overwork reduction events occur, $\gamma_{i,j}$, equal to 1.

6.1. Experiment with Load Effect

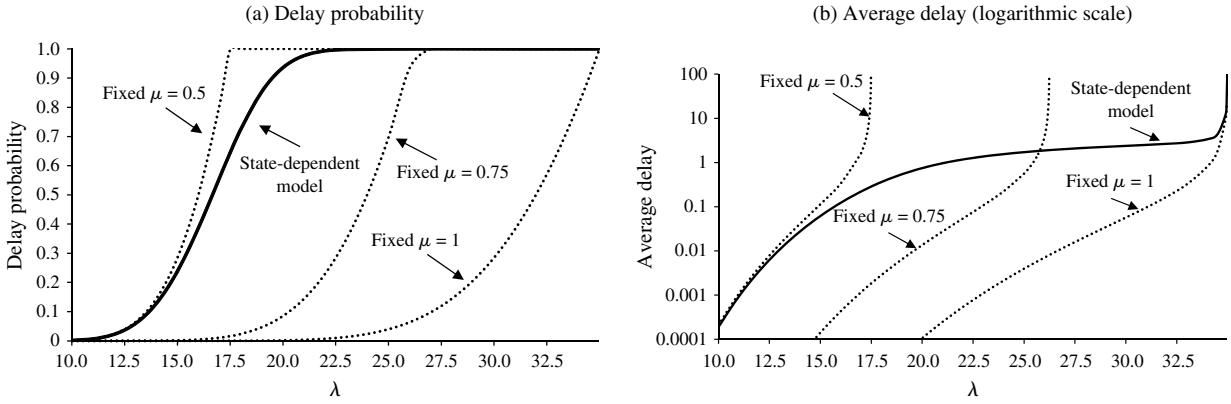
In the first experiment we let the service rate depend only on load:

$$\mu_{i,j} = \begin{cases} \mu_{\min} = 0.5 & i \leqslant 34, \\ \mu_{\min} + (i-35) \times 0.005 & 35 \leqslant i \leqslant 134, \\ \mu_{\max} = 1 & i \geqslant 135. \end{cases} \quad (12)$$

The servers operate at "minimum speed" when fewer than $k = 35$ servers are busy—no queue and at least one idle server. Servers start speeding up with rate 0.005 with each additional user in system for $i \geqslant 35$. The service rate stabilizes at the "full speed" rate of 1 when the queue length reaches 100 ($s' = 135$). The system is stable if $\lambda < 35 \times \mu_{s',m} = 35 \times \mu_{\max} = 35$. Staats and Gino (2012) document a real-life example of service rates that depend on load: Bank agents speed up as more loan applications are assigned to them.

One alternative to M_2 for this system is to use the Erlang C model with a well-chosen "representative" service rate. Three simple representative service rates are $\mu_{\min} = 0.5$, $\mu_{\max} = 1$, and their average ($\bar{\mu}_{SA} = 0.75$). The fixed-rate models with these service rates have stability limits of $\lambda <$ 17.5, 26.2, and 35. Figure 5 compares the delay probability

Figure 5. Exp. 6.1: Performance measures of state-dependent vs. fixed-rate models.



and the average delay (logarithmic scale) for Model M_2 and for the three fixed-rate models.

The $\mu_{\min}$ model overestimates the delay probability while the $\bar{\mu}_{\mathrm{SA}}$ and $\mu_{\max}$ models underestimate it. None of the fixed-rate models approximate the delay probability well for all arrival rates. The fixed-rate models perform even worse in approximating the average delay, with the $\mu_{\min}$ and $\bar{\mu}_{\mathrm{SA}}$ models unable to provide approximations for arrival rates beyond their stability limit.

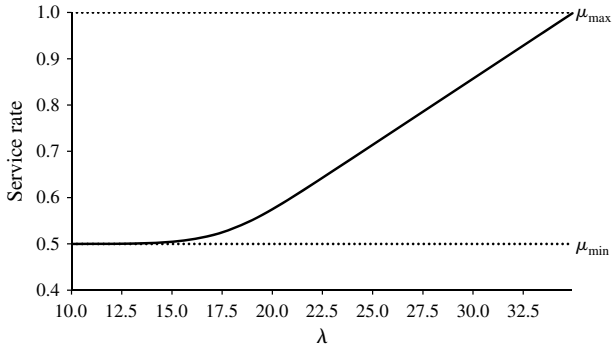
Conceivably, one could obtain a more accurate approximation by using a weighted average $\bar{\mu}$ of service rates $\mu_{i,j}$ as the input to the Erlang C model or by using a weighted average of the outputs from the Erlang C models that correspond to different service rates $\mu_{i,j}$. Let $C(\lambda/\mu, s)$ and $D(\lambda/\mu, s)$ be the Erlang C probability of delay and average delay, respectively. Averaging the input means using $\bar{\mu} = \sum_{i,j} w_{i,j}\mu_{i,j}$, for some set of weights $w_{i,j}$, as the input to the Erlang C model. Averaging the output means estimating the probability of delay and the average delay as $\sum_{i,j} w_{i,j} C(\lambda/\mu_{i,j}, s)$ and $\sum_{i,j} w_{i,j} D(\lambda/\mu_{i,j}, s)$, respectively.

It is not clear how one should choose the weights. We expect, however, that using the state probabilities, obtained by solving Model M_2, as weights ($w_{i,j} = \pi_{i,j}$) should result in greater accuracy than any other set of weights. We follow this conservative approach in assessing the accuracy of both input-averaging and output-averaging approximations.

Figure 6 shows that the weighted average service rate $\bar{\mu}$ shifts from $\mu_{\min} = 0.5$ to $\mu_{\max} = 1$ as the arrival rate approaches the stability limit of 35, because the state probabilities $\pi_{i,j}$ shift towards the full-speed region. We plot the input-averaging and output-averaging approximations in Figure 7. Although input-averaging and output-averaging improve accuracy, at least for the delay probability, considerable and systematic errors remain, especially in estimating the average delay. Note that one still needs to solve Model M_2 to compute the input-averaging and output-averaging approximations.

The output-averaging approximation results in higher delay probability and higher average delay than the input-averaging approximation. This is not a coincidence, but a consequence of Jensen's inequality and convexity properties of $C(\cdot)$ and $D(\cdot)$ with respect to μ (see EC.6).

Figure 6. Exp. 6.1: Weighted average service rate.



6.2. Experiment with Overwork Effect

In the second experiment we let the service rate depend only on overwork:

$$\mu_{i,j} = \begin{cases} \mu_{\max} = 1 & j = 0, \\ \mu_{\max} - j \times 0.05 & 1 \leqslant j \leqslant 9, \\ \mu_{\min} = 0.5 & j \geqslant 10. \end{cases} \tag{13}$$

This system operates at full speed when overwork is zero. During high-load periods ($i \geqslant k = 35$), the service rate is reduced by 0.05 with every service completion, until it stabilizes at 0.5. During the low-load period ($i < 35$), the service rate increases by 0.05 with every overwork reduction. We set $\gamma_{i,j} = 1$, for $i < 35$. A real-life example of service rate being dependent on overwork is call center agent service times lengthening with the number of calls answered since the last break of at least an hour, as reported in Gans et al. (2010).

In Figure 8 we compare the delay probability and average delay of M_2 (on a logarithmic scale) to fixed-rate Erlang C approximations with $\mu_{\min} = 0.5$, $\bar{\mu}_{\mathrm{SA}} = 0.75$, and $\mu_{\max} = 1$. The $\mu_{\min}$ approximations could be useful when load is close to the stability limit, but none of the fixed-rate

Figure 7. Exp. 6.1: Input and output-averaging approximations.

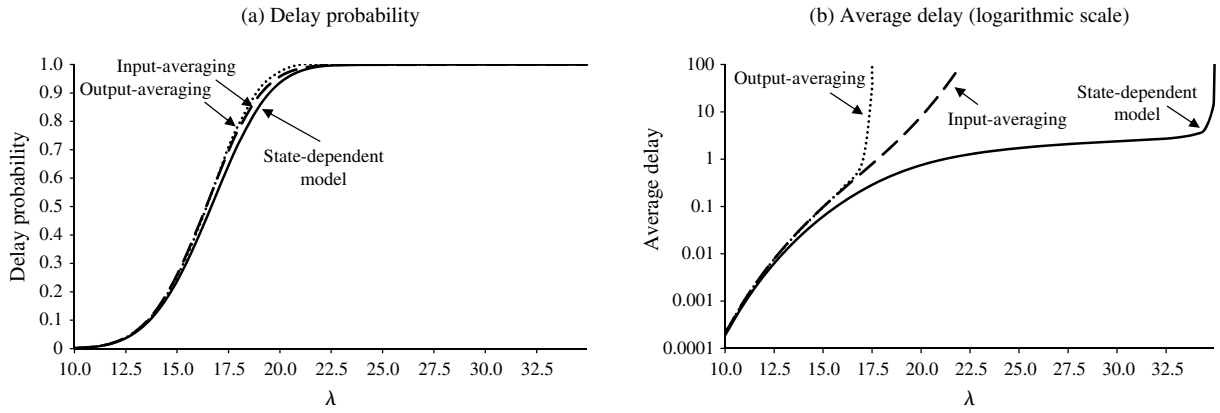
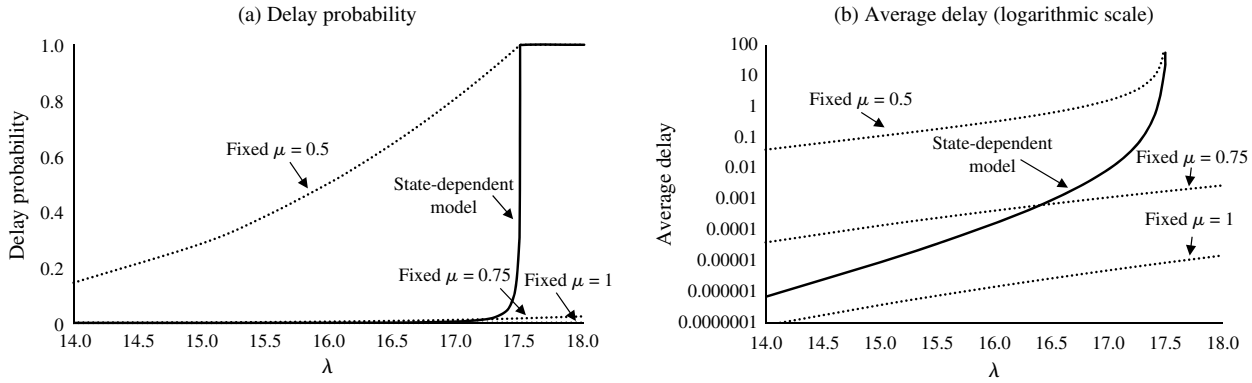


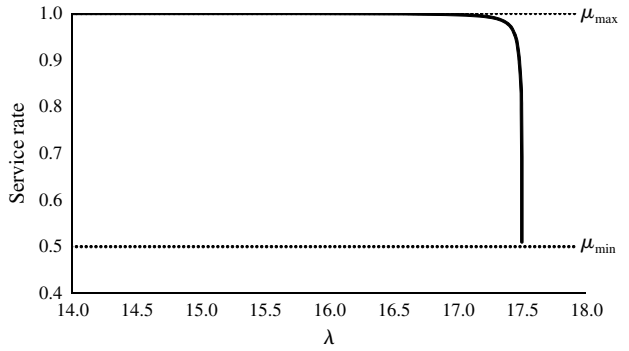
Figure 8. Exp. 6.2: Performance measures of state-dependent vs. fixed-rate models.



models captures the shift in system behavior as the load increases towards the stability limit.

Figure 9 shows how the weighted average service rate shifts from $\mu_{\max} = 1$ to $\mu_{\min} = 0.5$ as λ increases, causing the steady-state probabilities to shift towards the high-overwork region. We plot input-averaging and output-averaging approximations in Figures 10(a)–10(b). These approximations perform better than the Erlang C approximations in which the fixed service rate is independent of λ, but significant and systematic errors remain. In this experiment, output-averaging performs better, but in the speedup experiment in Section 6.1, input-averaging performed better.

Figure 9. Exp. 6.2: Weighted average service rate.



6.3. Experiment with Load and Overwork Effects

In the third experiment, we let the service rate depend on both load and overwork, as follows.

$$\mu_{i,j} = \begin{cases} \mu_1 = 0.9 & \text{Region 1: } i \leqslant 34;\ j = 0, \\ \mu_2 = 1 & \text{Region 2: } i \geqslant 35;\ j = 0, \\ \mu_3 = 0.75 & \text{Region 3: } i \geqslant 35;\ j > 0, \\ \mu_4 = 0.85 & \text{Region 4: } i \leqslant 34;\ j > 0. \end{cases} \tag{14}$$

Figure 11 illustrates the four service-rate regions. In words, the service rate is at full speed when all servers are busy but no service completions have occurred since the 35th server became busy. The servers slow down to 90% of full speed when fewer than 35 servers are busy and they slow down to 75% of full speed in the overwork accumulation region, which begins with the first service completion after the 35th server became busy and ends when a service completion leaves 34 busy servers. When the overwork accumulation region ends, the overwork attenuates gradually and service rate increases to 85% of full speed. We set $\gamma_{i,j} = 1$, for $i < 35$. The system is stable if $\lambda < 35 \times \mu_3 = 26.25$. KC and

Figure 10. Exp. 6.2: Input and output-averaging approximations.

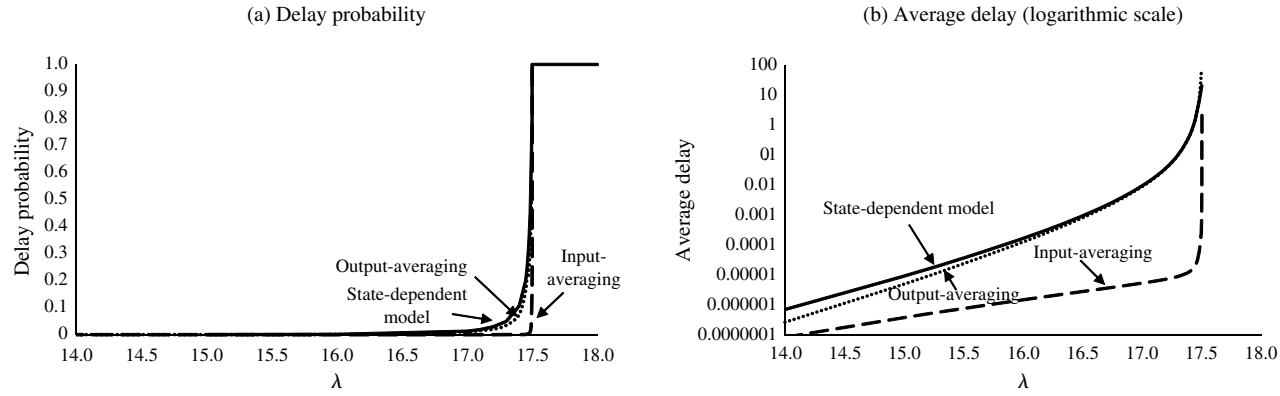
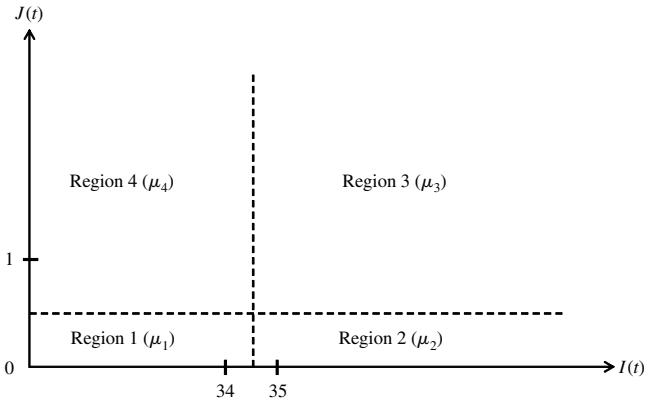


Figure 11. Exp. 6.3: Service-rate regions where rates vary with load and overwork.



Terwiesch (2009) document speedup in response to load and slowdown in response to overwork in a real-life setting: staff who transport patients in hospitals.

In this example high load initially causes speedup and eventually overwork causes slowdown. Speedup and slowdown, in turn, affect our overwork measure. Starting in the low-load Region 1 ($\mu_1 = 0.9$), the system eventually moves to the high-load Region 2 ($\mu_2 = 1$), where the faster service rate causes overwork to increase at a faster rate, thus speeding the entry into the overwork-accumulation Region 3 ($\mu_3 = 0.75$). Once in that region, the slowdown caused by overwork delays entry into the low-load Region 4, where the overwork begins to decrease. In general, we expect speedup and slowdown to affect our overwork measure, with speedup during a high-load period increasing overwork as service completions occur at a higher rate, and slowdown during a high-load period decreasing the rate at which overwork increases.

Figure 12 compares the delay probability and the average delay (logarithmic scale) for Model M_2 with delay probabilities and average delays of four Erlang C models with rates fixed at μ_1, μ_2, μ_3, and μ_4. The fixed-rate models with μ_1, μ_2, and μ_4 underestimate the delay probability and average delay, while the fixed-rate model with μ_3 overestimates the two measures. All four Erlang C models perform poorly for low and moderate arrival rates. The Erlang C model with service rate μ_3 (the one corresponding to the "overwork accumulation region") is accurate for arrival rates close to the stability limit. When the arrival rate approaches the stability limit, the Model M_2 probability for the overwork accumulation region approaches 1 and the Model M_2 average delay diverges, as it does in the Erlang C model. None of the four Erlang C models provide a good approximation over the entire arrival rate range.

Figure 12. Exp. 6.3: Performance measures of state-dependent vs. fixed-rate models.

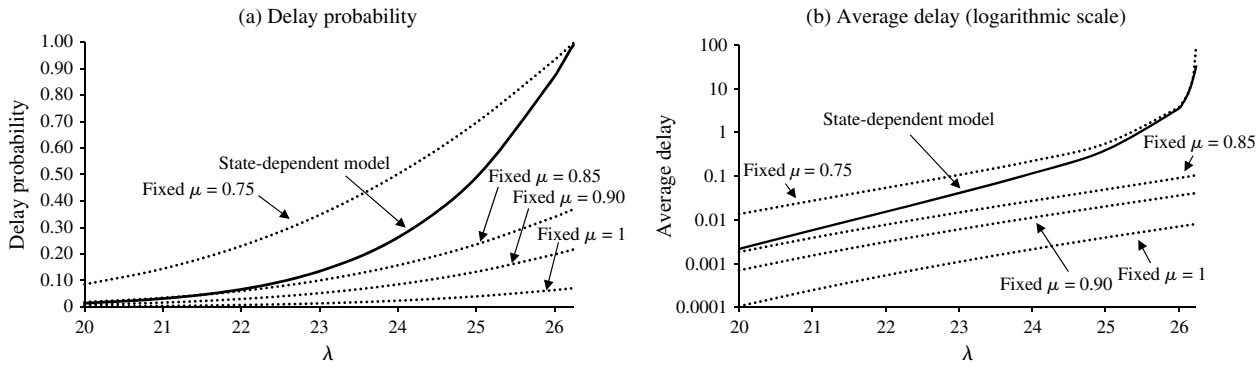


Figure 13. Exp. 6.3: Impact of arrival rate on state probabilities and average service rate.

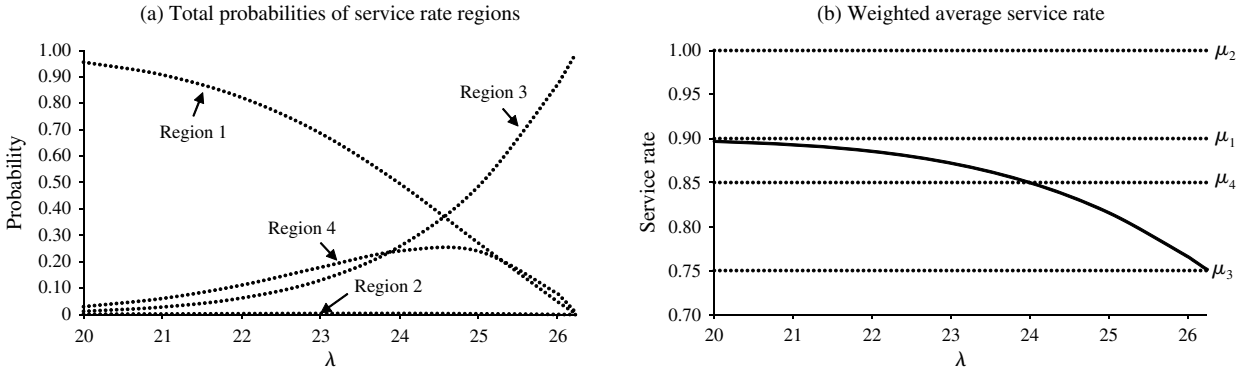


Figure 13(a) shows that the probability mass shifts from the low-load Region 1 to the overwork-accumulation Region 3 as the arrival rate increases from 20 to 26.25 (the stability limit), which results in the weighted average service rate $\bar{\mu}$ shifting from $\mu_1 = 0.9$ to $\mu_3 = 0.75$, as shown in Figure 13(b). In this example, we see that the net effect when the arrival rate increases is that the weighted average service rate decreases, that is, a slowdown effect. In other experiments (see EC.7), we have found that by varying $\mu_1, \mu_2, \mu_3, \mu_4$, and the boundary lines that separate Regions 1–4, we can find situations where the weighted average service rate increases with the arrival rate (net speedup) and where the weighted average service rate first increases and then decreases. Thus, our model encompasses three different phenomena that have been observed empirically for the net effect of load on service rates: increasing (KC and Terwiesch 2009), decreasing (Jaeker and Tucker 2012), and inverted U-shape (Tan and Netessine 2014).

Input-averaging and output-averaging provide better approximations (see Figure 14) for this experiment than for the pure speedup and pure slowdown experiments. Still, considerable and systematic errors remain.

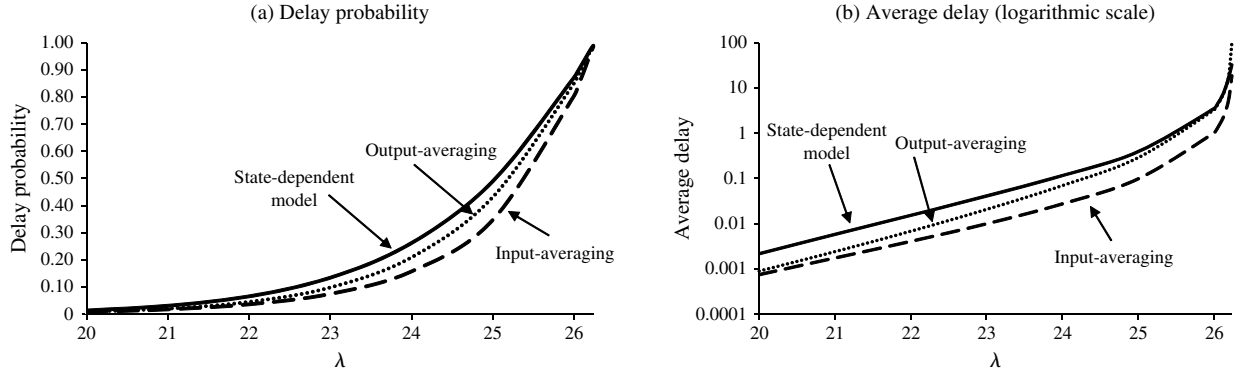
6.4. Using Load as a Proxy for Overwork

Load and overwork are likely to have a positive association. When the queue is long, overwork is likely to be high. Therefore, it is natural to wonder whether a model with load as the only state variable could capture the dynamics of Model M_2. We define Model M_3 as a birth-death process with state variable $I(t)$ and the corresponding steady-state random variable I^3_∞, with a constant birth rate λ and load-dependent death rates $\min(i, s)\mu_i$, where μ_i is the service rate per server when $I(t) = i$. Mirroring Assumption 1, we assume that $\mu_{s''+l} = \mu_{s''}$ for $l = 1, 2, \ldots$, where s'' can differ from s'.

We ask whether it is possible to find load-dependent service rates, μ_i, such that Model M_3 provides a good approximation to performance measures from Model M_2, specifically the probability of delay, $C^{(2)} = \Pr(I^2_\infty \geqslant s)$, and the average delay, $D^{(2)} = \mathrm{E}[(I^2_\infty - s)^+]/\lambda$.

If all parameters, including the arrival rate, are fixed, then the answer is yes. For any performance measure that can be expressed as the expected value of a function of I^2_∞, we can choose the service rates μ_i so that I^2_∞ and I^3_∞ are equal in distribution. We can achieve this by setting the Model M_3 service rates μ_i equal to the weighted average service rate for Level $i > 0$ in Model M_2, that is, $\mu_i = \sum_{j=0}^{m} \mu_{ij}\pi_{ij} / \sum_{j=0}^{m} \pi_{ij}$. Note, however, that to compute μ_i, one needs to solve Model M_2. Also, if λ changes, then the Model M_2 steady-state distribution $\{\pi_{ij}\}$ changes, and therefore the service rates μ_i for Model M_3 change, even if the service rates μ_{ij} in Model M_2 remain constant.

Figure 14. Exp. 6.3: Input and output-averaging approximations.



With this in mind, we refine our question to "can one find a single set of load-dependent service rates μ_i such that Model M_3 gives a good approximation to the probability of delay and the average delay, for multiple arrival rates?" Suppose that we use Model M_2 with arrival rates λ_u to obtain performance measures $C_u^{(2)}$ and $D_u^{(2)}$, for $u = 1, \ldots, U$. For a given set of service rates, μ_i, we can use Model M_3 to compute approximations, $C_u^{(3)}$ and $D_u^{(3)}$. We formulate the following optimization problem:

$$\text{minimize} \quad \alpha \sum_{u=1}^{U} (C_u^{(2)} - C_u^{(3)})^2 + (1-\alpha) \sum_{u=1}^{U} (\ln D_u^{(2)} - \ln D_u^{(3)})^2$$

$$\text{subject to} \quad \mu_i \geqslant \epsilon_1 > 0, \quad i = 1, \ldots, s'' - 1$$

$$\mu_{s''} \geqslant \max\{\lambda_1, \ldots, \lambda_U\}/s + \epsilon_2, \quad \text{where } \epsilon_2 > 0.$$

The objective function minimizes the sum of squared deviations between performance measures from the two models, M_2 and M_3. The average delay diverges as the arrival rate approaches the stability limit and therefore, we use a log transformation to prevent the average delay for the highest arrival rate from dominating. The first constraint ensures that Model M_3 is irreducible, and the second constraint ensures stability. We provide more information about the formulation and numerical solution of this nonlinear program in Appendix EC.8.

In the Section 6.1 scenario, Model M_2 reduces to Model M_3. We solved the optimization problem for the scenarios from Sections 6.2 and 6.3, first using $\alpha = 1$ (optimize to match the probability of delay values, $C_u^{(2)}$) and, second, using $\alpha = 0$ (optimize to match the average delay values, $D_u^{(2)}$). The results are shown in Figures 15–16. In the pure overwork scenario (Section 6.2), the optimized Model M_3 provides a fairly good approximation to both the probability of delay (Figure 15(a)) and the average delay (Figure 15(b)), by specifying $s'' = 200$ load-dependent service rates (Figure 15(c)).

In the load and overwork scenario (Section 6.3), we increase the number of load-dependent service rates gradually to $s'' = 600$ to achieve the best approximations (EC.8 provides details). Optimizing Model M_3 to match the probability of delay leads to poor fit with respect to the average delay, and vice versa (Figure 16). We performed additional optimization runs where we varied the weight α but we were not able to find a set of service rates that allowed Model M_3 to approximate both the probability of delay and the average delay values simultaneously.

In addition to the inability of Model M_3 to approximate the Model M_2 performance measures over a range of arrival rates, for the load and overwork scenario, we note the contrast between the service rates μ_{ij} specified in Section 6.3 and the service rates μ_i plotted in Figure 16(c). On the one hand, we have four distinct service rates in Model M_2 that vary in magnitude from 0.75 to 1, where each service rate has a physical interpretation as the rate of service completions in a particular subset of the Model M_2 state space (Figure 11). On the other hand, we have 600 load-dependent service rates in Model M_3 that vary in a non-monotonic fashion with load and that vary in magnitude from close to zero to almost 1.4. The variation in magnitudes and the large jumps seen in Figure 16(c) (and in Figure 15(c)) make a physical interpretation of these service rates implausible. For example, it is difficult to believe that servers will slow down drastically as the number of users is just below 600, only to jump back to the speed required for stability of the system when the number of users reaches 600 (see service rates optimized by D in Figure 16(c)).

Figure 15. Exp. 6.2: Optimizing the one-dimensional Model M_3.

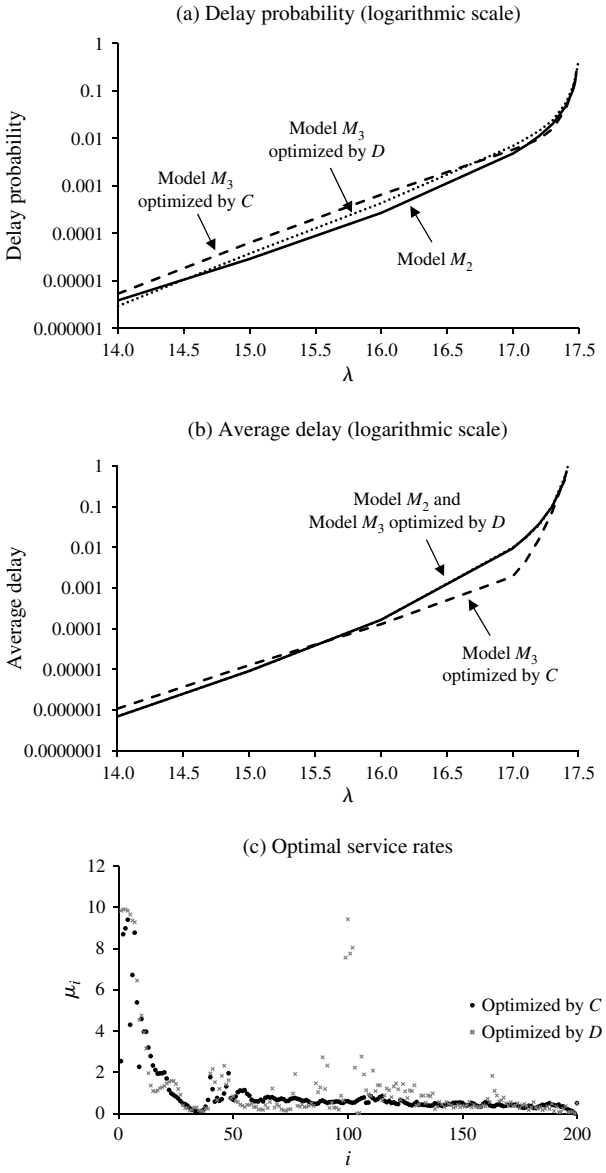
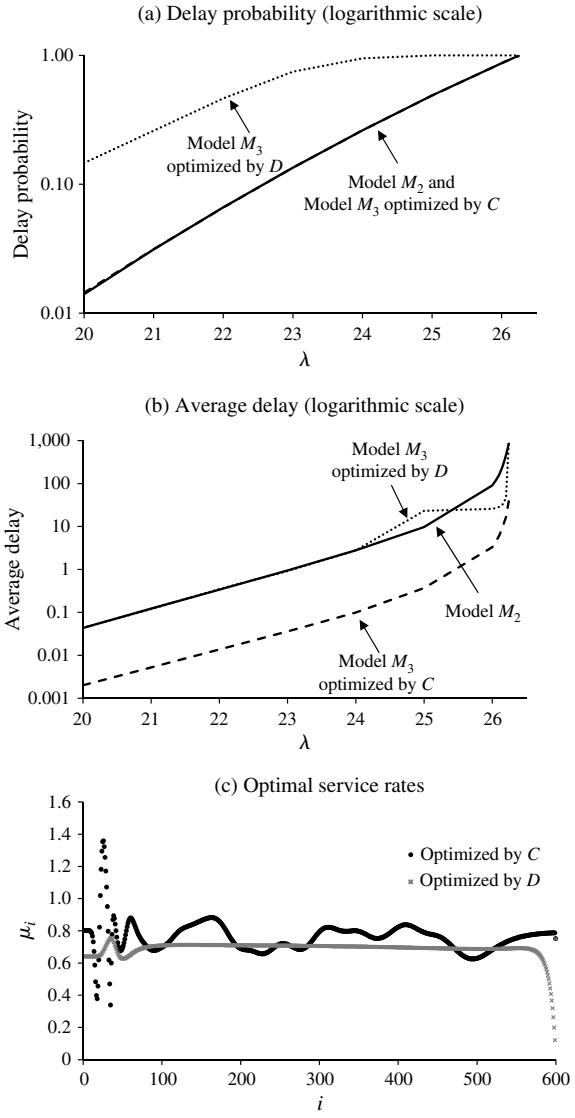


Figure 16. Exp. 6.3: Optimizing the one-dimensional Model M_3.



7. Consequences of Using the "Wrong" Model

In Section 6 we studied the errors that can occur when one uses the Erlang C model to evaluate performance of a system where the service rates vary with load and overwork. In this section we investigate the possible consequences of using the Erlang C model (the "wrong" model, if Model M_2 represents reality) on an ongoing basis to set staffing levels in a service system with state-dependent service rates.

The process of staffing a service system, like other business processes, requires mechanisms for monitoring and control (Powell et al. 2001) so that the process can adapt to changing conditions. Buffa et al. (1976, Figure 5) and Lin et al. (2000) describe monitoring and control frameworks for telephone operator staffing. Both frameworks involve a form of feedback control and ideally, ongoing monitoring and feedback of system performance would lead to self-correcting behavior, even if one uses an incorrect model. In our experiments we find that this desirable outcome does occur in some situations. However, it is not obvious how to design effective feedback control systems for business processes (Powell et al. 2001), and we find that using an incorrect model can result in a variety of undesirable system trajectories.

A primary purpose of feedback control is to adapt to changes in input parameters, but to focus on the consequences of using the wrong model, we consider a system whose arrival and state-dependent service rates are time invariant. We use the following procedure to imitate staffing decisions in such a system:

1. Set arrival rate (λ), state-dependent service rates ($\mu_{i,j}$), initial staffing ($s = s_0$), initial period counter ($n = 1$), target service level (φ), and monitoring period length (T).

2. Simulate the system for Period n and estimate the arrival rate ($\hat{\lambda}_n$) and the service rate ($\hat{\mu}_n$),

$$\hat{\lambda}_n = \frac{A_n}{T}, \quad 1/\hat{\mu}_n = \frac{\sum_{i=1}^{S_n} X_i}{S_n}, \tag{15}$$

where A_n and S_n are the number of arrivals and service completions in Period n, and X_i is the service time of user i whose service finished in Period n.

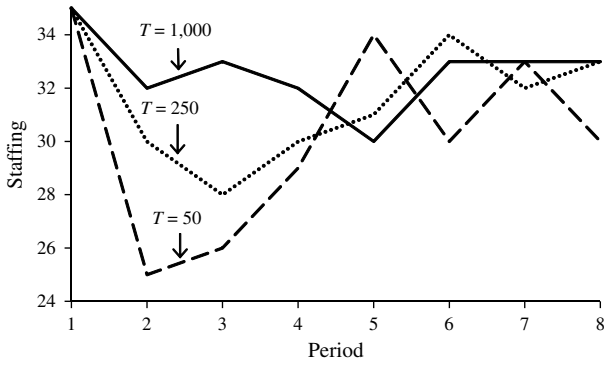
3. Use $\hat{\lambda}_n$ and $\hat{\mu}_n$ in the Erlang C model to find the minimum required staffing for the next period, s_{n+1}, to satisfy φ.

4. Set $n \leftarrow n + 1$ and return to Step 2.

The simulation model simulates the state-dependent system, which we assume represents reality. When a server begins serving a user, while in state (i, j), we simulate the service time as exponentially distributed with rate $\mu_{i,j}$. Whenever the system transitions from state (i, j) to state (i', j'), we update the remaining service times of the users in service by generating new exponentially distributed random variates with rate $\mu_{i',j'}$. If the number of servers increases from one period to the next one, then the service of the users in the queue at the end of the previous period, if any, is immediately initiated by the newly added servers. If the number of servers decreases, then the departure of a server, if busy, is postponed until the current service is completed.

In the experiments of this section, we set $\lambda = 20$ per hour and $\varphi = 0.90$ and define the service level as the probability that the virtual wait is less than or equal to 20 minutes. We use the service-rate function (14), but we vary the low-load service rates μ_1 and μ_4. We begin by illustrating the impact of noisy estimation of λ and μ. Figure 17 shows simulated staffing for eight periods, when $\mu_1 = 0.75$ and $\mu_4 = 0.7$ per hour, under three monitoring periods: short ($T = 50$ hours with 1,000 expected arrivals), medium ($T =$ 250 hours with 5,000 expected arrivals), and long ($T =$ 1,000 hours with 20,000 expected arrivals). The optimal staffing for this system, obtained from the state-dependent

Figure 17. The impact of noisy parameter estimation on staffing.



Model M_2, is $s = 33$. The staffing levels obtained from the Erlang C model converge to the optimal value after six periods when T is long. For the shorter monitoring periods, the staffing levels take longer to reach the optimal value and the staffing is not guaranteed to remain at the optimal value because of the estimation noise.

In the remainder of this section we focus on factors other than parameter estimation noise. Therefore, we assume that the monitoring period is long enough that parameter estimation errors are negligible and that the system reaches steady state, if the system is stable, which means that we can replace the simulation model with the numerical solution of Model M_2. In what follows, we vary the low-load service rates μ_1 and μ_4, as shown in Table 3, to illustrate three main staffing patterns that we have observed: (1) convergence (to a value that is too high, optimal, or too low), (2) oscillation, and (3) instability.

Convergence: As (μ_1, μ_4) varies from $(0.55, 0.50)$ to $(0.75, 0.7)$ to $(0.85, 0.82)$ per hour, the staffing converges in all cases, but to a value that changes from too high, to optimal, to too low, as illustrated in Figures 18–20. The rates of $\mu_1 = 0.55$ and $\mu_4 = 0.50$ per hour are so much lower than the full-speed service rate (μ_2) of one per hour that the Erlang C model cannot capture the speedup properly and converges to a staffing level (39 servers) that is far above the optimal value of 36 (Figure 18). When μ_1 and μ_4 increase to 0.75 and 0.7 per hour then the Erlang C model approximates the system performance sufficiently well that the staffing converges to the optimal value (Figure 19). With μ_1 and μ_4 above $\mu_3 = 0.75$, at 0.85 and 0.82, respectively, the Erlang C model fails to properly account for

Figure 18. Convergence to overstaffing.

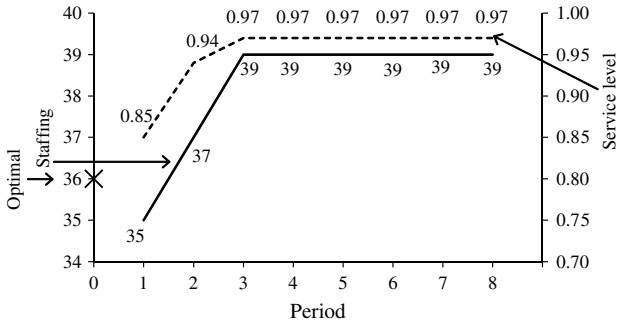


Figure 19. Convergence to optimal staffing.

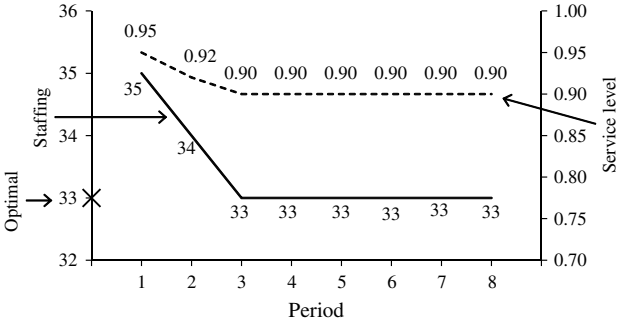
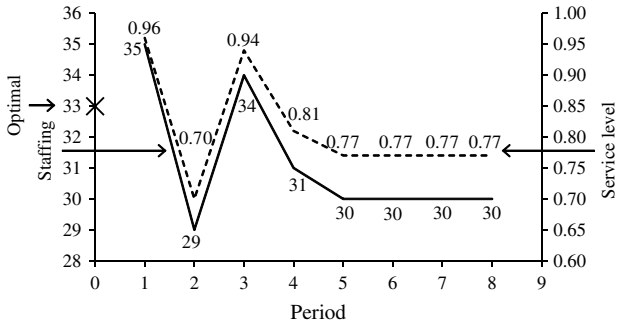


Figure 20. Convergence to understaffing.



overwork-induced slowdown, which results in convergence to understaffing (30 versus 33, as shown in Figure 20).

Interestingly, we see from Figure 18 that the procedure that we have described results in an increase in staffing even when the service level (as computed using Model M_2) is above target. Similarly, in Figure 20, we see one period of a decrease in staffing even though the service level is

Table 3. Experiment parameters.

Low-load service rates (per hour)		Behavior	Figure
$\mu_1 = 0.55$	$\mu_4 = 0.50$	Convergence to overstaffing	18
$\mu_1 = 0.75$	$\mu_4 = 0.70$	Convergence to optimal staffing	19
$\mu_1 = 0.85$	$\mu_4 = 0.82$	Convergence to understaffing	20
$\mu_1 = 0.90$	$\mu_4 = 0.90$	Oscillation	21
$\mu_1 = 0.95$	$\mu_4 = 0.93$	Instability	22

Figure 21. Staffing oscillation.

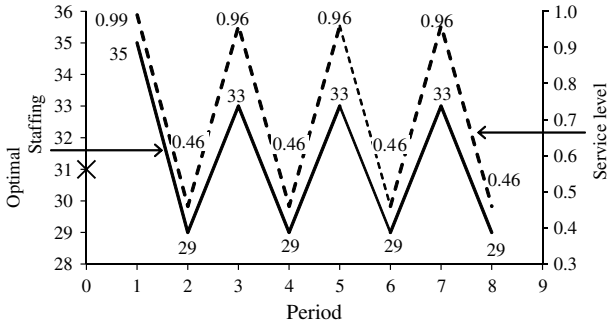
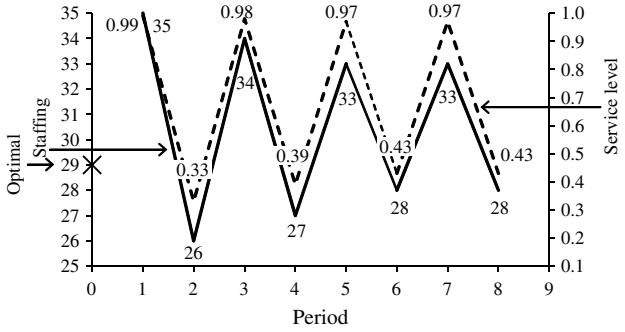


Figure 22. Unstable staffing.



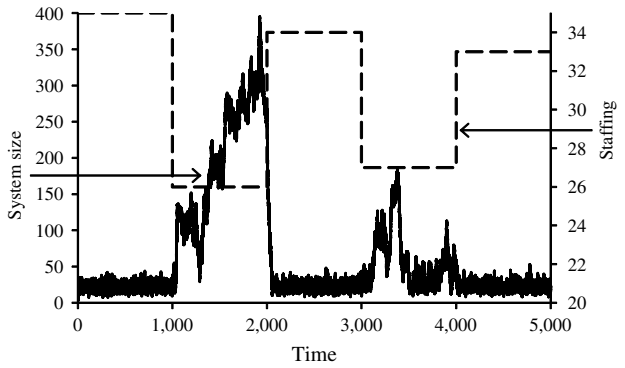
below the target. These counterintuitive decisions occur because in our procedure, changes in staffing are determined using the wrong Erlang C model. One can envision an alternative "model-free" procedure that determines changes in staffing based only on an empirically estimated service level, $\widehat{SL}$. The simplest such procedure might be to increase the staffing by one if $\widehat{SL} < \varphi$ and decrease the staffing by one if $\widehat{SL} > \varphi$. This procedure would change staffing at the beginning of every monitoring period, assuming that it is unlikely that the empirically estimated service level is exactly equal to its target.

Oscillation: When we increase the low-load service rates to $\mu_1 = \mu_4 = 0.90$, then the staffing levels no longer converge, but oscillate (Figure 21), between 29 (with service level = 0.46) and 33 (with service level = 0.96), while the optimal staffing level is 31. The long-term average service level in this oscillating system is 0.71 (the average of 0.46 and 0.96), which is below the target of 0.9. In addition, in a real system, one expects that the constant staffing changes would be costly and the constant changes in service level might impact user retention.

The oscillation occurs because with 29 servers, the system is very likely to remain in the overwork accumulation region, and with 33 servers the system is very likely to be in the low-load region. In this situation, the Erlang C model is unreliable in extrapolating service levels from the low-load region to the overwork accumulation region and vice versa. Specifically, with 33 servers, the average service rate is $\bar{\mu} = 0.88$—quite close to the average of the low-load service rates of $\mu_1 = \mu_4 = 0.90$. Similarly, with 29 servers, the average service rate of $\bar{\mu} = 0.77$ is close to the overwork service rate of $\mu_3 = 0.75$. As a consequence, the Erlang C model "overshoots," both when predicting how much to increase staffing and when predicting how much to decrease staffing.

Instability: When we increase the low-load service rates to $\mu_1 = 0.95$ and $\mu_4 = 0.95$, the Erlang C model overestimates the appropriate decrease in staffing so drastically that the system becomes unstable. Starting with 35 servers, the Erlang C model recommends a decrease to 26 servers, which is unstable because $\lambda = 20 > 26\mu_3 = 26 \times 0.75 = 19.5$. For this system, we used simulation (with $T = 1{,}000$ hours) to obtain the service levels shown in Figure 22 and the system size sample path in Figure 23.

Figure 23. Simulated system size.



8. Conclusion

Queueing models typically assume that servers work at a constant speed, independent of the system state. Real servers adapt their speed to their environment in various ways, as empirical studies of healthcare and other service systems have demonstrated. Our aim in this paper has been to formulate a tractable and flexible stochastic model that captures two types of adaptive server behaviors: Response to the system load and response to the system overwork. Markov chains with state-dependent rates can model changes in server speed in response to changes in system load, as measured by the system occupancy, but tracking overwork requires additional state variables.

The model that we formulate can be seen as an extension to the Erlang C model. In addition to the system occupancy state variable that is included in most queueing models, we added one other state variable to capture overwork: We increment the overwork variable with each service completion in a high-load period and we decrement overwork by one unit at a time at a rate proportional to the number of idle servers during low-load periods. This method of operationalizing overwork leads to a model that can be represented as a QBD process with special transition structure that makes it possible to compute steady-state probabilities and standard queueing system performance measures efficiently.

We demonstrated through numerical experiments that when service rates depend on load and overwork, use of the Erlang C model provides a poor approximation of the system performance, even if one uses input averaging or output averaging. Using a stylized model of staffing practice, we illustrated how ongoing use of the Erlang C model for staffing with periodically updated arrival and service rate estimates can lead to convergence to a staffing level that is either too high or too low, staffing oscillation, and even to staffing levels that result in an unstable system.

One limitation of our model is that it has free parameters (k, m, s', and $\gamma_{i,j}$) for which the most appropriate values are not obvious. The two empirical operationalizations that we refer to share this limitation, having two free parameters for KC and Terwiesch (2009) (the load level $\bar{x}$ above which overwork accumulates and the time duration T) and one free parameter for Gans et al. (2010) (the "one hour" length of a service gap). KC and Terwiesch (2009) fix $\bar{x}$ to the average load and choose a value for T that maximizes model fit, but Gans et al. (2010) do not vary the "one hour."

We suggest setting $k = s$, so that overwork accumulates when all servers are busy and attenuates when at least one server is free. A natural default value for s' is $s' = s$, especially for a system with an invisible queue. We recommend setting m large enough to make $\Pr(J_\infty^2 = m)$, negligibly small where J_∞^2 is the steady-state random variable for $J(t)$ in Model M_2.

We expect that it will be difficult or impossible to determine values for our model parameters that are appropriate in all situations. We consider it important to investigate whether it is possible to develop guidelines for how to set these parameters in specific situations in which the mechanisms through which servers adapt to load and overwork are well understood. More generally, developing and testing statistical parameter estimation methods for the M_2 model is important.

Another limitation of our model is that it focuses on system adaptation to load and overwork rather than adaptation by individual servers. Aggregate modeling of a pool of servers is common in queueing theory (for example, this is done in the Erlang C model), but investigating models that capture individual server adaptation (see Zhan and Ward 2015, Gopalakrishnan et al. 2014, for example) is an important topic for future research.

Supplemental Material

Supplemental material to this paper is available at http://dx.doi.org/10.1287/opre.2016.1499.

Acknowledgments

The authors thank the editorial team for their valuable comments. This research was partly funded by the Canadian Natural Sciences and Engineering Research Council [Discovery Grant 203534 and the NSERC CREATE Program in Healthcare Operations and Information Management program]. This support is gratefully acknowledged.

References

Aksin OZ, Harker PT (2001) Modeling a phone center: Analysis of a multichannel, multiresource processor shared loss system. *Management Sci.* 47(2):324–336.

Alizamir S, de Véricourt F, Sun P (2013) Diagnostic accuracy under congestion. *Management Sci.* 59(1):157–171.

Armony M, Israelit S, Mandelbaum A, Marmor YN, Tseytlin Y, Yom-Tov GB (2015) On patient flow in hospitals: A data-based queueing-science perspective. *Stochastic Systems* 5(1):146–194.

Artalejo JR, Lopez-Herrero MJ (2001) Analysis of the busy period for the *M*/*M*/*c* queue: An algorithmic approach. *J. Appl. Probab.* 38(1): 209–222.

Batt R, Terwiesch C (2016) Early Task Initiation and Other Load-adaptive Mechanisms in the Emergency Department. *Management Sci*. Forthcoming.

Bellman R (1961) *Adaptive Control Processes: A Guided Tour* (Princeton University Press, Princeton, NJ).

Berk E, Moinzadeh K (1998) The impact of discharge decisions on health care quality. *Management Sci.* 44(3):400–415.

Bertsekas DP, Tsitsiklis JN (2008) *Introduction to Probability* (Athena Scientific, Belmont, MA).

Buffa ES, Cosgrove MJ, Luce BJ (1976) An integrated work shift scheduling system. *Decision Sci.* 7(4):620–630.

Chan CW, Farias VF, Escobar G (2014) The impact of delays on service times in the intensive care unit. Working paper. Columbia Business School, New York.

Chan CW, Yom-Tov G, Escobar G (2014) When to use speedup: An examination of intensive care units with readmissions. *Oper. Res.* 62(2): 462–482.

Crabill T (1972) Optimal control of a service facility with variable exponential service times and constant arrival rate. *Management Sci.* 18(9):560–566.

Dietz DC (2011) Practical scheduling for call center operations. *Omega* 39(5):550–557.

Dong J, Feldman P, Yom-Tov G (2015) Service systems with slowdowns: Potential failures and proposed solutions. *Oper. Res.* 63(2):305–324.

Edie LC (1954) Traffic delays at toll booths. *Oper. Res.* 2(2):107–138.

Gans N, Liu N, Mandelbaum A, Shen H, Ye H (2010) Service times in call centers: Agent heterogeneity and learning with some operational consequences. *A Festschrift for Lawrence D. Brown, IMS Collections* 6:99–123.

George JM, Harrison JM (2001) Dynamic control of a queue with adjustable service rate. *Oper. Res.* 49(5):720–731.

Gopalakrishnan R, Doroudi S, Ward AR, Wierman A (2014) Routing and staffing when servers are strategic. Working paper, Xerox Research Centre India, Bangalore, India.

Harris CM (1967) Queues with state-dependent stochastic service rates. *Oper. Res.* 15(1):117–130.

Hopp WJ, Iravani SM, Yuen GY (2007) Operations systems with discretionary task completion. *Management Sci.* 53(1):61–77.

Ingolfsson A, Tang L (2012) Efficient and reliable computation of birth-death process performance measures. *INFORMS J. Comput.* 24(1): 29–41.

Jackson JR (1963) Jobshop-like queueing systems. *Management Sci.* 10(1):131–142.

Jaeker JB, Tucker AL (2012). Hurry up and wait: Differential impacts of congestion, bottleneck pressure, and predictability on patient length of stay. Working Paper 13–052, Harvard Business School, Boston.

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

KC D, Terwiesch C (2012) An econometric analysis of patient flows in the cardiac intensive care unit. *Manufacturing Service Oper. Management* 14(1):50–65.

Khudyakov P, Gorfine M, Mandelbaum A (2010) Phase-type models of service times. In preparation.

Kuntz L, Mennicken R, Scholtes S (2014) Stress on the ward: Evidence of safety tipping points in hospitals. *Management Sci.* 61(4):754–771.

Lin CKY, Lai KF, Hung SL (2000) Development of a workforce management system for a customer hotline service. *Comput. Oper. Res.* 27(10):987–1004.

Neuts MF (1981) *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach* (Johns Hopkins Press, Baltimore).
Powell SG, Schultz KL (2004) Throughput in serial lines with state-dependent behavior. *Management Sci.* 50(8):1095–1105.
Powell SG, Schwaninger M, Trimble C (2001) Measurement and control of business processes. *System Dynam. Rev.* 17(1):63–91.
Ramaswami V, Lucantoni DM (1985) Stationary waiting time distribution in queues with phase type service and in quasi-birth-and-death processes. *Stochastic Models* 1(2):125–136.
Selen J, Adan I, Kulkarni V, van Leeuwaarden J (2015) The snowball effect of customer slowdown in critical many-server systems. Working paper, Eindhoven University of Technology, Eindhoven, Netherlands.
Staats BR, Gino F (2012) Specialization and variety in repetitive tasks: Evidence from a Japanese bank. *Management Sci.* 58(6):1141–1159.
Sze D (1984) A queueing model for telephone operator staffing. *Oper. Res.* 32(2):229–249.
Tan T, Netessine S (2014) When does the devil make work? An empirical study of the impact of workload on worker productivity. *Management Sci.* 60(6):1574–1593.
Van Houdt B, van Leeuwaarden JSH (2011) Triangular *M*/*G*/1-type and tree-like quasi-birth-death Markov chains. *INFORMS J. Comput.* 23(1):165–171.
van Leeuwaarden JSH, Winands EMM (2006) Quasi-birth-and-death processes with an explicit rate matrix. *Stochastic Models* 22(1): 77–98.
Welch PD (1964) On a generalized *M*/*G*/1 queuing process in which the first customer of each busy period receives exceptional service. *Oper. Res.* 12(5):736–752.
Zhan D, Ward AR (2015) Compensation and staffing to trade off speed and quality in large service systems. Working paper, Marshall School of Business, University of Southern California, Los Angeles.

Mohammad Delasay is a postdoctoral fellow of operations management at the Tepper School of Business, Carnegie Mellon University. His research is in the application of queueing and stochastic models to design and improve the performance of service and healthcare systems.

Armann Ingolfsson is the Roger H. Smith Professor of Business at the Alberta School of Business, University of Alberta. His research interests focus on operations management in the service and healthcare sectors and developing methodology to analyze congested systems. He is a past president of the Canadian Operational Research Society.

Bora Kolfal is an associate professor of operations management at the Alberta School of Business, University of Alberta. His main research interests are in service operations, especially healthcare applications of operations management, supply chain management, design and control of flexible systems, and queueing systems. He is also interested in operations management/information systems and operations management/marketing interdisciplinary projects with game theory models.