Exact sampling for some multi-dimensional queueing models with renewal input

Jose Blanchet * Yanan Pei * Karl Sigman *

May 26, 2017

Abstract

Using a recent result of Blanchet and Wallwater (2015: Exact sampling of stationary and time-reversed queues. *ACM TOMACS*, **25**, 26) for exactly simulating the maximum of a negative drift random walk queue endowed with i.i.d. increments, we extend it to a multi-dimensional setting and then we give a new algorithm for simulating exactly the stationary distribution of a *first-in-first-out* (FIFO) multi-server queue in which the arrival process is a general renewal process and the service times are independent and identically distributed; the FIFO $GI/GI/c$ queue with $2 \leq c < \infty$. Our method utilizes dominated coupling from the past (DCFP) as well as the Random Assignment (RA) discipline, and complements the earlier work in which Poisson arrivals were assumed, such as the recent work of Connor and Kendall (2015: Perfect simulation of $M/G/c$ queues. *Advances in Applied Probability*, **47**, 4). We also consider the models in continuous-time and show that with mild further assumptions, the exact simulation of those stationary distributions can also be achieved. We also give, using our FIFO algorithm, a new exact simulation algorithm for the stationary distribution of the infinite server case, the $GI/GI/\infty$ model. Finally, we even show how to handle *Fork-Join* queues, in which each arriving customer brings c jobs, one for each server.

Keywords EXACT SAMPLING; MULTI-SERVER QUEUE, RANDOM WALKS, RANDOM ASSIGNMENT; DOMINATED COUPLING FROM THE PAST.

AMS Subject Classification: 65C05; 90B22; 60K25;60J05;60K05;68U20.

*Department of Industrial Engineering and Operations Research, Columbia University, New York, NY 10027.

1 Introduction

In recent years, the method of *exact simulation* has evolved as a powerful way of sampling from stationary distributions of queueing models for which such distributions can not be derived explicitly. The main method itself is referred to as *coupling from the past* (CFP) as introduced in Propp and Wilson [15] for finite-state discrete-time Markov chains. Since then, the method has been generalized to cover general state space Markov chains by using dominating processes; this is known as *dominated coupling from the past* (DCFP) as in Kendall [14]. The main purpose of such methods is to produce a copy by simulation that exactly (not approximately) has the stationary distribution desired. These methods involve simulating processes backwards in time. In the present paper we consider using such methods for the FIFO multi-server queue, denoted as the FIFO $GI/GI/c$ queue, $2 \leq c < \infty$, where c denotes the number of servers working in parallel, and arriving customers wait in one common queue (line).

The first algorithms yielding exact simulation in stationarity of the FIFO $GI/GI/c$ queue are [16] and [17], in which Poisson arrivals are assumed; i.e., the $M/G/c$ case. In [16], a DCFP method is used, but the strong condition of *super stability* is assumed, $\rho < 1$, instead of $\rho < c$. ($\rho = E(S)/E(T)$, where T and S denote an interarrival time and service time respectively; stability only requires that $\rho < c$.) As a dominating process, the $M/G/1$ queue is used under Processor Sharing (PS) together (key) with its time-reversibility properties. In PS, there is no line; all customers are served simultaneously but at a rate $1/n$ when there are $n \geq 1$ customers in service. Then in [17], the general $\rho < c$ case is covered by using a forward time regenerative method (a general method developed in [3]) and using the $M/G/c$ model under a random assignment (RA) discipline as an upper bound; a model in which each arrival joins the i^{th} queue with probability $1/c$ independently. (The general forward-time regenerative method in [3] unfortunately always yields infinite expected termination time.) Then in [8], Connor and Kendall generalize the DCFP/PS method in [16] by using the RA model. They accomplish this by first exactly simulating the RA model in stationarity backwards in time under PS at each node, then re-constructing it to obtain the RA model with FIFO at each node and doing so in such a way that a sample-path upper bound for the FIFO $M/G/c$ is achieved.

As for renewal arrivals (the general FIFO $GI/GI/c$ queue considered here) the methods used above break down for various reasons, primarily because while under Poisson arrivals the c stations under RA become independent, they are not independent for general renewal arrivals. Also, the time-reversibility property of PS no longer holds, nor does Poisson Arrivals See Time Averages (PASTA). Finally, under general renewal arrivals, the system may never empty once it begins operating. New methods are needed. Blanchet, Dong and Pei ([6]) solve the problem by utilizing a vacation model as an upper bound. In the present paper, however, we utilize DCFP by directly simulating the RA model in reverse-time (under FIFO at each node). Our method involves extending, to a multi-dimensional setting, a recent result of Blanchet and Wallwater ([7]) for exactly simulating the maximum of a negative drift random walk endowed with i.i.d. increments. We also remark on how our approach can lead to new results for other models too, such as multi-server queues under the *last-in-first-out* (LIFO) discipline, or the *randomly choose next* discipline, and even Fork-Join models (also called split and match models).

2 The FIFO $GI/GI/c$ model

In what follows, as input to a c-server in parallel multi-server queue, we have i.i.d. service times $\{S_n : n \geq 0\}$ distributed as $G(x) = P(S \leq x),\ x \geq 0$, with finite and non-zero mean $0 < E(S) = 1/\mu < \infty$. Independently, the arrival times $\{t_n : n \geq 0\}$ ($t_0 = 0$) to the model form a renewal process with i.i.d. interarrival times $T_n = t_{n+1} - t_n,\ n \geq 0$ distributed as $A(x) = P(T \leq x),\ x \geq 0$, and finite non-zero arrival rate $0 < \lambda = E(T)^{-1} < \infty$. The FIFO

$GI/GI/c$ model has only one queue (line), and we let $\mathbf{W}_n = (W_n(1), \ldots, W_n(c))^T$ denote the *Kiefer-Wolfowitz workload vector* (see for example, Page 341 in Chapter 12 of [1]). It satisfies the recursion

$$\mathbf{W}_{n+1} = \mathcal{R}(\mathbf{W}_n + S_n\mathbf{e} - T_n\mathbf{f})^+, \ n \geq 0, \tag{1}$$

where $\mathbf{e} = (1, 0, \ldots 0)^T$, $\mathbf{f} = (1, 1, \ldots, 1)^T$, $\mathcal{R}$ places a vector in ascending order, and $^+$ takes the positive part of each coordinate. Let C_n denote the n^{th} customer and $D_n = W_n(1)$ is the customer delay in queue (line) of C_n. The idea is that at any time t we could (knowing all the whole or partial service times of all in the system) place the customers in front of the servers where they will eventually enter service. This results in c queues, each with its own total workload. The workload vector above does this at the arrival times t_n- and places these c workloads in ascending order; the smallest one $W_n(1)$ is where C_n will go for service, since that is the server that will be free first for C_n; thus $D_n = W_n(1)$. Recursion (1) defines a Markov chain due to the given i.i.d. assumptions.

With stability condition $\rho = \lambda/\mu < c$, it is well known that $\mathbf{W}_n$ converges in distribution as $n \to \infty$ to a proper stationary distribution. Let π denote this stationary distribution. Our main objective in the present paper is to provide a simulation algorithm for sampling exactly from π.

3 The RA $GI/GI/c$ model

Given a c-server queueing system, the random assignment model (RA) is the case when each of the c servers forms its own FIFO single-server queue, and each arrival to the system, independent of the past, randomly chooses queue i to join with probability $1/c$, $1 \leq i \leq c$. In the $GI/GI/c$ case, we refer to this as the RA $GI/GI/c$ model. The following is a special case of Lemma 1.3, Page 342 in [1]. (Such results and others even more general are based on [21], [11], and [12].)

Lemma 3.1 *Let $Q_F(t)$ denote total number of customers in system at time $t \geq 0$ for the FIFO $GI/GI/c$ model, and let $Q_{RA}(t)$ denote total number of customers in system at time $t \geq 0$ for the corresponding RA $GI/GI/c$ model in which both models are initially empty and fed with exactly the same input of renewal arrivals $\{t_n : n \geq 0\}$ and i.i.d. service times $\{S_n : n \geq 0\}$. Assume further that for both models the service times are used by the servers in the order in which service initiations occur (S_n is the service time used for the n^{th} such initiation). Then*

$$P(Q_F(t) \leq Q_{RA}(t), \ \textit{for all } t \geq 0) = 1. \tag{2}$$

The importance of Lemma 3.1 is that it allows us to jointly simulate versions of the two stochastic processes $\{Q_F(t) : t \geq 0\}$ and $\{Q_{RA}(t) : t \geq 0\}$ while achieving a coupling such that (2) holds. In particular, whenever an arrival finds the RA model empty, the FIFO model is found empty as well. (But we need to impose further conditions if we wish to ensure that indeed the RA $GI/GI/c$ queue will empty with certainty.) Letting time t be sampled at arrival times of customers, $\{t_n : n \geq 0\}$, we thus also have

$$P(Q_F(t_n-) \leq Q_{RA}(t_n-), \ \text{for all } n \geq 0) = 1. \tag{3}$$

In other words, the total number in system as found by the n^{th} arrival is sample-path ordered as well. Note that for the FIFO model, the n^{th} arriving customer C_n initiates the n^{th} service since FIFO means "First-In-Queue-First-Out-of-Queue" where by "queue" we mean the line before entering service. This means that for the FIFO model we can attach S_n to C_n upon arrival if we so wish when applying Lemma 3.1. For the RA model, however, customers are not served in the order they arrive. For example consider $c = 2$ servers (system initially empty)

and suppose C_1 is assigned to node 1 with service time S_1, and C_2 also is assigned to node 1 (before C_1 departs) with service time S_2. Meanwhile, before C_1 departs, suppose C_3 arrives and is assigned to the empty node 2 with service time S_3. Then S_3 is used for the second service initiation. *For RA, the service times in order of initiation are a random permutation of the originally assigned* $\{S_n\}$.

To use Lemma 3.1, it is crucial to simply let the server hand out service times one at a time when they are needed for a service initiation. Thus, customers waiting in a queue before starting service do not have a service time assigned until they enter service. In simulation terminology, this amounts to generating the service times in order of when they are needed.

One disadvantage of generating service times only when they are needed, is that it then does not allow workload[1] to be defined; only the amount of work in service. To get around this if need be, one can simply generate service times upon arrival of customers, and give them to the server to be used in order of service initiation. The point is that when C_n arrives, the total work in system jumps up by the amount S_n. But S_n is not assigned to C_n, it is assigned (perhaps later) to which ever customer initiates the n^{th} service. This allows Lemma 3.1 to hold true for total amount of work in the system: If we let $\{V_F(t) : t \geq 0\}$ and $\{V_{RA}(t) : t \geq 0\}$ denote total workload in the two models with the service times used in the manner just explained, then in addition to Lemma 3.1 we have

$$P(V_F(t) \leq V_{RA}(t), \text{ for all } t \geq 0) = 1, \tag{4}$$

$$P(V_F(t_n-) \leq V_{RA}(t_n-), \text{ for all } n \geq 0) = 1. \tag{5}$$

It is important, however, to note that what one can't do is define workload at the individual nodes i by doing this, because that forces us to assign S_n to C_n so that workload at the node that C_n attends (i say) jumps by S_n and C_n enters service using S_n; that destroys the proper coupling needed to obtain Lemma 3.1. We can only handle the total (sum over all c nodes) workload. In the present paper, our use of Lemma 3.1 is via a kind of reversal:

Lemma 3.2 *Let* $\{S'_n\}$ *be an i.i.d. sequence of service times distributed as* G*, and assign* S'_n *to* C_n *in the RA model. Define* S_n *as the service time used in the* n^{th} *service initiation. Then* $\{S_n\}$ *is also i.i.d. distributed as* G.

Proof : The key is noting that we are re-ordering based only on the order in which service times begin being used, not when they are completed (which would thus introduce a bias). The service time chosen for the next initiation either enters service immediately (e.g., is one that is routed to an empty queue by an arriving customer) or is chosen from among those waiting in lines, and all those waiting are i.i.d. distributed as G. Let $\hat{t}_n$ denote the time at which the n^{th} service initiation begins. The value S_n of the n^{th} service time chosen (at time $\hat{t}_n$) by a server is independent of the past service time values used before time $\hat{t}_n$, and is distributed as G (the choice of service time chosen as the next to be used is not based on the value of the service time, only its position in the lines). Letting $k(n) =$ the index of the $\{S'_n\}$ that is chosen, e.g., $S_n = S'_{k(n)}$, it is this index (a random variable) that depends on the past, but the value S_n is independent of $k(n)$ since it is a new one. Thus the $\{S_n\}$ are i.i.d. distributed as G. ■

The point of the above Lemma 3.2 is that we can, if we so wish, simulate the RA model by assigning S'_n to C_n (to be used as their service time), but then assigning S_n, i.e. $S'_{k(n)}$, to C_n in the FIFO model. By doing so the requirements of Lemma 3.1 are satisfied and (2), (3), (4) and (5) hold. Interestingly, however, it is not possible to first simulate the RA model up

[1]Workload (total) at any time t is defined as the sum of all whole and remaining service times in the system at time t.

to a fixed time t, and then stop and reconstruct the FIFO model up to this time t: At time t, there may still be RA customers waiting in lines and hence not enough of the S_n have been determined yet to construct the FIFO model. But all we have to do, if need be, is to continue the simulation of the RA model beyond t until enough S_n have been determined to construct fully the FIFO model up to time t.

4 Simulating exactly from the stationary distribution of the RA $GI/GI/c$ model

By Lemma 3.1, the RA $GI/GI/c$ queue, which shares the same arrival stream $\{t_n : n \geq 0\}$ ($t_0 = 0$) and same service times in the order of service initiations $\{S_n : n \geq 0\}$, will serve as a sample path upper bound (in terms of total number of customers in system and total workload) of the target FIFO $GI/GI/c$ queue. Independent of $\{T_n : n \geq 0\}$ and $\{S_n : n \geq 0\}$, we let $\{U_n : n \geq 0\}$ be an i.i.d. sequence of random variables from discrete uniform distribution on $\{1, 2, \ldots, c\}$, i.e., U_n represents the choice that customer C_n makes about which single-server queue to join under RA discipline. Let $\mathbf{V}_n = (V_n(1), \ldots, V_n(c))^T$ denote the workload vector as found by C_n in the RA $GI/GI/c$ model, so for $i = 1, \ldots, c$,

$$V_{n+1}(i) = (V_n(i) + S_n I(U_n = i) - T_n)^+, \quad n \geq 0. \tag{6}$$

These c processes are dependent through the common arrival times $\{t_n : n \geq 0\}$ (equivalently common interarrival times $\{T_n : n \geq 0\}$) and the common $\{U_n : n \geq 0\}$ random variables. Because of all the i.i.d. assumptions, $\{\mathbf{V}_n : n \geq 0\}$ forms a Markov chain. Define $\tilde{\mathbf{S}}_n = (S_n I(U_n = 1), \ldots, S_n I(U_n = c))^T$ and $\mathbf{T}_n = T_n \mathbf{f}$, then we can express (6) in vector form as

$$\mathbf{V}_{n+1} = \left(\mathbf{V}_n + \tilde{\mathbf{S}}_n - \mathbf{T}_n\right)^+, \quad n \geq 0. \tag{7}$$

Each node i as expressed in (6) can be viewed as a FIFO $GI/GI/1$ queue with common renewal arrival process $\{t_n : n \geq 0\}$, but with i.i.d. service times $\{\tilde{S}_n(i) : n \geq 0\}$. Across i, the service times $(\tilde{S}_n(1), \ldots, \tilde{S}_n(c))$ are not independent, but they are identically distributed: marginally, with probability $1/c$, $\tilde{S}_n(i)$ is distributed as G, and with probability $(c-1)/c$ it is distributed as the point mass at 0; i.e., $E(\tilde{S}(i)) = E(S)/c$. The point here is that we are not treating node i as a single-server queue endowed only with its own arrivals (a thinning of the $\{t_n : n \geq 0\}$ sequence) and its own service times i.i.d. distributed as G. Defining i.i.d. increments $\Delta_n(i) = S_n I(U_n = i) - T_n$ for $n \geq 0$, each node i has an associated negative drift random walk $\{R_n(i) : n \geq 0\}$ with $R_0(i) = 0$ and

$$R_n(i) = \sum_{j=1}^{n} \Delta_j(i), \quad n \geq 1. \tag{8}$$

With $\rho = \lambda E(S) < c$, we define $\rho_i = \lambda E(\tilde{S}(i)) = \lambda E(S)/c = \rho/c < 1$; equivalently $E(\Delta(i)) < 0$ for all $i = 1, \ldots, c$. Let $V^0(i)$ denote a random variable with the limiting (stationary) distribution of $V_n(i)$ as $n \to \infty$, it is well known (due to the i.i.d. assumptions) that $V^0(i)$ has the same distribution as

$$M(i) \triangleq \max_{m \geq 0} R_m(i)$$

for $i = 1, \ldots, c$.

More generally, even when the increment sequence is just stationary ergodic, not necessarily i.i.d. (hence not time reversible as in the i.i.d. case), it is the backwards in time maximum that is used in constructing a stationary version of $\{V_n(i)\}$. We will need this backwards approach in our simulation so we go over it here; it is usually referred to as *Loynes' Lemma*. We extend the arrival point process $\{t_n : n \geq 0\}$ to be a two-sided point stationary renewal process $\{t_n : n \in \mathbb{Z}\}$

$$\cdots - t_{-2} < t_{-1} < 0 = t_0 < t_1 < t_2 \cdots$$

Equivalently, $T_n = t_{n+1} - t_n,\ n \in \mathbb{Z}$, form i.i.d. interarrival times; $\{T_n : n \in \mathbb{Z}\}$ forms a two-sided i.i.d. sequence.

Similarly, the i.i.d. sequences $\{S_n : n \geq 0\}$ and $\{U_n : n \geq 0\}$ are extended to be two-sided i.i.d., $\{S_n : n \in \mathbb{Z}\}$ and $\{U_n : n \in \mathbb{Z}\}$. These extensions further allow two-sided extension of the i.i.d. increment sequences $\{\Delta_n(i) : n \in \mathbb{Z}\}$ for $i = 1, \ldots, c$, i.e.,

$$\Delta_n(i) = S_n I\left(U_n = i\right) - T_n, \quad n \in \mathbb{Z}.$$

Then we define c time-reversed (increments) random walks $\{R_n^{(r)}(i) : n \geq 0\}$ for $i = 1, \ldots, c$, by $R_0^{(r)}(i) = 0$ and

$$R_n^{(r)}(i) = \sum_{j=1}^{n} \Delta_{-j}(i), \quad n \geq 1. \tag{9}$$

A (from the infinite past) stationary version of $\{V_n(i)\}$ denoted by $\{V_n^0(i) : n \leq 0\}$ is then constructed via

$$\begin{aligned}
V_0^0(i) &= \max_{m \geq 0} R_m^{(r)}(i), && (10)\\
V_{-1}^0(i) &= \max_{m \geq 1} R_m^{(r)}(i) - R_1^{(r)}(i), && (11)\\
V_{-2}^0(i) &= \max_{m \geq 2} R_m^{(r)}(i) - R_2^{(r)}(i), && (12)\\
&\vdots \\
V_{-n}^0(i) &= \max_{m \geq n} R_m^{(r)}(i) - R_n^{(r)}(i), && (13)
\end{aligned}$$

for all $i = 1, \ldots, c$.

By construction, the process $\mathbf{V}_n^0 = (V_n^0(1), \ldots, V_n^0(c))^T,\ n \leq 0$, is jointly stationary representing a (from the infinite past) stationary version of $\{\mathbf{V}_n : n \leq 0\}$, and satisfies the forward-time recursion (7):

$$\mathbf{V}_{n+1}^0 = (\mathbf{V}_n^0 + \tilde{\mathbf{S}}_n - \mathbf{T}_n)^+, \quad n \leq -1. \tag{14}$$

Thus, by starting at $n = 0$ and walking backwards in time, we have (theoretically) a time-reversed copy of the RA model. Furthermore, $\{\mathbf{V}_n^0 : n \leq 0\}$ can be extended to include forward time $n \geq 1$ via using the recursion further:

$$\mathbf{V}_n^0 = (\mathbf{V}_{n-1}^0 + \tilde{\mathbf{S}}_{n-1} - \mathbf{T}_{n-1})^+, \quad n \geq 1, \tag{15}$$

where $\tilde{\mathbf{S}}_n = (S_n I(U_n = 1), \ldots, S_n I(U_n = c))^T$ for $n \in \mathbb{Z}$.

In fact once we have a copy of just $\mathbf{V}_0^0$, we can start off the Markov chain with it as initial condition and use (15) to obtain a forward in time stationary version $\{\mathbf{V}_n^0 : n \geq 0\}$.

The above "construction", however, is theoretical. We do not yet have any explicit way of obtaining a copy of $\mathbf{V}_0^0$, let alone an entire from-the-infinite-past sequence $\{\mathbf{V}_n^0 : n \leq 0\}$. In

Blanchet and Wallwater [7], a simulation algorithm is given that yields (when applied to each of our random walks), for each $1 \leq i \leq c$, a copy of $\{(R_n^{(r)}(i), V_{-n}^0(i)) : 0 \leq n \leq N\}$ for any desired $0 \leq N < \infty$ including stopping times N. We modify the algorithm so that it can do the simulation jointly across the c systems, that is, we extend it to a multi-dimensional form.

In particular, it yields an algorithm for obtaining a copy of $\mathbf{V}_0^0$, as well as a finite segment (of length N) of a backwards in time copy of the RA model; $\{\mathbf{V}_{-n}^0 : 0 \leq n \leq N\}$, a stationary into the past construction up to discrete time $n = -N$.

Finite exponential moments are not required (because only *truncated* exponential moments are needed $E(e^{\gamma\Delta(i)}I\{|\Delta(i)| \leq a\})$, which in turn allow for the simulation of the exponential tilting of truncated $\Delta(i)$, via acceptance-rejection). To get finite expected termination time (at each individual node) one needs moments slightly beyond 2: For some (explicitly known) $\epsilon > 0$, $E(S^{2+\epsilon}) < \infty$ and $E(T^{2+\epsilon}) < \infty$.

As our first case, we will be considering a stopping time N such that $\mathbf{V}_{-N} = \mathbf{0}$. Before we give the definition of the stopping time N, we introduce the main idea of our simulation algorithm.

Let us define the maximum of a sequence of vectors. Suppose we have $\mathbf{X}_1, \cdots, \mathbf{X}_k$, where $\mathbf{X}_i \in \mathbb{R}^c$ with $c \geq 1$ and $k \in \mathbb{N}_+ \cup \{\infty\}$, define

$$\max(\mathbf{X}_1, \cdots, \mathbf{X}_k) = \left(\max_{1\leq l\leq k} X_l(1), \ldots, \max_{1\leq l\leq k} X_l(c)\right)^T.$$

Next define, for $n \in \mathbb{Z}$, that

$$\mathbf{U}_n = (I(U_n = 1), \ldots, I(U_n = c))^T \quad \text{and} \quad \mathbf{\Delta}_n = \tilde{\mathbf{S}}_n - \mathbf{T}_n = S_n\mathbf{U}_n - T_n\mathbf{f},$$

where $\{U_n\}_{n\in\mathbb{Z}}$ are i.i.d. from discrete uniform distribution over $\{1, 2, \ldots, c\}$, and independently $\{T_n\}_{n\in\mathbb{Z}}$ are i.i.d. from distribution A (as introduced in Section 2). Our goal is to simulate a stopping time $N \in \mathbb{N}$ such that

$$\mathbf{V}_{-N}^0 = \max_{k\geq N} \mathbf{R}_k^{(r)} - \mathbf{R}_N^{(r)} = \max_{k\geq N} \sum_{j=1}^{k} \mathbf{\Delta}_{-j} - \sum_{j=1}^{N} \mathbf{\Delta}_{-j} = \mathbf{0},$$

jointly with $\{(\mathbf{R}_n^{(r)}, \mathbf{V}_{-n}^0) : 0 \leq n \leq N\}$. (By convention, the value of any empty sum of numbers is zero, i.e. $\sum_{j=1}^{0} a_j = 0$.)

Because of the stability condition $\rho = \lambda/\mu < c$, we can find some value $a \in (1/\mu, c/\lambda)$ such that $\mathbb{P}(T > a) > 0$, then for any $n \geq 0$,

$$\begin{aligned}
\max_{k\geq n} \mathbf{R}_k^{(r)} &= \max_{k\geq n} \sum_{j=1}^{k} \mathbf{\Delta}_{-j} = \max_{k\geq n} \sum_{j=1}^{k} (S_{-j}\mathbf{U}_{-j} - \mathbf{T}_{-j}) \\
&= \max_{k\geq n} \sum_{j=1}^{k} ((S_{-j}\mathbf{U}_{-j} - a\mathbf{U}_{-j}) + (a\mathbf{U}_{-j} - \mathbf{T}_{-j})) \\
&\leq \max_{k\geq n} \sum_{j=1}^{k} ((S_{-j} - a)\,\mathbf{U}_{-j}) + \max_{k\geq n} \sum_{j=1}^{k} (a\mathbf{U}_{-j} - \mathbf{T}_{-j}). \qquad (16)
\end{aligned}$$

Let

$$\mathbf{X}_{-k} := \sum_{j=1}^{k} (S_{-j} - a)\,\mathbf{U}_{-j} \quad \text{and} \quad \mathbf{Y}_{-k} := \sum_{j=1}^{k} (a\mathbf{U}_{-j} - \mathbf{T}_{-j})$$

for $k \in \mathbb{N}$, and define

$$N = \inf\{n \geq 0 : \max_{k \geq n} \mathbf{X}_{-k} = \mathbf{X}_{-n}, \ \max_{k \geq n} \mathbf{Y}_{-k} = \mathbf{Y}_{-n}\}. \tag{17}$$

By the inequality (16), we have $\max_{m \geq N} \mathbf{R}_m^{(r)} = \mathbf{R}_N^{(r)}$, hence $\mathbf{V}_{-N}^0 = \mathbf{0}$.

To ensure that $P(N < \infty) = 1$, in addition to $\rho < c$ (stability), it is required that $P(T > S) > 0$, for which the most common sufficient conditions are that T has unbounded support, $P(T > t) > 0, \ t \geq 0$, or S has mass arbitrarily close to 0, $P(S < t) > 0, \ t > 0$. But as we shall show in Section 6, given we know that $P(T > S) > 0$, we can assume without loss of generality that interarrival times are bounded. It is that assumption which makes the extension of [7] to a multidimensional form easier to accomplish. Then, we show (in Section 4.2 and Section 9) how to still simulate from π even when $P(T > S) = 0$. We do that in two different ways, one as sandwiching argument and the other involving Harris recurrent Markov chain regenerations.

4.1 Algorithm for simulating exactly from π for the FIFO $GI/GI/c$ queue: The case $P(T > S) > 0$

As mentioned earlier, we will assume that $P(T > S) > 0$, so that the stable ($\rho < c$) RA and FIFO $GI/GI/c$ Markov chains (7) and (1) will visit 0 infinitely often with certainty. (That the RA model empties infinitely often when $P(T > S) > 0$ is proved for example in [19]). We imagine that at the infinite past $n = -\infty$, we start both (7) and (1) from empty. We construct the RA model forward in time, while using Lemma 3.2 for the service times for the FIFO model, so that Lemma 3.1 applies and we have it in the form of (3), for all $t_n \leq 0$ up to and including at time $t_0 = 0$, at which time both models are in stationarity. We might have to continue the construction of the RA model so that $\mathbf{W}_0$ (distributed as π) can be constructed (e.g., enough service times have been initiated by the RA model for using Lemmas 3.1 and 3.2). Formally, one can theoretically justify the existence of such infinite from the past versions (that obey Lemma 3.1) – by use of Loynes' Lemma. Each model (when started empty) satisfies the monotonicity required to use Loynes' Lemma. In particular, noting that $Q_{RA}(t_n-) = 0$ if and only if $\mathbf{V}_n = \mathbf{0}$, we conclude that if at any time n it holds that $\mathbf{V}_n = \mathbf{0}$, then $\mathbf{W}_n = \mathbf{0}$. By the Markov property, given that $\mathbf{V}_n = \mathbf{0} = \mathbf{W}_n$, the future is independent of the past for each model, or said differently, *the past is independent of the future*. This remains valid if n is replaced by a stopping time (strong Markov property).

We outline the simulation algorithm steps as follows.

1. Simulate $\{\{(R_n^{(r)}(i), V_{-n}^0(i)) : 0 \leq n \leq N\}, 1 \leq i \leq c\}$ with N as defined in (17). If $N = 0$, go to next step. Otherwise, having stored all data, reconstruct $\mathbf{V}_n^0$ forwards in time from $n = -N$ (initially empty) until $n = 0$, using the recursion (14). During this forward-time reconstruction, re-define S_j as the j-th service initiation used by the RA model (i.e., we are using Lemma 3.2 to gather service times in the proper order to feed in the FIFO model, which is why we do the re-construction). If at time $n = 0$ there have not yet been N service initiations, then continue simulating the RA model out in forward time until finally there is a N^{th} service initiation, and then stop. This will require, at most, simulating out to t_n with $n = N^{(+)} = \min\{n \geq 0 : \mathbf{V}_{-n}^0 = \mathbf{0}\}$. Take the vector $(S_{-N}, S_{-N+1}, \ldots, S_{-1})$ and reset $(S_0, S_1, \ldots, S_{N-1}) = (S_{-N}, S_{-N+1}, \ldots, S_{-1})$. Also, store the interarrival times $(T_{-N}, T_{-N+1}, \ldots, T_{-1})$, and reset $(T_0, \ldots, T_{N-1}) = (T_{-N}, T_{-N+1}, \ldots, T_{-1})$.

2. If $N = 0$, then set $\mathbf{W}_0 = \mathbf{0}$ and stop. Otherwise use (1) with $\mathbf{W}_0 = \mathbf{0}$, recursively go forwards in time for N steps until obtaining $\mathbf{W}_N$, by using the N re-set service $(S_0, S_1, \ldots, S_{N-1})$ and interarrival times $(T_0, \ldots, T_{N-1})$. Reset $\mathbf{W}_0 = \mathbf{W}_N$.

3. Output $\mathbf{W}_0$.

Detailed simulation steps are discussed in Appendix (Section 10).

Proposition 4.1 *If $\rho = \lambda/\mu < c$, there exists some $\epsilon > 0$ such that $E(S^{2+\epsilon}) < \infty$ and $E(T^{2+\epsilon}) < \infty$, and there exists constant $a > 0$ such that $1/\mu < a < c/\lambda$ and $P(T > a) < \infty$, then*

$$E(N) < \infty.$$

Proof : ■

4.2 A more efficient algorithm: sandwiching

In this section, we no longer even need to assume that $P(T > S) > 0$. (Another method allowing for $P(T > S) = 0$ involving Harris recurrent regeneration is given later in Section 9.) Instead of waiting for the workload vector of the $GI/GI/c$ queue under RA discipline to become $\mathbf{0}$, we choose an "inspection time" $t_{-\kappa} < 0$ for some $\kappa \in \mathbb{Z}_+$ to stop the backward simulation of the RA $GI/GI/c$ queue, then construct two bounding processes of the target FIFO $GI/GI/c$ queue and evolve them forward in time, using the same stream of arrivals and service time requirements (in the order of service initiations), until coalescence or time zero. In particular we let the upper bound process to be a FIFO $GI/GI/c$ queue starting at time $t_{-\kappa}$ with workload vector being $\mathbf{V}^0_{-\kappa}$, and let the lower bound process to be a FIFO $GI/GI/c$ queue starting at the same time $t_{-\kappa}$ from empty, i.e., with workload vector being $\mathbf{0}$.

Let $\mathbf{W}(t)$ denote the ordered (ascendingly) workload vector of the original FIFO $GI/GI/c$ queueing process, starting from the infinite past, evaluated at time t. For $t \geq t_{-\kappa}$, we define $\mathbf{W}^u_{-\kappa}(t)$ and $\mathbf{W}^l_{-\kappa}(t)$ to be the ordered (ascendingly) workload vectors of the upper bound and lower bound processes, initiated at the inspection time $t_{-\kappa}$, evaluated at time t. By our construction and Theorem 3.3 in [8],

$$\mathbf{W}^u_{-\kappa}(t_{-\kappa}) = \mathcal{R}\left(\mathbf{V}^0_{-\kappa}\right) \geq \mathbf{W}(t_{-\kappa}) \geq \mathbf{W}^l_{-\kappa}(t_{-\kappa}) = \mathbf{0},$$

and for all $t > t_{-\kappa}$

$$\mathbf{W}^u_{-\kappa}(t) \geq \mathbf{W}(t) \geq \mathbf{W}^l_{-\kappa}(t),$$

where all the above inequalities hold coordinate-wise.

Note that we can evolve the ordered workload vectors of the two bounding processes as follows: for $t_{n-1} \leq t < t_n$ when $-\kappa < n \leq -1$,

$$\begin{aligned} \mathbf{W}^u_{-\kappa}(t) &= \mathcal{R}\left(\mathbf{W}^u_{-\kappa}(t_{n-1}) + S_{n-1}\mathbf{e} - (t - t_{n-1})\mathbf{f}\right)^+, \\ \mathbf{W}^l_{-\kappa}(t) &= \mathcal{R}\left(\mathbf{W}^l_{-\kappa}(t_{n-1}) + S_{n-1}\mathbf{e} - (t - t_{n-1})\mathbf{f}\right)^+. \end{aligned} \tag{18}$$

Similarly let $Q(t)$ denote the number of customers in the original FIFO $GI/GI/c$ queueing process, starting from the infinite past, evaluated at time t. For $t \geq t_{-\kappa}$, we let $Q^u_{-\kappa}(t)$ and $Q^l_{-\kappa}(t)$ denote the number of customers in the upper and lower bound queueing processes respectively, both initiated at the inspection time $t_{-\kappa}$, evaluated at time t. If at some time $\tau \in [t_{-\kappa}, 0]$, we observe that $\mathbf{W}^u_{-\kappa}(\tau) = \mathbf{W}^l_{-\kappa}(\tau)$, then it must be true that $\mathbf{W}(\tau) = \mathbf{W}^u_{-\kappa}(\tau) = \mathbf{W}^l_{-\kappa}(\tau)$ and $Q(\tau) = Q^u_{-\kappa}(\tau) = Q^l_{-\kappa}(\tau)$ (because the ordered remaining workload vectors of two bounding processes can only meet when they both have idle servers). We call such time τ "coalescence time" and from then on we have full information of the target FIFO $GI/GI/c$ queue, hence we can continue simulate it forward in time until time 0.

However if coalescence does not happen by time 0, we can adopt the so-called "binary back-off" method by letting the arrival time $t_{-2\kappa}$ be our new inspection time and redo the above procedure to detect coalescence. Theorem 3.3 in [8] ensures that for any $t_{-\kappa} \le t \le 0$

$$\mathbf{W}^u_{-\kappa}(t) \ge \mathbf{W}^u_{-2\kappa}(t) \ge \mathbf{W}(t) \ge \mathbf{W}^l_{-2\kappa}(t) \ge \mathbf{W}^l_{-\kappa}(t).$$

We summarize the sandwiching algorithm as follows.

1. Simulate $\{(\mathbf{R}^{(r)}_n, \mathbf{V}^0_{-n}) : 0 \le n \le \kappa\}$ with all data stored.
2. Use the stored data to reconstruct $\mathbf{V}^0_n$ forward in time from $n = -\kappa$ until $n = 0$, using equation (14), and re-define S_j as the j^{th} service initiation used by the RA model.
3. Set $\mathbf{W}^u_{-\kappa}(t_{-\kappa}) = \mathcal{R}(\mathbf{V}^0_{-\kappa})$ and $\mathbf{W}^l_{-\kappa}(t_{-\kappa}) = \mathbf{0}$. Then use the same stream of interarrival times $(T_{-\kappa}, T_{-\kappa+1}, \cdots, T_{-1})$ and service times $(S_{-\kappa}, S_{-\kappa+1}, \cdots, S_{-1})$ to simulate $\mathbf{W}^u_{-\kappa}(t)$, $\mathbf{W}^l_{-\kappa}(t)$ forward in time using equation (18).
4. If at some time $t \in [t_{-\kappa}, 0]$ we detect $\mathbf{W}^u_{-\kappa}(t) = \mathbf{W}^l_{-\kappa}(t)$, set $\tau = t$, $\mathbf{W}(\tau) = \mathbf{W}^u_{-\kappa}(\tau)$, $Q(\tau) = \sum_{i=1}^c I(\mathbf{W}(\tau; i) > 0)$, where $\mathbf{W}(t; i)$ is the i-th entry of vector $\mathbf{W}(t)$. Then use the remaining interarrival times and service times to evolve the original FIFO $GI/GI/c$ queue forward in time until time $t_0 = 0$, output $(\mathbf{W}(0), Q(0))$ and stop.
5. If no coalescence is detected by time 0, set $\kappa = 2\kappa$, then continue to simulate the backward RA $GI/GI/c$ process until $(-\kappa)$-th arrival, i.e., $\{(\mathbf{R}^{(r)}_n, \mathbf{V}^0_{-n}) : 0 \le n \le \kappa\}$, with all data stored. Go to Step 2.

Next we analyze properties of the coalescence time. Define

$$\kappa^*_- = \inf\left\{ n \ge 0 : \inf_{t_{-n} \le t \le 0} \|\mathbf{W}^u_{-n}(t) - \mathbf{W}^l_{-n}(t)\|_\infty = 0 \right\}.$$

If at time $t_{-\kappa^*_-}$ we start an upper bound FIFO $GI/GI/c$ queue with workload vector being $\mathbf{W}^u_{-\kappa^*_-}(t_{-\kappa^*_-})$, and a lower bound FIFO $GI/GI/c$ queue with workload vector being $\mathbf{0}$, they will coalesce by time $t_0 = 0$. Therefore if we simulate the RA system backwards in time to $t_{-\kappa^*_-}$, we will be able to detect a coalescence. We next show that $E(-t_{-\kappa^*_-}) < \infty$.

By stationarity we have that κ^*_- is equal in distribution to

$$\kappa^*_+ = \inf\left\{ n \ge 0 : \inf_{0 \le t \le t_n} \|\mathbf{W}^u_0(t) - \mathbf{W}^l_0(t)\|_\infty = 0 \right\},$$

hence $-t_{-\kappa^*_-} \stackrel{d}{=} t_{\kappa^*_+}$.

Proposition 4.2 *If $\rho = E(S)/E(T) < c$ and there exists some $\epsilon > 0$ such that $E(S^{2+\epsilon}) < \infty$ and $E(T^{2+\epsilon}) < \infty$, then*

$$E(t_{\kappa^*_+}) < \infty.$$

Proof : By Wald's identity, it suffices to show that $E(\kappa^*_+) < \infty$ because $E(T) < \infty$. Next we only provide a proof outline here since it follows the same argument as in the proof of Proposition 3 in [6].

Firstly, we construct a sequence of events $\{\Omega_k : k \ge 1\}$ which leads to the occurrence of κ^*_+. Secondly, we split the process $\{\mathbf{W}^u_0(t_n) : n \ge 0\}$ into cycles with bounded expected cycle length. We also ensure the probability that the event happens during each cycle is bounded from below by a constant, which allows us to bound the number of cycles we need to check before finding κ^*_+ by a Geometric random variable. Finally we could establish an upper bound for $E(\kappa^*_+)$ by applying Wald's identity again. ■

5 Continuous-time stationary constructions

For a stable FIFO $GI/GI/1$ queue, let D denote stationary customer delay (time spent in queue (line)); i.e., it has the limiting distribution of $D_{n+1} = (D_n + S_n - T_n)^+$ as $n \to \infty$.

Independently, let S_e denote a random variable distributed as the *equilibrium distribution* G_e of service time distribution G,

$$G_e(x) = \mu \int_0^x P(S > y)dy, \ x \geq 0, \tag{19}$$

where $S \sim G$. Let $V(t)$ denote total work in system at time t; the sum of all whole or remaining service times in the system at time t. $D_n = V(t_n-)$, and one can construct $\{V(t)\}$ via

$$V(t) = (D_n + S_n - (t - t_n))^+, \ t_n \leq t < t_n + 1.$$

(It is to be continuous from the right with left limits.) Let V denote stationary workload; e.g., it has the limiting distribution

$$P(V \leq x) = \lim_{t\to\infty} \frac{1}{t} \int_0^t P(V(s) \leq x)ds, \ x \geq 0. \tag{20}$$

The following is well known to hold (see Section 6.3 and 6.4 in [18], for example):

$$P(V > x) = \rho P(D + S_e > x), \ x \geq 0. \tag{21}$$

Letting $F_D(x) = P(D \leq x)$ denote the probability distribution of D, and δ_0 denote the point mass at 0, and $*$ denote convolution of distributions, this means that the distribution of V can be written as a mixture

$$(1 - \rho)\delta_0 + \rho F_D * G_e.$$

This leads to the following:

Proposition 5.1 *For a stable* $(0 < \rho < 1)$ *FIFO* $GI/GI/1$ *queue, if* ρ *is explicitly known, and one can exactly simulate from* D *and* G_e*, then one can exactly simulate from* V*.*

Proof :

1. Simulate a Bernoulli (ρ) r.v. B.
2. If $B = 0$, then set $V = 0$. Otherwise, if $B = 1$, then: simulate D and independently simulate a copy $S_e \sim G_e$. Set $V = D + S_e$. Stop.

■

Another algorithm requiring instead the ability to simulate from A_e (equilibrium distribution of the interarrival-time distribution A) instead of G_e follows from another known relation:

$$V \stackrel{d}{=} (D + S - T_e)^+, \tag{22}$$

where D, S and $T_e \sim A_e$ are independent (see, for example Equation 88 on Page 426 in [20]). Thus by simulating D, S, and T_e, simply set $V = (D + S - T_e)^+$. Equation 22 extends analogously to the FIFO $GI/GI/c$ model, where our objective is to exactly simulate

from the time-stationary distribution of the continuous-time Kiefer-Wolfowitz workload vector, $\mathbf{W}(t) = (W(t;1), \ldots, W(t;c))^T,\ t \geq 0$, where it can be constructed via

$$\mathbf{W}(t) = \mathcal{R}(\mathbf{W}_n + S_n\mathbf{e} - (t - t_n)\mathbf{f})^+,\ t_n \leq t < t_{n+1}.$$

It is to be continuous from the right with left limits; $\mathbf{W}_n = \mathbf{W}(t_n-)$. Total workload $V(t)$, for example, is obtained from this via

$$V(t) = \sum_{i=1}^{c} W(t;i).$$

Letting $\mathbf{W}^*$ have the time-stationary distribution of $\mathbf{W}(t)$ as $t \to \infty$, and letting $\mathbf{W}_0$ have the discrete-time stationary distribution π and letting S, T_e and $\mathbf{W}_0$ be independent, then

$$\mathbf{W}^* \stackrel{d}{=} \mathcal{R}(\mathbf{W}_0 + S\mathbf{e} - T_e\mathbf{f})^+. \tag{23}$$

So once we have a copy of $\mathbf{W}_0$ (distributed as π) from our algorithm in Section 4.1 or Section 4.2, we can easily construct a copy of $\mathbf{W}^*$ as long as we can simulate from A_e. Of course, if arrivals are Poisson then the distribution of $\mathbf{W}^*$ is identical to that of $\mathbf{W}_0$ by PASTA, but otherwise we can use (23).

5.1 Numerical Results

As a sanity check, we have implemented our perfect sampling algorithm in Matlab for the case of $Erlang(k_1, \lambda)/Erlang(k_2, \mu)/c$ queue. We provide our implementation codes for both algorithms in the online appendix of this paper.

Firstly we consider $M/M/c$ queues, which are special cases of $Erlang(k_1, \lambda)/Erlang(k_2, \mu)/c$ with $k_1 = k_2 = 1$. For the quantity of interest, number of customers in the FIFO $M/M/c$ queue at stationary, we obtain its empirical distribution from a large number of independent runs of our algorithm and compare it to the theoretical distribution which has a well-established closed form

$$\pi_0 = \left(\sum_{k=0}^{c-1} \frac{\rho^k}{k!} + \frac{\rho^c}{(c-1)!} \frac{1}{c-\rho} \right)^{-1},$$

$$\pi_k = \begin{cases} \pi_0 \cdot \rho^k/k! & \text{if } 0 < k < c \\ \pi_0 \cdot \rho^k c^{c-k}/c! & \text{if } k \geq c \end{cases},$$

where $\rho = \lambda/\mu < c$.

As an example, Figure 1 shows the result of such test when $\lambda = 3$, $\mu = 2$ and $c = 2$. Grey bars are the empirical results of $5,000$ draws using our algorithm, and black bars are the theoretical distribution number of customers in system from stationarity. A Pearson's chi-squared test between the theoretical and empirical distributions gives a p-value equal to 0.8781, indicating close agreement (i.e., we cannot reject the null-hypothesis that there is no difference between these two distributions). For another set of parameters $\lambda = 10$, $\mu = 2$ and $c = 10$, the results are shown in Figure 2 with a p-value being 0.6069 for the chi-squared fitness test.

For the general $Erlang(k_1, \lambda)/Erlang(k_2, \mu)/c$ queue when $k_1 > 1$ and $k_2 > 1$ when $\rho = \lambda k_2/(c\mu k_1) = 0.9$, we compare the empirical distribution of number of customers in system at stationarity, obtained from a large number of runs of our perfect sampling algorithm, to the numerical results (with precision at least 10^{-4}) provided in Table III of [13]. The results for an

$Erlang(2,9)/Erlang(2,5)/c$ queue are given in Figure 3. Grey bars are the empirical results of $5,000$ draws using our algorithm and black bars are the numerical values given in [13], and they are very close to each other. The Pearson's chi-squared test gives a p-value of 0.9464, thus we cannot reject the null-hypothesis that these two distributions agree well.

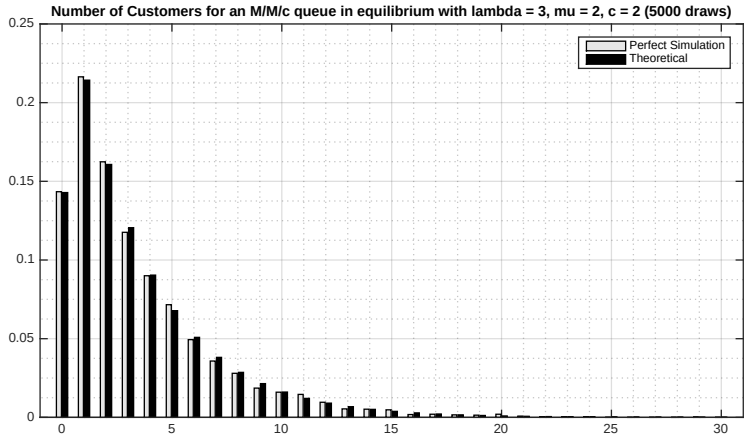

Figure 1: Number of customers for an $M/M/c$ queue in stationarity when $\lambda = 3$, $\mu = 2$, $c = 2$.

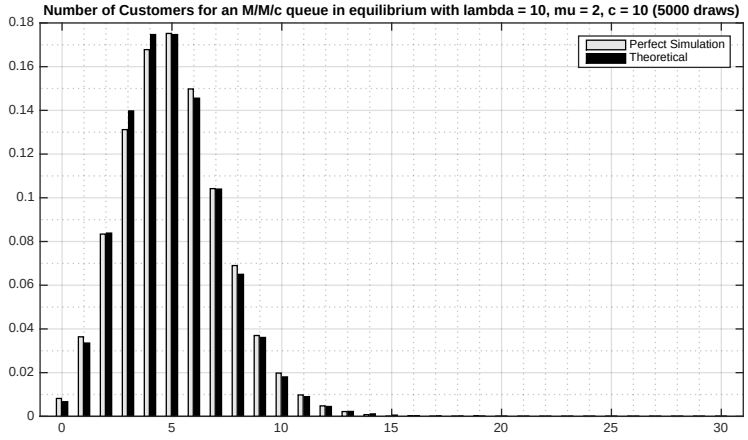

Figure 2: Number of customers for an $M/M/c$ queue in stationarity when $\lambda = 10$, $\mu = 2$, $c = 10$.

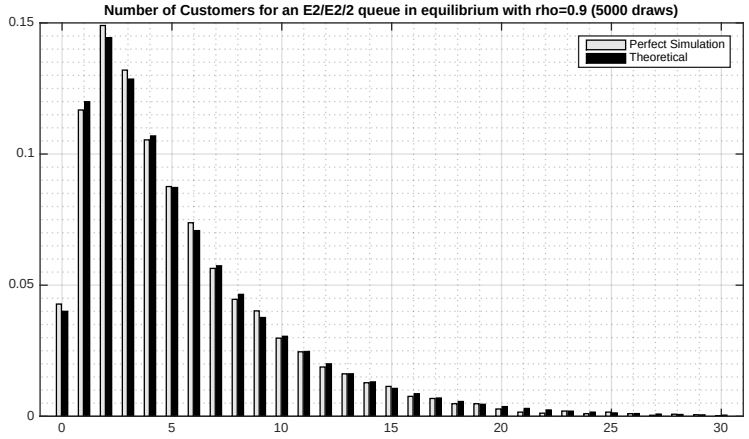

Figure 3: Number of customers for an $Erlang(k_1, \lambda)/Erlang(k_2, \mu)/c$ queue in stationarity when $k_1 = 2$, $\lambda = 9$, $k_2 = 2$, $\mu = 5$, $c = 2$ and $\rho = 0.9$.

Next we run a numerical experiment to compare how far we need to simulate the dominating process backwards to detect coalescence. For the first algorithm in Section 4.1, we let running time $\hat{T} = \sum_{i=1}^{N} T_{-i}$, i.e. the time taken for the queue under RA discipline to empty; and for the second algorithm in Section 4.2, we let running time $\hat{T} = \sum_{i=1}^{\kappa} T_{-i}$, i.e. the time taken for the first successful inspection time in order to detect coalescence before time 0. Figure 4 we give distributions of the time taken for the first time coalescence ever detected under two algorithms for an $M/M/c$ queue with parameters $\lambda = 10$, $\mu = 2$, $c = 10$, from 5000 runs. The result indicates that the second algorithm (sandwiching) performs significantly faster than the first one.

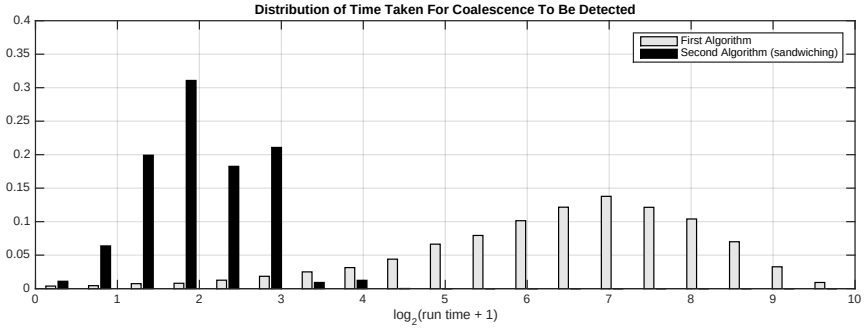

Figure 4: Distributions of time taken to detect coalescence under two algorithms for an $M/M/c$ queue

Finally we compare how computational complexity of our sandwiching algorithm compare to the algorithm given in [6]. Notice these two algorithms do look similar: they both use back-off strategies to run two bounding processes from some inspection time and check if they meet before time 0. The difference is that in [6] they use a so-called "vacation system" to construct upper bound process, whereas we use the same queue but under RA discipline instead. In the following numerical experiment, we define the computational complexity as the total number of arrivals each algorithm samples backwards to detect coalescence. Table 1 shows

how they vary with traffic intensity, $\rho/c = \lambda/(c\mu)$, based on 5000 independent runs of both algorithms using the same back-off strategy with same initial $\kappa = 1$. The result suggests that our second algorithm (sandwiching) outperform the one proposed in [6] as the magnitude of our computational complexity does not increase as fast as theirs when traffic intensity increases.

Table 1: simulation result for computational complexities with varying traffic intensities
$M/M/c$ queue with fixed $\mu = 5$ and $c = 2$

λ	ρ/c	95% confidence interval of number of arrivals simulated backwards	
		Algorithm in Section 4.2	Algorithm in [6]
5	0.5	54.8194 ± 0.5758	146.5618 ± 2.3598
6	0.6	86.5394 ± 1.0536	308.4448 ± 4.9413
7	0.7	152.6552 ± 2.2695	730.1130 ± 11.2783
8	0.8	337.9544 ± 6.3021	2201.8254 ± 32.1556
9	0.9	1521.3502 ± 31.8267	12277.8686 ± 161.5824

6 Why we can assume that interarrival times are bounded

Lemma 6.1 *Consider the recursion*

$$D_{n+1} = (D_n + S_n - T_n)^+, \ n \geq 0, \tag{24}$$

where both $\{T_n\}$ *and* $\{S_n\}$ *are non-negative random variables, and* $D_0 = 0$.

Suppose for another sequence of non-negative random variables $\{\hat{T}_n\}$, *it holds that*

$$P(\hat{T}_n \leq T_n, \ n \geq 0) = 1.$$

Then for the recursion

$$\hat{D}_{n+1} = (\hat{D}_n + S_n - \hat{T}_n)^+, \ n \geq 0, \tag{25}$$

with $\hat{D}_0 = 0$, *it holds that*

$$P(D_n \leq \hat{D}_n, \ n \geq 0) = 1. \tag{26}$$

Proof : The proof is by induction on $n \geq 0$: Because (w.p.1 in the following arguments) $\hat{T}_0 \leq T_0$, we have

$$D_1 = (S_0 - T_0)^+ \leq (S_0 - \hat{T}_0)^+ = \hat{D}_1.$$

Now suppose the result holds for some $n \geq 0$. Then $D_n \leq \hat{D}_n$ and by assumption $\hat{T}_n \leq T_n$; hence

$$D_{n+1} = (D_n + S_n - T_n)^+ \leq (\hat{D}_n + S_n - \hat{T}_n)^+ = \hat{D}_{n+1},$$

and the proof is complete. ■

Proposition 6.1 *Consider the stable RA GI/GI/c model in which $P(T > S) > 0$. In order to use this model to simulate from the corresponding stationary distribution of the FIFO GI/GI/c model as explained in the Section 4.1, without loss of generality we can assume that the interarrival times $\{T_n\}$ are bounded: There exists $b > 0$ such that*

$$P(T_n \leq b,\ n \geq 0) = 1.$$

Proof : By stability, $cE(T) > E(S)$, and by assumption $P(T > S) > 0$. If the $\{T_n\}$ are not bounded, then for $b > 0$, define $\hat{T}_n = \min\{T_n, b\},\ n \geq 0$; truncated T_n. Choose b sufficiently large so that $cE(\hat{T}) > E(S)$ and $P(\hat{T} > S) > 0$ still holds. Now use the $\{\hat{T}_n\}$ in place of the $\{T_n\}$ to construct an RA model, denoted by $\widehat{RA}$. Denote this by

$$\hat{\mathbf{V}}_n = (\hat{V}_n(1), \ldots, \hat{V}_n(c)),$$

where it satisfies the recursion (7) in the form

$$\hat{\mathbf{V}}_{n+1} = (\hat{\mathbf{V}}_n + \tilde{\mathbf{S}}_n - \hat{\mathbf{T}}_n)^+,\ n \geq 0,$$

where $\hat{\mathbf{T}}_n = \hat{T}_n \cdot \mathbf{f}$.

Starting from $\mathbf{V}_0 = \hat{\mathbf{V}}_0 = \mathbf{0}$, then from Lemma 6.1, it holds (coordinate-wise) that

$$\mathbf{V}_n \leq \hat{\mathbf{V}}_n,\ n \geq 0,$$

and thus, if for some $n \geq 0$ it holds that $\hat{\mathbf{V}}_n = \mathbf{0}$, then $\mathbf{V}_n = \mathbf{0}$ and hence $\mathbf{W}_n = \mathbf{0}$ (as explained in our previous section). Since b was chosen ensuring that $cE(\hat{T}) > E(S)$ and $P(\hat{T} > S) > 0$, $\{\hat{\mathbf{V}}_n\}$ is a stable RA $GI/GI/c$ queue that will indeed empty infinitely often. Thus we can use it to do the backwards in discrete-time stationary construction until it empties, at time (say) $-\hat{N}$; $\hat{N} = \min\{n \geq 0 : \hat{\mathbf{V}}_{-n} = 0\}$. Then, we can re-construct the original RA model (starting empty at time $-\hat{N}$) using the (original untruncated) $\hat{N}$ interarrival times $(T_{-\hat{N}}, T_{-\hat{N}+1}, ..., T_{-1})$ in liu of $(\hat{T}_{-\hat{N}}, \hat{T}_{-\hat{N}+1}, ..., \hat{T}_{-1})$, so as to collect $\hat{N}$ re-ordered S_n needed in construction of $\mathbf{W}_0$ for the FIFO model. ■

Remark 6.1 One would expect that the reconstruction of the original RA model in the above proof is unnecessary, that instead we only need to re-construct the $\widehat{RA}$ model until we have $\hat{N}$ service initiations from it, as opposed to $\hat{N}$ service initiations from the original RA model. Although this might be true, the subtle problem is that the order in which service times are initiated in the $\widehat{RA}$ model will typically be different than for the original RA model; they have different arrival processes (counterexamples are easy to construct). Thus it is not clear how one can utilize Lemma 3.1 and Lemma 3.2 and so on. One would need to generalize Lemma 3.1 to account for truncated arrival times used in the RA model, but not the FIFO model, in perhaps a form such as a variation of Equation (3),

$$P(Q_F(t_n-) \leq Q_{\widehat{RA}}(\hat{t}_n-),\ \text{for all } n \geq 0) = 1, \tag{27}$$

where $\{\hat{t}_n\}$ is the truncated renewal process. We did not explore this further.

7 Infinite server systems and other service disciplines

In this Section we sketch how one can utilize our FIFO $GI/GI/c$ results to obtain exact sampling of some other models including the infinite server queue, and the multi-server queue under other disciplines.

In [5] an exact simulation algorithm is presented for simulating from the stationary distribution of the infinite server queue; the $GI/GI/\infty$. Here we sketch how to utilize our new FIFO $GI/GI/c$ results to accomplish this by using a FIFO $GI/GI/c$ model as an upper bound. The $GI/GI/\infty$ model has an infinite number of servers, there is no line, every arrival enters service immediately upon arrival; the n^{th} customer arrives at time t_n and departs at time $t_n + S_n$.

For $0 < \rho = \lambda/\mu < \infty$, this model is always stable. Let c denote the smallest integer strictly larger than ρ; $c - 1 \leq \rho < c$. Letting $V_\infty(t)$ denote the total amount of work in the $GI/GI/\infty$ model, and $V_c(t)$ denote the total amount of work in the (necessarily stable) FIFO $GI/GI/c$ model being fed exactly the same input (of service times and interarrival times), and both starting initially empty, the following is easily established:

$$P(V_\infty(t) \leq V_c(t), \text{ for all } t \geq 0) = 1, \tag{28}$$

hence

$$P(V_\infty(t_n-) \leq V_c(t_n-), \text{ for all } n \geq 0) = 1. \tag{29}$$

(Note that both models use the service times in the same order of initiation, which makes the coupling easy from the start.)

Thus, if, for example $P(T > S) > 0$, then the FIFO model will empty and can be used to detect times when the $GI/GI/\infty$ model will empty. Let $L_\infty(t_n-)$ denote the total number of busy servers in the $GI/GI/\infty$ model as found by C_n.

Simulating the FIFO model backwards in time in stationarity (using our previous algorithm), until it first empties, can then be used to detect a time when the $GI/GI/\infty$ model is empty, and then one can construct it back up to time $t = 0$ to obtain a stationary copy of $V_\infty(t_n-)$ and of $L_\infty(t_n-)$.

Now we consider alternatives disciplines to FIFO for the $GI/GI/c$ model. It is immediate that when service times are generated only when needed by a server, the total number of customers in the system process $\{Q(t)\}$ remains the same under FIFO as under last-in first out (LIFO) in which the next customer to enter service is the one at the bottom of the line, or random selection next (RS) in which the next customer to enter service from the line is selected at random by the server. Thus, they all share the same stationary distribution of $Q(t)$ as $t \to \infty$, as well as the stationary distribution of $Q(t_n-)$ as $n \to \infty$. Let Q_0 have this limiting (as $n \to \infty$) distribution. This fact can be used to exactly simulate, for example, stationary delay D under LIFO or RS (they are not the same as for FIFO). The method (sketch) is as follows: Simulate a copy of Q_0, jointly with the remaining service times of those in service, by assuming FIFO. This represents the distribution of the system as found in stationarity (at time 0) by arrival C_0. Consider RS for example. If the line is empty, then define $D_{RS} = 0$; C_0 enters service immediately. Otherwise, place C_0 in the line, and continue simulating but now using RS instead of FIFO. As soon as C_0 enters service, stop and define D_{RS} as that length of time.

8 Fork-Join Models

The RA recursion (7),

$$\mathbf{V}_{n+1} = (\mathbf{V}_n + \tilde{\mathbf{S}}_n - \mathbf{T}_n)^+, \ n \geq 0, \tag{30}$$

is actually a special case for the modeling of *Fork-Join* (FJ) queues (also called *Split and Match*) with c nodes. In an FJ model, each arrival is a "job" with c components, the i^{th} component

requiring service at the i^{th} FIFO queue. So upon arrival at time t_n, the job splits into its c components to be served. As soon as all c components have completed service, then and only then, does the job depart. Such models are useful in manufacturing applications. The n^{th} job (C_n) thus arrives with a service time vector attached of the form $\mathbf{S}_n = (S_n(1), ..., S_n(c))$. Let us assume that the vectors are i.i.d., but otherwise can be generally jointly distributed; for then (30) still forms a Markov chain. We will denote this model as the $GI/GI/c - FJ$ model. The sojourn time of the i^{th} component is given by $V_n(i) + S_n(i)$, and thus the sojourn time of the n^{th} job, C_n, is given by

$$H_n = \max_{1 \le i \le c} \{V_n(i) + S_n(i)\}. \tag{31}$$

Of great interest is obtaining the limiting distribution of H_n as $n \to \infty$; we denote a r.v. with this distribution as H^0. FJ models are notoriously difficult to analyze analytically: Even the special case of Poisson arrivals and i.i.d. exponential service times is non-trivial because of the dependency of the c queues through the common arrival process. (A classic paper is Flatto [10]). In fact when $c \ge 3$, only bounds and approximations are available. As for exact simulation, there is a paper by Hongsheng Dai [9], in which Poisson arrivals and independent exponential service times are assumed. Because of the continuous-time Markov chain (CTMC) model structure, the author is able to construct (simulate) the time-reversed CTMC to use in a coupling from the past algorithm. But with general renewal arrivals and or general distribution service times, such CTMC methods no longer can be used. Our simulation method for the RA model outlined in Section 4, however (modified to accommodate generally jointly distributed service time vectors under suitable conditions), yields an exact copy of H^0 for the general $GI/GI/c - FJ$ model via first exactly simulating $\mathbf{V}_0^0$ then simulating, independently, a vector of service times $\mathbf{S} = (S(1), ..., S(c))$ and setting

$$H^0 = \max_{1 \le i \le c} \{V_0^0(i) + S(i)\}.$$

Even when the service time components within $\mathbf{S}$ are independent, or the case when service time distributions are assumed to have a finite moment generating function (in a neighborhood of the origin), such results are new and non-trivial.

9 The case when $P(T > S) = 0$: Harris recurrent regeneration

For a stable FIFO $GI/GI/c$ queue, the stability condition can be re-written as $E(T_1 + \cdots + T_c) > E(S)$, which implies also that $P(T_1 + \cdots + T_c > S) > 0$. Thus assuming that $P(T > S) > 0$ is not necessary for stability. When $P(T > S) = 0$, the system will never empty again after starting, and so using consecutive visits to 0 as regeneration points is not possible. But the system does regenerate in a more general way via the use of Harris recurrent Markov chain theory; see [19] for details and history of this approach. The main idea is that while the system will not empty infinitely often, the number in system process $\{Q_F(t_n-) : n \ge 0\}$ will visit an integer $1 \le j \le c-1$ infinitely often.

For illustration here, we will consider the $c = 2$ case (for the general case $c \ge 2$ the specific regeneration points analogous to what we present here are carefully given in Equation (4.6) on page 396 of [19]). Let assume that $1 < \rho < 2$. (Note that if $\rho < 1$, then equivalently $E(T) > E(S)$ and so $P(T > S) > 0$; that is why we rule out $\rho < 1$ here.) We now assume that $P(T > S) = 0$. This implies that for $\underline{s} \triangleq \inf\{s > 0 : P(S > s) > 0\}$ and $\overline{t} \triangleq \sup\{t > 0 : P(T > t) > 0\}$, we must have $0 < \overline{t} < \underline{s} < \infty$. It is shown in [19] that for $\epsilon > 0$ sufficiently small, the following event will happen infinitely often (in n) with probability 1,

$$\{Q_{RA}(t_n-) = 1,\ V_n(1) = 0,\ V_n(2) \le \epsilon,\ T_n > \epsilon,\ U_n = 1\}. \tag{32}$$

If n is such a time, then at time $n+1$, we have

$$\{Q_{RA}(t_{n+1}-) = 1,\ V_{n+1}(2) = 0,\ V_{n+1}(1) = (S_n - T_n) \mid T_n > \epsilon\}. \tag{33}$$

The point is that C_n finds one server (server 1) empty, and the other queue with only one customer in it, and that customer is in service with a remaining service time $\leq \epsilon$. C_n then enters service at node 1 with service time S_n; but since $T_n > \epsilon$, C_{n+1} arrives finding the second queue empty, and the first server has remaining service time $S_n - T_n$ conditional on $T_n > \epsilon$. Under the coupling of Lemma 3.1, the same will be so for the FIFO model (see Remark 9.1 below): At such a time n,

$$\{Q_F(t_n-) = 1,\ W_n(1) = 0,\ W_n(2) \leq \epsilon,\ T_n > \epsilon\}, \tag{34}$$

and at time $n+1$ we have

$$\{Q_F(t_{n+1}-) = 1,\ W_n(1) = 0,\ W_n(2) = (S_n - T_n) \mid T_n > \epsilon\}. \tag{35}$$

Eq (33) and (35) define positive recurrent regeneration points for the two models (at time $n+1$); the consecutive times at which regenerations occur forms a (discrete-time) positive recurrent renewal process.

To put this to use, we change the stopping time N given in (**??**) to:

$$\begin{aligned} N+1 \quad = \quad & \min\{n \geq 1 : Q^0_{RA}(t_{-(n+1)}-) = 1,\ V^0_{-(n+1)}(1) = 0, \\ & V^0_{-(n+1)}(2) \leq \epsilon,\ T_{-(n+1)} > \epsilon,\ U_{-(n+1)} = 1\}. \end{aligned} \tag{36}$$

Then we do our reconstructions for the algorithm in Section 4.1 by starting at time $-N$, with both models starting with the same starting value

$$\{Q_{RA}(t_{-N}-) = 1,\ V^0_{-N}(2) = 0,\ V^0_{-N}(1) = (S_{-(N+1)} - T_{-(N+1)}) \mid T_{-(N+1)} > \epsilon\} \tag{37}$$

$$\{Q_F(t_{-N}-) = 1,\ W_{-N}(1) = 0,\ W_{-N}(2) = (S_{-(N+1)} - T_{-(N+1)}) \mid T_{-(N+1)} > \epsilon\}. \tag{38}$$

Remark 9.1 The service time used in (37) and (38) for coupling via Lemma 3.2, $S_{-(N+1)}$, is in fact identical for both systems because (subtle): At time $-(N+1)$, both systems have only one customer in system, and thus total work is in fact equal to the remaining service time; so we use Equation (5) to conclude that both remaining service times (even if different) are $\leq \epsilon$ (e.g., that is why (34) follow from (32)). Meanwhile, $C_{-(N+1)}$ enters service immediately across both systems, so it is indeed the same service time $S_{-(N+1)}$ used for both for this initiation.

10 Appendix

Now we describe the simulation algorithm in Section 4.1 in detail.

10.1 Simulation algorithm for process $\{\mathbf{Y}_{-n} : n \geq 0\}$ and its running time maximums

We first consider simulating the multidimensional random walk $\{\mathbf{Y}_{-n} : n \geq 0\}$ with $\mathbf{Y}_0 = \mathbf{0}$. Since for all $j \geq 1$, $\mathbb{E}(a\mathbf{U}_{-j} - \mathbf{T}_{-j}) < \mathbf{0}$ and $aU_{-j}(i) - T_{-j} \leq a$ for all $1 \leq i \leq c$, we can simulate the running time maximum $\max_{k \geq n} \mathbf{Y}_{-k}$ jointly with the path $\{\mathbf{Y}_{-k} : 0 \leq k \leq n\}$ via exponential change of measure method developed in [4] with the following assumptions.

Assumption 1: Suppose that in every dimension $i \in \{1, \ldots, c\}$ (they are marginally identical distributed), there exists $\theta^* \in (0, \infty)$ such that

$$\phi_i(\theta^*) \triangleq \log \mathbb{E}_0 \exp\left(\theta^* \left(aU_{-j}(i) - T_{-j}\right)\right) = 0,$$

where $\mathbb{E}_0$ denote expectation under the canonical probability $\mathbb{P}_0$.

Assumption 2: We can choose a constant $m \geq 0$ large enough such that

$$m > \frac{\ln(c)}{\theta^*}. \tag{39}$$

Given a vector $\mathbf{b} \in \mathbb{R}_+^c$, define

$$T_{\mathbf{b}} \triangleq \inf\{n \geq 0 : Y_{-n}(i) > b(i) \ \text{ for some } i\}, \tag{40}$$

$$T_{-\mathbf{b}} \triangleq \inf\{n \geq 0 : Y_{-n}(i) < -b(i) \ \text{ for all } i = 1, \ldots, c\}. \tag{41}$$

Next we define a sequence of upward and downward "milestone" events for this multidimensional random walk. Let $D_0 \triangleq 0$ and $\Gamma_0 \triangleq \infty$. For $k \geq 1$, let

$$D_k \triangleq \inf\{n \geq D_{k-1} \vee \Gamma_{k-1} I\left(\Gamma_{k-1} < \infty\right) : Y_{-n}(i) < Y_{-D_{k-1}}(i) - m \ \text{ for all } i\}, \tag{42}$$

$$\Gamma_k \triangleq \inf\{n \geq D_k : Y_{-n}(i) > Y_{-D_k}(i) + m \ \text{ for some } i\}. \tag{43}$$

Note that by convention, $\Gamma_k I\left(\Gamma_k < \infty\right) = 0$ if $\Gamma_k = \infty$ for any $k \geq 0$. We let $\mathbf{C} \in \mathbb{R}^c$, initially set as $(\infty, \ldots, \infty)^T$, to be the upper bound of $\{\mathbf{Y}_{-n} : n \geq 0\}$ process. From the construction of "milestones" in (42) and (43), we can see that if for some $k \geq 1$ such that $\Gamma_k = \infty$, it means after D_k the process will never cross over the level $\mathbf{Y}_{-D_k} + \mathbf{m}$ coordinate-wise, where $\mathbf{m} = m \cdot \mathbf{f}$ (i.e., $\forall n \geq D_k$, $Y_{-n}(i) \leq Y_{-D_k}(i) + m$ for all $1 \leq i \leq c$). Therefore if $\Gamma_k = \infty$ for some $k \geq 1$, we can update the upper bound vector $\mathbf{C} = \mathbf{Y}_{-D_k} + \mathbf{m}$.

10.1.1 Global maximum of $\{\mathbf{Y}_{-n} : n \geq 0\}$

Define

$$\Delta \triangleq \inf\{D_k : \Gamma_k = \infty, k \geq 1\}, \tag{44}$$

then by the construction of "milestone" events, if $n \geq \Delta$,

$$Y_{-n}(i) \leq Y_{-\Delta}(i) + m < 0 = Y_0(i),$$

for all $i = 1, \ldots, c$. Hence we can evaluate the global maximum of the process $\{\mathbf{Y}_{-n} : n \geq 0\}$ to be

$$\mathbf{M}_0 \triangleq \max_{k \geq 0} \mathbf{Y}_{-k} = \max_{0 \leq k \leq \Delta} \mathbf{Y}_{-k}, \tag{45}$$

and we give the detailed sampling procedure in the following algorithm.

Algorithm 1: Simulate global maximum of $\{\mathbf{Y}_{-n} : n \geq 0\}$ jointly with the sub-path and subsequence of "milestone" events.

1. Set $n = 0$, $\mathbf{Y}_0 = \mathbf{0}$, $\mathbf{D} = [0]$, $\mathbf{\Gamma} = [\infty]$, $\mathbf{L} = \mathbf{0}$.

2. Generate $U \sim Unif\{1, \cdots, c\}$ and let $\mathbf{U} = (I(U = 1), \ldots, I(U = c))^T$. Independently sample $T \sim A$, and let $\mathbf{T} = T \cdot \mathbf{f}$. Set $n = n + 1$, $\mathbf{Y}_{-n} = \mathbf{Y}_{-(n-1)} + a\mathbf{U} - \mathbf{T}$, $U_{-n} = U$ and $T_{-n} = T$.

3. If there is some $1 \leq i \leq c$ such that $Y_{-n}(i) \geq L(i) - m$, then go to Step 2; otherwise set $\mathbf{D} = [\mathbf{D}, n]$ and $\mathbf{L} = \mathbf{Y}_{-n}$.

4. Independently sample $J \sim Ber\left(P_0\left(T_{\mathbf{m}} < \infty\right)\right)$.

5. If $J = 1$, simulate a new conditional path $\{(\mathbf{y}_{-k}, u_{-k}, t_{-k}) : 1 \leq k \leq T_{\mathbf{m}}\}$ with $\mathbf{y}_0 = \mathbf{0}$, following the conditional distribution of $\{\mathbf{Y}_{-k} : 0 \leq k \leq T_{\mathbf{m}}\}$ given $T_{\mathbf{m}} < \infty$. Set $\mathbf{Y}_{-(n+k)} = \mathbf{Y}_{-n} + \mathbf{y}_{-k}$, $U_{-(n+k)} = u_{-k}$, $T_{-(n+k)} = t_{-k}$ for $1 \leq k \leq T_{\mathbf{m}}$. Set $n = n + T_{\mathbf{m}}$, $\mathbf{\Gamma} = [\mathbf{\Gamma}, n]$. Go to Step 2.

6. If $J = 0$, set $\Delta = n$ and $\mathbf{\Gamma} = [\mathbf{\Gamma}, \infty]$.

7. Output $\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 1 \leq k \leq \Delta\}$, $\mathbf{D}$, $\mathbf{\Gamma}$ and global maximum $\mathbf{M}_0 = \max_{0 \leq k \leq \Delta} \mathbf{Y}_{-k}$.

Now we explain how to execute Step 4, 5 and 6 in the previous algorithm. The procedure is similar to the multi-dimensional procedure given in [4], so we describe briefly here.

As $P_0\left(\cdot\right)$ denotes the canonical probability, we let $P_0^*(\cdot) = P_0(\cdot | T_{\mathbf{m}} < \infty)$. Our goal is to simulate from the conditional law of $\{\mathbf{Y}_{-k} : 0 \leq k \leq T_{\mathbf{m}}\}$ given that $T_{\mathbf{m}} < \infty$ and $\mathbf{Y}_0 = \mathbf{0}$, i.e., to simulate from P_0^*. Let $P_0'(\cdot)$ denote the proposal distribution we will use. A typical element ω' sampled under $P_0'\left(\cdot\right)$ is of the form $\omega' = ((\mathbf{Y}_{-k} : k \geq 0), Index)$, where $Index \in \{1, \cdots, c\}$ and it indicates the direction we pick to do exponential tilting. The distribution of ω' induced by $P_0'(\cdot)$ is, firstly,

$$P_0'\left(Index = i\right) = w_i \triangleq \frac{1}{c}. \tag{46}$$

Then conditioning on $Index = i$, for every set $A \in \sigma\left(\{\mathbf{Y}_{-k} : 0 \leq k \leq n\}\right)$,

$$P_0'\left(A | Index = i\right) = \mathbb{E}_0\left(\exp\left(\theta^* Y_{-n}(i)\right) I_A\right).$$

In our case, given $Index = i$, under $P_0'(\cdot)$ we have

$$P_0'(U = j | Index = i) = \begin{cases} \frac{\exp(\theta^* a)}{\exp(\theta^* a) + c - 1} & \text{if } j = i \\ \frac{1}{\exp(\theta^* a) + c - 1} & \text{if } j \neq i \end{cases} \tag{47}$$

and the distribution of interarrival time T is obtained by exponential tilting such that

$$\frac{dP_0'}{dP_0}\left(T\right) = \frac{\theta^* a + c - 1}{c} \exp\left(-\theta^* T\right) \tag{48}$$

Note that since

$$\begin{aligned} \mathbb{E}_0'\left(Y_{-n}(Index)\right) &= \sum_{i=1}^{c} \mathbb{E}_0\left(Y_{-n}(i) \exp\left(\theta^* Y_{-n}(i)\right)\right) P_0'\left(Index = i\right) \\ &= \sum_{i=1}^{c} \frac{d\phi_i\left(\theta^*\right)}{d\theta} w_i > 0, \end{aligned}$$

then $Y_{-n}\,(Index) \to \infty$ as $n \to \infty$ almost surely under $P_0'(\cdot)$, therefore $T_{\mathbf{m}} < \infty$ with probability one under P_0'. This is a valid acceptance / rejection method because

$$\begin{aligned}
&\frac{dP_0^*}{dP_0'}\left(\mathbf{Y}_{-k} : 0 \le k \le T_{\mathbf{m}}\right) \\
=&\frac{1}{P_0\left(T_{\mathbf{m}} < \infty\right)} \times \frac{dP_0}{dP_0'}\left(\mathbf{Y}_{-k} : 0 \le k \le T_{\mathbf{m}}\right) \\
=&\frac{1}{P_0\left(T_{\mathbf{m}} < \infty\right)} \times \frac{1}{\sum_{i=1}^{c} w_i \exp\left(\theta^* Y_{-T_{\mathbf{m}}}(i)\right)} \\
\le&\frac{1}{P_0\left(T_{\mathbf{m}} < \infty\right)} \times \frac{c}{\exp\left(\theta^* m\right)} \\
<&\frac{1}{P_0\left(T_{\mathbf{m}} < \infty\right)},
\end{aligned}$$

where the last inequality is guaranteed by (39) in Assumption 2.

We use the acceptance/rejection method to replace Step 4, 5 and 6 in **Algorithm 1** as follows:

4' Sample $\{(\mathbf{y}_{-k}, u_{-k}, t_{-k}) : 0 \le k \le T_{\mathbf{m}})\}$ with $\mathbf{y}_0 = \mathbf{0}$ from $P_0'(\cdot)$ as indicated via equations (46), (47), (48). Then simulate a Bernoulli J with success probability

$$\frac{1}{\sum_{i=1}^{c} w_i \exp\left(\theta^* y_{-T_{\mathbf{m}}}(i)\right)}.$$

5' If $J = 1$, set $\mathbf{Y}_{-(n+k)} = \mathbf{Y}_{-n} + \mathbf{y}_{-k}$, $U_{-(n+k)} = u_{-k}$, $T_{-(n+k)} = t_{-k}$ for $1 \le k \le T_{\mathbf{m}}$. Set $n = n + T_{\mathbf{m}}$ and $\mathbf{\Gamma} = [\mathbf{\Gamma}, n]$. Go to Step 2.

6' If $J = 0$, set $\mathbf{\Gamma} = [\mathbf{\Gamma}, \infty]$ and $\Delta = n$.

10.1.2 Simulate $\{\mathbf{Y}_{-n} : n \ge 0\}$ with "milestone" events

In this section we provide an algorithm to sequentially simulate the multidimensional random walk $\{\mathbf{Y}_{-n} : n \ge 0\}$ along with its downward and upward "milestone" events. At the same time, we also maintain an index list, defined as

$$\tau^Y \triangleq \mathcal{R}\left\{n \ge 0 : \max_{k \ge n} \mathbf{Y}_{-k} = \mathbf{Y}_{-n}\right\},$$

i.e. set τ^Y records the indices when the process achieves its running time maximums. For any $n \ge 0$, define

$$\begin{aligned}
d_1(n) &\triangleq \inf\{D_j : D_j \ge n, j \ge 0\}, \\
d_2(n) &\triangleq \inf\{D_k : D_k > d_1(n), \Gamma_k = \infty, k \ge 0\}.
\end{aligned}$$

Obviously

$$\mathbf{M}_{d_1(n)} \triangleq \max_{j \ge d_1(n)} \mathbf{Y}_{-j} = \max_{d_1(n) \le j \le d_2(n)} \mathbf{Y}_{-j},$$

so for any $n \le k \le d_1(n)$, if $\mathbf{Y}_{-k} = \max_{k \le j \le d_2(n)} \mathbf{Y}_{-j}$, then we put k into the list τ^Y as the process $\{\mathbf{Y}_{-n} : n \ge 0\}$ achieves its running time maximum at step k.

Lemma 10.1 *Let* $\mathbf{0} < \mathbf{a} < \mathbf{b} \leq \infty \cdot \mathbf{f}$ *(coordinate-wise) and consider any sequence of bounded positive measurable functions* $f_k : \mathbb{R}_{c\times(k+1)} \to [0,\infty)$*,*

$$\begin{aligned}&\mathbb{E}_0\left(f_{T_{-\mathbf{a}}}\left(\mathbf{Y}_0,\cdots,\mathbf{Y}_{-T_{-\mathbf{a}}}\right)|T_{\mathbf{b}}=\infty\right)\\ =&\frac{\mathbb{E}_0\left(f_{T_{-\mathbf{a}}}\left(\mathbf{Y}_0,\cdots,\mathbf{Y}_{-T_{-\mathbf{a}}}\right)I\left\{Y_{-j}(i)\leq b(i),\forall j<T_{-\mathbf{a}},i=1,\ldots,c\right\}P_{\mathbf{Y}_{-T_{-\mathbf{a}}}}\left(T_{\mathbf{b}}=\infty\right)\right)}{P_0\left(T_{\mathbf{b}}=\infty\right)}.\end{aligned}$$

So if $P_0^{**}(\cdot) := P(\cdot|T_{\mathbf{b}}=\infty)$*, then*

$$\frac{dP_0^{**}}{dP_0}=\frac{I\left(Y_{-j}(i)\leq b(i),\forall j<T_{-\mathbf{a}},i=1\ldots,c\right)P_{\mathbf{Y}_{-T_{-\mathbf{a}}}}\left(T_{\mathbf{b}}=\infty\right)}{P_0\left(T_{\mathbf{b}}=\infty\right)}\leq\frac{1}{P_0\left(T_{\mathbf{b}}=\infty\right)}. \tag{49}$$

Lemma 10.1 enables us to sample a downward patch by using the acceptance/rejection method with the nominal distribution as proposal. Suppose our current position is $\mathbf{Y}_{-D_j}$ and we know that the process will never go above the upper bound $\mathbf{C}$ (coordinate-wise). Next we simulate the path up to time D_{j+1}. If we can propose a downward patch $\left(\mathbf{y}_{-1},\cdots,\mathbf{y}_{-T_{-\mathbf{m}}}\right) := \left(\mathbf{Y}_{-1},\cdots,\mathbf{Y}_{-T_{-\mathbf{m}}}\right)$, under the unconditional probability given $\mathbf{y}_0 = \mathbf{0}$ and $\mathbf{y}_{-k} \leq \mathbf{m}$ for $1 \leq k \leq T_{-\mathbf{m}}$, then we accept the downward patch with probability $P_0(T_\sigma = \infty)$, where $\sigma = \mathbf{C} - \mathbf{Y}_{-D_j} - \mathbf{y}_{-T_{-\mathbf{m}}}$. A more efficient way to sample would be to sequentially generate $(\mathbf{y}_{-1},\cdots,\mathbf{y}_{-\Delta})$ with $\mathbf{y}_0 = \mathbf{0}$ as long as $\mathbf{M}_0 = \max_{0\leq k\leq\Delta}\mathbf{y}_{-k} \leq \mathbf{m}$ coordinate-wise, and then concatenate these sub-paths together. We give the efficient implementation procedure in the next algorithm.

Algorithm 2: Continue sample partial set τ^Y, partial "milestone" event lists $\mathbf{D}$ and $\mathbf{\Gamma}$ jointly with the process $\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 0 \leq k \leq n\}$ until $f(\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 0 \leq k \leq n\}, \tau^Y) \geq K$.

Input: previously sampled partial process $\{(\mathbf{Y}_{-j}, U_{-j}, T_{-j}) : 0 \leq j \leq l\}$, partial "milestone" sequences $\mathbf{D}$ and $\mathbf{\Gamma}$, and partial index set τ^Y. Function f and value K.

1. Set $n = l$. If $n = 0$, call **Algorithm 1** to get Δ, $\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 0 \leq k \leq \Delta\}$, $\mathbf{D}$, $\mathbf{\Gamma}$. Set $n = \Delta$, and $\tau^Y = \mathcal{R}\{k : 0 \leq k \leq \mathbf{D}(end-1), \max_{k\leq j\leq n}\mathbf{Y}_{-j} = \mathbf{Y}_{-k}\}$.

2. If $f(\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 0 \leq k \leq n\}, \tau^Y) \geq K$, stop and output $\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 0 \leq k \leq n\}$ with updated $\mathbf{D}$, $\mathbf{\Gamma}$, τ^Y. Otherwise go to Step 3.

3. Call **Algorithm 1** to get $\tilde{\Delta}$, $\left\{\left(\tilde{\mathbf{Y}}_{-j}, \tilde{U}_{-j}, \tilde{T}_{-j}\right) : 0 \leq j \leq \tilde{\Delta}\right\}$, $\tilde{\mathbf{D}}$, $\tilde{\mathbf{\Gamma}}$ and $\tilde{\mathbf{M}}_0$.

4. If $\tilde{\mathbf{M}}_0 \leq \mathbf{m}$ coordinate-wise, then accept and concatenate it to the previous sub-path, i.e. set $\mathbf{Y}_{-(n+j)} = \mathbf{Y}_{-n} + \tilde{\mathbf{Y}}_{-j}$, $U_{-(n+j)} = U_{-n} + \tilde{U}_{-j}$, $T_{-(n+j)} = T_{-n} + \tilde{T}_{-j}$ for $0 \leq j \leq \tilde{\Delta}$. Update the sequences of "milestone" events to be $\mathbf{D} = [\mathbf{D}, n + \tilde{\mathbf{D}}]$, $\mathbf{\Gamma} = [\mathbf{\Gamma}, n + \tilde{\mathbf{\Gamma}}]$ and set $n = n + \tilde{\Delta}$. Set $\tau^Y = [\tau^Y, \mathcal{R}\{k : \tau^Y(end) + 1 \leq k \leq \mathbf{D}(end-1), \max_{k\leq j\leq n}\mathbf{Y}_{-j} = \mathbf{Y}_{-k}\})]$. Go to Step 2.

5. Else go to Step 3.

10.2 Simulation algorithm for the process $\{\mathbf{X}_{-n} : n \geq 0\}$ and its running time maximums

As we sample the partial process $\{\mathbf{Y}_{-k} : 0 \leq k \leq n\}$, we get the partial sequence $\{U_{-k} : 0 \leq k \leq n\}$ along with it. Conditioning on that information, we can separate the multi-dimensional process $\{\mathbf{X}_{-k} : k \geq 0\}$ to be c independent processes $\{X_{-k}(i) : k \geq 0\}$, $i = 1, \ldots, c$.

For each $1 \leq i \leq c$ and $a, b \in \mathbb{N}_+$ with $a < b$, let

$$N_i^{(r)}(a, b) \triangleq \sum_{j=a}^{b} I\{U_{-j} = i\} \tag{50}$$

to denotes the total number of times that a customer is routed to server i from the $(-a)^{th}$ arrival time epoch to the $(-b)^{th}$ arrival time epoch backwards in time. Let $R_i(j) \triangleq \inf\{n \geq 1 : N_i^{(r)}(1, n) = j\}$ and $\tilde{S}_{-j}^{(i)} \triangleq S_{-R_i(j)}$ denote the arrival index and the service time of the j-th customer that goes to server i, counting backwards in time from 0, hence

$$\max_{k \geq n} X_{-k}(i) = \max_{k \geq n} \sum_{j=1}^{k} (S_{-j} - a)\, I\{U_{-j} = i\} = \max_{k \geq n} \sum_{j=1}^{N_i^{(r)}(n,k)} \left(\tilde{S}_{-j}^{(i)} - a\right). \tag{51}$$

Let $\{\tilde{X}_{-n}^{(i)} : n \geq 0\}$ with $\tilde{X}_0^{(i)} = 0$ be an auxiliary process that corresponds to the process $\{X_{-n}(i) : n \geq 0\}$ for each $i = 1, \ldots, c$ such that

$$\tilde{X}_{-n}^{(i)} \triangleq \sum_{j=1}^{n} \left(\tilde{S}_{-j}^{(i)} - a\right)$$

for $n \geq 0$. Note that the correspondence between $\{\tilde{X}_{-n^{(i)}}^{(i)} : n^{(i)} \geq 0\}$ and $\{X_{-n}(i) : n \geq 0\}$ is, for any $n^{(i)} \in \mathbb{N}$,

$$\tilde{X}_{-n}^{(i)} = X_{-R_i(n)}(i) = X_{-(R_i(n)+1)}(i) = \cdots = X_{-(R_i(n+1)-1)}(i). \tag{52}$$

Next for each one dimensional auxiliary process $\{\tilde{X}_{-n}^{(i)} : n \geq 0\}$, we adopt the algorithm developed in [7] by choosing $m' > 0$ and $L' \geq 1$ properly and define the sequences of upward and downward "milestone" events by letting $D_0^{(i)} \triangleq 0$, $\Gamma_0^{(i)} \triangleq \infty$, and for $j \geq 1$,

$$D_j^{(i)} \triangleq \inf\left\{n^{(i)} \geq \Gamma_{j-1}^{(i)} I\left\{\Gamma_{j-1}^{(i)} < \infty\right\} \vee D_{j-1}^{(i)} : \tilde{X}_{-n^{(i)}}^{(i)} < \tilde{X}_{-D_{j-1}^{(i)}}^{(i)} - L'm'\right\}, \tag{53}$$

$$\Gamma_j^{(i)} \triangleq \inf\left\{n^{(i)} \geq D_j^{(i)} : \tilde{X}_{-n^{(i)}}^{(i)} - \tilde{X}_{-D_j^{(i)}}^{(i)} > m'\right\}, \tag{54}$$

with the convention that if $\Gamma_j^{(i)} = \infty$, then $\Gamma_j^{(i)} I\left\{\Gamma_j^{(i)} < \infty\right\} = 0$ for any $j \geq 0$.

Define

$$\tilde{\tau}_i^X \triangleq \mathcal{R}\left\{n \geq 0 : \max_{k \geq n} \tilde{X}_{-k}^{(i)} = \tilde{X}_{-n}^{(i)}\right\}, \tag{55}$$

and for any $n \geq 0$, let

$$d_1^{(i)}(n) \triangleq \inf\{D_j^{(i)} : D_j^{(i)} \geq n, j \geq 0\},$$
$$d_2^{(i)}(n) \triangleq \inf\{D_j^{(i)} : D_j^{(i)} > d_1^{(i)}(n), \Gamma_j^{(i)} = \infty, j \geq 0\}.$$

Similar as the argument in the previous section, it's easy to check that

$$\tilde{M}_{d_1^{(i)}(n)} \triangleq \max_{j \geq d_1^{(i)}(n)} \tilde{X}_{-j}^{(i)} = \max_{d_1^{(i)}(n) \leq j \leq d_2^{(i)}(n)} \tilde{X}_{-j}^{(i)},$$

hence for any $n \leq k \leq d_1^{(i)}(n)$, if $\tilde{X}_{-k}^{(i)} = \max_{k \leq j \leq d_2^{(i)}(n)} \tilde{X}_{-j}^{(i)}$, then the auxiliary process $\{\tilde{X}_{-n}^{(i)} : n \geq 0\}$ achieves its running time maximum at step k, thus we insert k into $\tilde{\tau}_i^X$.

The following algorithm gives the the sampling procedure for each auxiliary process with a given $i \in \{1, \ldots, c\}$.

Algorithm 3: Continue sample partial list $\tilde{\tau}_i^X$, partial "milestone" event lists $\mathbf{D}^{(i)}$ and $\mathbf{\Gamma}^{(i)}$ jointly with the process $\{(\tilde{X}_{-n}^{(i)}, \tilde{S}_{-n}^{(i)}) : 0 \leq n \leq n_i\}$ until $f(\{(\tilde{X}_{-n}^{(i)}, \tilde{S}_{-n}^{(i)}) : 0 \leq n \leq n_i\}, \tilde{\tau}_i^X) \geq K$.

Input: Previously sampled partial process $\{(\tilde{X}_{-n}^{(i)}, \tilde{S}_{-n}^{(i)}) : 0 \leq n \leq l_i\}$, partial index set $\tilde{\tau}_i^X$, partial "milestone" event sequences $\mathbf{D}$ and $\mathbf{\Gamma}$. Function f and value K.

Simulation steps are the same as **Algorithm 3** in [7]. Construction of set $\tilde{\tau}_i^X$ is the same as in our **Algorithm 2** discussed in the previous section.

10.3 Simulation algorithm for $\{\mathbf{R}_n^{(r)} : n \geq 0\}$ and coalescence detection

In this section we combine the previous simulation algorithms for processes $\{\tilde{X}_{-n}^{(i)} : n \geq 0\}_{i=1}^c$ and $\{\mathbf{Y}_{-n} : n \geq 0\}$ together to exactly simulate the multi-dimensional random walk of interest.

The idea is that for each $i \in \{1, \ldots, c\}$, given previously sampled sub-path of $\{\tilde{X}_{-k}^{(i)} : 0 \leq k \leq l_i\}$, we sample the process $\{\tilde{X}_{-k}^{(i)} : -0 \leq k \leq n_i\}$ with partial "milestone" event sequences $\mathbf{D}^{(i)}$ and $\mathbf{\Gamma}^{(i)}$, as well as partial index set $\tilde{\tau}_i^X$, until $n_i > l_i$. Since $n_i = \mathbf{D}^{(i)}(end)$ and $\mathbf{\Gamma}^{(i)} = \infty$ when the simulation algorithm stops, we have

$$\tilde{\tau}_i^X = \mathcal{R}\left\{0 \leq k \leq D^{(i)}(end-1) : \tilde{X}_{-k}^{(i)} = \max_{j \geq k} \tilde{X}_{-j}^{(i)}\right\}.$$

Then given the previously sampled sub-path of $\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 0 \leq n \leq l\}$, we continue sample the process $\{\mathbf{Y}_{-k} : 0 \leq k \leq n\}$, jointly with partial "milestone" event sequences $\mathbf{D}$ and $\mathbf{\Gamma}$ and partial index set τ^Y, until $n \geq \sum_{i=1}^c n_i$, i.e. enough U_{-k}'s have been simulated to determine all the auxiliary service times $\{\{\tilde{S}_{-j}^{(i)} : 0 \leq j \leq n_i\}, 1 \leq i \leq c\}$ in the correct order of service initiations in the multi-dimensional system.

Finally for each $i \in \{1, \ldots, c\}$, we use the correspondence relationship in (52) to get a partial index set of running time maximums of process $\{X_{-k}(i) : 0 \leq k \leq n\}$, i.e.,

$$\tau_i^X \triangleq \mathcal{R}\left(\cup_{k \in \tilde{\tau}_i^X}\{R_i(k), R_i(k)+1, \cdots, R_i(k+1)-1\}\right). \tag{56}$$

If $\tau^Y \cap (\cap_{i=1}^c \tau_i^X) \neq \emptyset$, the minimum value of the non-empty set gives a stopping time n^* as defined in (17) and we stop. Otherwise we continue to sample the processes backwards in time until n^* is found.

Algorithm 4: Sample coalescence time N jointly with processes $\{\mathbf{X}_{-n} : n \geq 0\}$ and $\{\mathbf{Y}_n : n \geq 0\}$.

1. Initialize $\mathbf{Y}_0 = \mathbf{0}$, $\mathbf{D} = [0]$, $\mathbf{\Gamma} = [\infty]$, $\tau^Y = []$ and $l = 0$; for each $i = 1, \ldots, c$, $\tilde{X}_0^{(i)} = 0$, $\mathbf{D}^{(i)} = [0]$, $\mathbf{\Gamma}^{(i)} = [\infty]$, $\tilde{\tau}_i^X = []$, $l_i = 0$.
2. For $i = 1, \ldots, c$, call **Algorithm 3** with $f = I(n_i > l_i)$ and $K = 1$ to get updated partial path $\{\tilde{X}_k^{(i)} : 0 \leq k \leq n_i\}$, partial index set $\tilde{\tau}_i^X$ and partial "milestone" event sequences $\mathbf{D}^{(i)}$ and $\mathbf{\Gamma}^{(i)}$.
3. Call **Algorithm 2** with $f = I(n \geq \sum_{i=1}^c n_i)$ and $K = 1$ to get updated partial path $\{(\mathbf{Y}_{-k}, U_{-k}, T_{-k}) : 0 \leq k \leq n\}$, partial index set τ^Y and partial "milestone" event sequence $\mathbf{D}$ and $\mathbf{\Gamma}$.
4. For $i = 1, \ldots, c$, set τ_i^X as indicated in (56).
5. If $\tau^Y \cap (\cap_{i=1}^c \tau_i^X) = \emptyset$, set $l = n$ and $l_i = n_i$ for $1 \leq i \leq c$, then go to Step 2.
6. Otherwise, set $N = \min\{k : k \in \tau^Y \cap (\cap_{i=1}^c \tau_i^X)\}$. Output $\{(U_{-k}, T_{-k}, S_{-k}) : 0 \leq k \leq N\}$ and N.

Acknowledgement: Support from NSF through grant CMMI-1538217 is gratefully acknowledged.

References

[1] S. Asmussen (2003). *Applied Probability and Queues* (2nd Ed.). Springer-Verlag, New York.

[2] S. Asmussen, P. W. Glynn (2007). *Stochastic Simulation*, Springer-Verlag, New York.

[3] S. Asmussen, P. W. Glynn, & H. Thorisson (1992). Stationary detection in the initial transient problem. *ACM TOMACS*, **2**, 130-157.

[4] J. Blanchet and X. Chen (2015). Steady-state simulation of reflected Brownian motion and related stochastic networks. *The Annals of Applied Probability*, **25**, 3209–3250.

[5] J. Blanchet and J. Dong (2015). Perfect sampling for infinite server and loss systems. *Advances in Applied Probability*, **47(3)**, pp 761-786.

[6] J. Blanchet, J. Dong and Y. Pei (2016). Perfect sampling of $GI/GI/c$ queues. *Manuscript submitted for publication.*

[7] J. Blanchet and A. Wallwater (2015). Exact sampling of stationary and time-reversed queues. *ACM TOMACS*, **25**, 26.

[8] S. B. Connor and W. S. Kendall (2015). Perfect simulation of M/G/c queues. *Advances in Applied Probability*, **47**, 4.

[9] H. Dai (2011). Exact Monte Carlo simulation for fork-join networks. *Advances in Applied Prob.*, **43**, 484-503.

[10] L. Flatto and S. Hahn (1984). Two parallel queues created by arrivals with two demands. *SIAM J. Appl. Math.*, **44**, 1041-1053.

[11] S. G. Foss (1980). Approximation of mutichannel queueing systems. *Siberian Math. J.*, **21**, 132-40.

[12] S. G. Foss and N. I. Chernova (2001). On optimality of the FCFS discipline in mutiserver queueing systems and networks. *Siberian Math. J.*, **42**, 372-385.

[13] F. S. Hillier and F. D. Lo (1971). Tables for mutiple-server queueing systems involving erlang distributions. *Technical Report, Department of Operations Research, Stanford University*, **31**.

[14] W. S. Kendall (1998). Perfect simulation using dominated processes on ordered spaces, with application to locally stable point processes. *Probability towards 2000*, 218-234. Springer.

[15] J. G. Propp and D.G. Wilson (1996) Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, **9**, 223-253.

[16] K. Sigman (2011). Exact simulation of the stationary distribution of the FIFO M/G/c queue. *Journal of Applied Probability.* **Special Volume 48A: New Frontiers in Applied Probability**, 209-216.

[17] K. Sigman (2011). Exact simulation of the stationary distribution of the FIFO M/G/c queue: The general case of $\rho < c$. *Queueing Systems.* **70**, 37-43.

[18] K. Sigman (1995). *Stationary Marked Point Processes: An Intuitive Approach*, Chapman and Hall, New York.

[19] K. Sigman (1988). Regeneration in Tandem Queues with Multi-Server Stations. *Journal of Applied Probability*, **25**, 391-403.

[20] R.W. Wolff (1989). *Stochastic Modeling and the Theory of Queues.* Prentice Hall, New Jersey.

[21] R.W. Wolff (1987). Upper bounds on work in system for multi-channel queues. *J. Appl. Probab.* **14**, 547-551.

[22] R.W. Wolff (1982). Poisson Arrivals See Time Averages. *Operations Research.* **30**, 223-231.