Ann Oper Res (2013) 209:67–84
DOI 10.1007/s10479-011-1021-1

Approximate analysis of single-server tandem queues with finite buffers

Remco Bierbooms · Ivo J.B.F. Adan · Marcel van Vuuren

Published online: 16 November 2011

Abstract In this paper, we study single-server tandem queues with general service times and finite buffers. Jobs are served according to the Blocking-After-Service protocol. To approximately determine the throughput and mean sojourn time, we decompose the tandem queue into single-buffer subsystems, the service times of which include starvation and blocking, and then we iteratively estimate the unknown parameters of the service times of each subsystem. The crucial feature of this approach is that in each subsystem successive service times are no longer assumed to be independent, but a successful attempt is made to include dependencies due to blocking by employing the concept of Markovian Arrival Processes. An extensive numerical study shows that this approach produces very accurate estimates for the throughput and mean sojourn time, outperforming existing methods, especially for longer tandem queues and for tandem queues with service times with a high variability.

Keywords Blocking · Decomposition · Finite buffer · Flow line · Markovian arrival process · Matrix-analytic methods

1 Introduction

The subject of this paper is the approximative analysis of single-server tandem queues with general service times and finite buffers. The blocking protocol is Blocking-After-Service (BAS): if the downstream buffer is full upon service completion the server is blocked and has to wait until space becomes available before starting to serve the next job (if there is any).

R. Bierbooms · I.J.B.F. Adan (✉)
Eindhoven University of Technology, P.O. Box 513, 5600 MB, Eindhoven, The Netherlands
e-mail: i.j.b.f.adan@tue.nl

R. Bierbooms
e-mail: r.bierbooms@tue.nl

M. van Vuuren
CQM B.V., P.O. Box 414, 5600 AK, Eindhoven, The Netherlands
e-mail: vanvuuren@cqm.nl

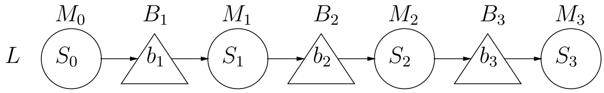

Fig. 1 A tandem queue with 4 servers

Networks of queues (and in particular, tandem queues) with blocking, have been extensively investigated in the literature; see e.g. Buzacott et al. (1995), Colledani and Tolio (2011), Dallery and Gershwin (1992), Perros and Altiok (1986), Perros (1989, 1994). In most cases, however, queueing networks with finite buffers are analytically intractable and therefore the majority of the literature is devoted to approximate analytical investigations. The approximation developed in this paper is based on decomposition, following the pioneering work of Gershwin (1987): the tandem queue is decomposed into single-buffer subsystems, the parameters of which are determined iteratively. In each subsystem, the "actual" service time, starvation and blocking are aggregated in a single service time, and these aggregate service times are typically assumed to be successively independent. However, these aggregate service times are not independent. For instance, knowledge that the server is blocked after service completion (resulting into a long aggregate service time) makes it more likely that the server will also be blocked after the next service. Especially in longer tandem queues with small buffers and in tandem queues with service times with high variability, dependencies of successive aggregate service times may have a strong impact on the performance. In this paper, an approach is proposed to include such dependencies in the aggregate service times.

The model considered in the current paper is a tandem queue L consisting of N servers and $N-1$ buffers in between. The servers (or machines) are labeled M_i, $i = 0, 1, \ldots, N-1$. The first server M_0 acts as a source for the tandem queue, i.e., there is always a new job available for servicing. The service times of server M_i are independent and identically distributed, and they are also independent of the service times of the other servers; S_i denotes the generic service time of server M_i, with rate μ_i and squared coefficient of variation $c^2_{S_i}$. The buffers are labeled B_i and the size of buffer B_i is b_i (i.e., b_i jobs can be stored in B_i). We assume that each server employs the BAS blocking protocol. An example of a tandem queue with 4 machines is illustrated in Fig. 1.

The approximation is based on decomposition of the tandem queue into subsystems, each one consisting of a single buffer. To take into account the relation of buffer B_i with the upstream and downstream part of the tandem queue, the service times of the server in front of buffer B_i and the one after buffer B_i are adapted by aggregating the "real" service times S_{i-1} and possible starvation of M_{i-1} before service, and S_i and possible blocking of M_i after service. The aggregate service processes of M_{i-1} and M_i are described by employing the concept of Markovian Arrival Processes (MAPs; see e.g. Neuts 1989), the parameters of which are determined iteratively. It is important to note that Markovian Arrival Processes can be used to describe dependencies between *successive* service times. Although decomposition techniques for single-server queueing networks have also been widely used in the literature, see e.g. Gershwin (1987), Helber (2005), Kerbache and MacGregor Smith (1987), Perros (1994), van Vuuren et al. (2005), van Vuuren and Adan (2009), the distinguishing feature of the current approximation is the inclusion of dependencies between successive (aggregate) service times by employing Markovian Arrival Processes.

The paper is organized as follows. In Sect. 2 we describe the decomposition of the tandem queue in subsystems. Section 3 presents the iterative algorithm. The service processes of each subsystem are explained in detail in Sects. 4 and 5, after which the subsystem is analyzed in Sect. 6. Numerical results can be found in Sect. 7 and they are compared to simulation and other approximation methods. Finally, Sect. 8 contains some concluding remarks and gives suggestions for further research.

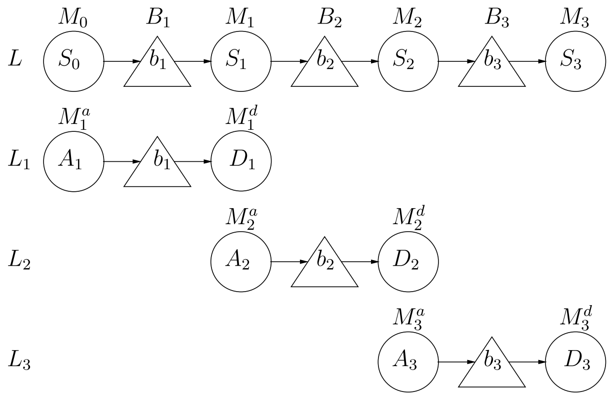

Fig. 2 Decomposition of the tandem queue of Fig. 1 into 3 subsystems

2 Decomposition

The original tandem queue L is decomposed into $N-1$ subsystems $L_1, L_2, \dots, L_{N-1}$. Subsystem L_i consists of buffer B_i of size b_i, an arrival server M_i^a in front of the buffer, and a departure server M_i^d after the buffer. Figure 2 displays the decomposition of line L of Fig. 1.

The arrival server M_i^a of subsystem L_i is, of course, server M_{i-1}, but to account for the connection with the upstream part of L, its service times are different from S_{i-1}. The random variable A_i denotes the service time of the arrival server M_i^a in subsystem L_i. This random variable aggregates S_{i-1} and possible starvation of M_{i-1} before service because of an empty upstream buffer B_{i-1}. Accordingly, the random variable D_i represents the service time of the departure server M_i^d in subsystem L_i; it aggregates S_i and possible blocking of M_i after service completion, because the downstream buffer B_{i+1} is full. Note that successive service times D_i of departure server M_i^d are *not independent*: a long D_i induced by blocking is more likely to be followed by again a long one. The same holds for long service times A_i induced by starvation. We try to include dependencies between successive aggregate service times in the modeling of D_i, but they will be ignored in A_i. The reason for modeling A_i and D_i differently is that starvation occurs before the service start and blocking after service completion, so there is an "asymmetry" in the available information at the end of A_i and D_i, respectively. In the subsequent sections we construct an algorithm to iteratively determine the characteristics of A_i and D_i for each $i = 1, \dots, N-1$.

3 Iterative method

This section is devoted to the description of the iterative algorithm to approximate the performance of tandem queue L. The algorithm is based on decomposition of L in $N-1$ subsystems $L_1, L_2, \dots, L_{N-1}$ as explained in the previous section.

Step 0: Initialization
The first step of the algorithm is to initially assume that there is no blocking. This means that the random variables D_i are initially assumed to be equal to S_i.

Step 1: Evaluation of subsystems
We subsequently evaluate each subsystem, starting from L_1 and up to L_{N-1}. First we determine new estimates for the first two moments of A_i, before calculating the equilibrium distribution of L_i.

(a) Service process of the arrival server
For the first subsystem L_1, the service time A_1 is equal to S_0, because server M_0 cannot be starved. For the other subsystems we proceed as follows in order to determine the first two moments of A_i. Define p_{i,b_i+2} as the long-run fraction of time arrival server M_i^a of subsystem L_i is blocked, i.e., buffer B_i is full, M_i^d is busy, and M_i^a has completed service and is waiting to move the completed job into B_i. By Little's law we have for the throughput T_i of subsystem L_i,

$$T_i = \frac{1 - p_{i,b_i+2}}{\mathbb{E}[A_i]}. \tag{1}$$

By substituting in (1) the estimate $T_{i-1}^{(k)}$ for T_i, which is the principle of conservation of flow, and $p_{i,b_i+2}^{(k-1)}$ for p_{i,b_i+2} we get as new estimate for $\mathbb{E}[A_i]$,

$$\mathbb{E}\big[A_i^{(k)}\big] = \frac{1 - p_{i,b_i+2}^{(k-1)}}{T_{i-1}^{(k)}}, \tag{2}$$

where the superscripts indicate in which iteration the quantities have been calculated. The second moment of A_i cannot be obtained by using Little's law. Instead we calculate the second moment using (3).

(b) Analysis of subsystem L_i
Based on the new estimates for the first two moments of A_i, we translate subsystem L_i to a Markov process and calculate its steady-state distribution as described in Sect. 6.

(c) Determination of the throughput of L_i
Once the steady-state distribution is known, we determine the new throughput $T_i^{(k)}$ according to (7).

Step 2: Service process of the departure server
From subsystem L_{N-2} down to L_1, we adjust the parameters to construct the distribution of D_i, as will be explained in Sect. 5. Note that $D_{N-1} = S_{N-1}$, because server M_{N-1} can never be blocked.

Step 3: Convergence
After Steps 1 and 2 we verify whether the iterative algorithm has converged or not by comparing the throughputs in the $(k-1)$-th and k-th iteration. When

$$\sum_{i=1}^{N-1} \big|T_i^{(k)} - T_i^{(k-1)}\big| < \varepsilon,$$

we stop and otherwise repeat Steps 1 and 2.

4 Service process of the arrival server

In this section, we model the service process of arrival server M_i^a of subsystem L_i (cf. Step 1(a) in Sect. 3). As an approximation, we act as if the service times A_i are independent and identically distributed, thus ignoring dependencies between successive service times A_i.

Note that an arrival in buffer B_i, i.e., a job being served by M_i^a moves to buffer B_i when space becomes available, corresponds to a departure from M_{i-1}^d in the upstream subsystem

L_{i-1}. Just after this departure, two situations may occur: subsystem L_{i-1} is empty with probability (w.p.) q^e_{i-1}, or it is not empty with probability $1-q^e_{i-1}$. By convention, we do not count the job at M^a_{i-1} as being in L_{i-1}. So subsystem L_{i-1} is empty whenever there are no jobs in B_{i-1} and M^d_{i-1}. In the former situation, M_{i-1} has to wait for a residual service time of arrival server M^a_{i-2} of subsystem L_{i-1}, denoted as RA_{i-1}, before the actual service S_{i-1} can start. In the latter situation, the actual service S_{i-1} can start immediately. Hence, since the service time A_i of arrival server M^a_i includes possible starvation of M_{i-1} before the actual service S_{i-1}, we have

$$A_i = \begin{cases} RA_{i-1} + S_{i-1} & \text{with probability } q^e_{i-1}, \\ S_{i-1} & \text{otherwise.} \end{cases}$$

This representation is used to determine the second moment of A_i. Based on q^e_{i-1} and the first two moments of RA_{i-1}, the determination of which is deferred to Sect. 6 (cf. (8)), we obtain the second moment $\mathbb{E}[A_i^2]$ as

$$\mathbb{E}\big[A_i^2\big] = q^e_{i-1}\mathbb{E}\big[RA^2_{i-1}\big] + 2q^e_{i-1}\mathbb{E}[RA_{i-1}]\mathbb{E}[S_{i-1}] + \mathbb{E}\big[S^2_{i-1}\big]. \tag{3}$$

The first moment $\mathbb{E}(A_i)$ follows from (2) expressing conservation of flow.

5 Service process of the departure server

In this section, we describe the service process of the departure server M^d_i of subsystem L_i in detail (cf. Step 2 in Sect. 3). To describe D_i we take into account the occupation of the last position in buffer B_{i+1} (or server M^d_{i+1} if $b_{i+1} = 0$). A job served by M^d_i may encounter three situations in downstream subsystem L_{i+1} on departure from L_i, or equivalently, *on arrival at* L_{i+1}; see Fig. 3. The situation encountered on arrival has implications for possible blocking of the next job served by M^d_i, as will be explained below.

(i) The arrival is triggered by a service completion of departure server M^d_{i+1} of L_{i+1}, i.e., server M^a_{i+1} was *blocked* because the last position in B_{i+1} was occupied, and waiting for M^d_{i+1} to complete service. Then the next service of M^d_i (if there is a job) and M^d_{i+1} start simultaneously and buffer B_{i+1} is full. We denote the time elapsing till the next service completion of departure server M^d_{i+1} by D^b_{i+1}, which is equal to the time the last position in B_{i+1} will be occupied before it becomes available again. Hence, in this situation, the next service time D_i of M^d_i is equal to the maximum of S_i and D^b_{i+1}, if M^d_i can immediately start with the next service. Otherwise, if M^d_i is starved just after the departure, D_i is equal to the maximum of S_i and the *residual time* of D^b_{i+1} at the service start of M^d_i.

(ii) Just before the arrival there is only one position left in buffer B_{i+1}. So, right after this arrival, B_{i+1} is full. Now we denote the time elapsing till the next service completion of departure server M^d_{i+1} by D^f_{i+1}, which is again the time the last position in B_{i+1} will stay occupied. Thus D_i is equal to the maximum of S_i and the residual time of D^f_{i+1} at the service start of M^d_i.

(iii) Finally, when neither of the above situations occurs, the arrival does not fill up buffer B_{i+1}, because there are at least two positions available in B_{i+1}. Hence, the last position in B_{i+1} stays empty and the next service time D_i is equal to S_i.

Note that only in situation (i) and (ii) the next job to be served by M^d_i can be possibly blocked at completion of S_i. If a departure from L_i encounters situation (i), (ii), or (iii) in L_{i+1}, then what is the probability that the next departure from L_i encounters one of these

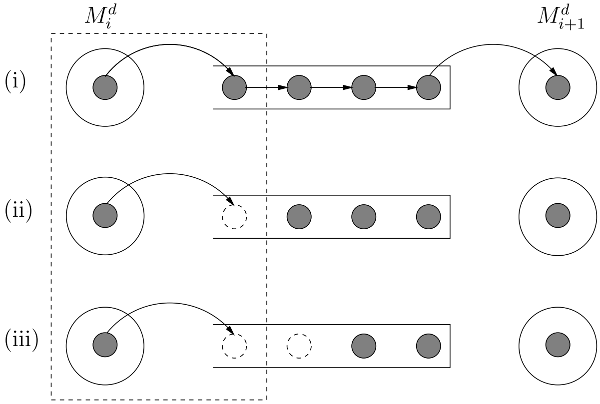

Fig. 3 Possible situations in downstream subsystem L_{i+1} encountered on departure from L_i

situations? Now we are not going to act as if the probability that the next departure from L_i encounters either of the three situations is independent of the past. This would imply that successive service times D_i are independent (and they are not). Instead, we are going to introduce transition probabilities between the above three situations, i.e., the probability that a departure encounters situation (i), (ii) or (iii) depends on the situation encountered by the previous one. Hence, the service process of M_i^d will be described by a Markov chain.

If a departure from L_i sees situation (i), and M_i^d can immediately start with the next S_i and finishes before M_{i+1}^d finishes D_{i+1}^b, then the next departure from L_i sees again (i). However, if M_{i+1}^d finishes first, then on completion of S_i by M_i^d, both (ii) or (iii) may be seen. We denote by $p_{i+1}^{b,nf}$ the probability that M_{i+1}^d completes at least two services before the next arrival at L_{i+1}, given that M_{i+1}^d completes at least one service before the next arrival. So, if M_{i+1}^d finishes first, then the next departure from L_i sees (iii) with probability $p_{i+1}^{b,nf}$, and (ii) otherwise. We assumed that M_i^d can immediately start service after a departure. If, on the other hand, M_i^d is starved and has to wait for the next job to arrive, then D_{i+1}^b should be replaced by the residual time of D_{i+1}^b at the service start of M_i^d.

The transitions are the same from situation (ii), except that D_{i+1}^b should be replaced by D_{i+1}^f. So, if a departure from L_i sees situation (ii), and M_i^d finishes before M_{i+1} (i.e., $S_i < D_{i+1}^f$), then the next departure from L_i certainly sees (i). If $S_i > RD_{i+1}^f$, then the next departure from L_i sees (iii) with probability $p_{i+1}^{f,nf}$ and (ii) otherwise.

Finally, in situation (iii), the next departure from L_i will never see (i). It will see (ii) with probability $p_{i+1}^{nf,f}$ and (iii) otherwise, where $p_{i+1}^{nf,f}$ is defined as the probability that, on an arrival at L_{i+1}, there is exactly one position left in the buffer of L_{i+1}. The different situations and possible transitions are summarized in Table 1, where we assume that M_i^d can immediately start with the next service after a departure. If this is not the case, then D_{i+1}^b and D_{i+1}^f should be replaced by their residual times at the start of the next service of M_i^d (since D_{i+1}^b and D_{i+1}^f will always start at the moment of a departure).

This completes the description of the service processes of the arrival and departure servers of L_i. In the next section, we translate subsystem L_i to a Quasi-Birth-Death (QBD) process; see Latouche and Ramaswami (1999).

Table 1 Different situations and possible transitions of the service process of departure server M_i^d

Service starts in	Aggregate service time	Next service starts in		
(i)	$\max(S_i, D_{i+1}^b)$	if $S_i < D_{i+1}^b$:	(i)	
		if $S_i > D_{i+1}^b$:	(ii)	w.p. $1 - p_{i+1}^{b,nf}$
			(iii)	w.p. $p_{i+1}^{b,nf}$
(ii)	$\max(S_i, D_{i+1}^f)$	if $S_i < D_{i+1}^f$:	(i)	
		if $S_i > D_{i+1}^f$:	(ii)	w.p. $1 - p_{i+1}^{f,nf}$
			(iii)	w.p. $p_{i+1}^{f,nf}$
(iii)	S_i		(ii)	w.p. $p_{i+1}^{nf,f}$
			(iii)	w.p. $1 - p_{i+1}^{nf,f}$

6 Subsystem

In this section, we describe the analysis of a subsystem L_i (cf. Steps 1(b) and 1(c) in Sect. 3). For ease of notation, we drop the subscript i in the sequel of this section. In order to translate L to a Markov process, we will describe the random variables introduced in the foregoing sections in terms of exponential phases, commonly referred to as phase-type distributed random variables (see e.g. Tijms 1994). In Sect. 6.1, we first explain how to fit phase-type distributions on the first and second moment. By employing this concept, we translate subsystem L to a Quasi-Birth-and-Death process (QBD) in Sect. 6.2. Based on the steady-state distribution of this QBD, we derive performance measures, which can be used to model the service process of the arrival server succeeding the subsystem and the service process of the departure server preceding the subsystem.

6.1 Fitting phase-type distributions on the first two moments

Consider a random variable X with mean $\mathbb{E}[X]$ and second moment $\mathbb{E}[X^2]$. The squared coefficient of variation c_X^2 is defined as

$$c_X^2 = \frac{\mathrm{var}(X)}{\mathbb{E}^2[X]} = \frac{\mathbb{E}[X^2]}{\mathbb{E}^2[X]} - 1.$$

We adopt the following recipe to fit a phase-type distribution on $\mathbb{E}[X]$ and c_X^2, see Tijms (1994). If $1/k \leq c_X^2 \leq 1/(k-1)$ for some $k = 2, 3, \dots$, then the mean and squared coefficient of variation of the Erlang$_{k-1,k}$ distribution with density

$$f(x) = p\mu^{k-1}\frac{x^{k-2}}{(k-2)!}e^{-\mu t} + (1-p)\mu^k\frac{x^{k-1}}{(k-1)!}e^{-\mu x}, \quad x \geq 0, \tag{4}$$

matches $\mathbb{E}[X]$ and c_X^2, provided the parameters p and μ are chosen as

$$p = \frac{1}{1+c_X^2}\left(kc_X^2 - \left(k\left(1+c_X^2\right) - k^2 c_X^2\right)^{1/2}\right), \qquad \mu = \frac{k-p}{\mathbb{E}(X)}.$$

Hence, in this case we may describe X in terms of a random sum of $k-1$ or k independent exponential phases, each with rate μ. The phase diagram of the Erlang$_{k-1,k}$ distribution is

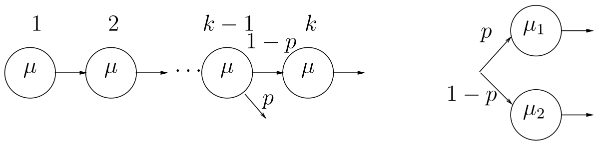

Fig. 4 Phase diagram of $\text{Erlang}_{k-1,k}$ distribution (*left*) and $\text{Hyper-exponential}_2$ distribution (*right*)

illustrated in the left part of Fig. 4. Alternatively, if $c_X^2 > 1$, then the $\text{Hyper-Exponential}_2$ distribution with density

$$f(t) = p\mu_1 e^{-\mu_1 x} + (1-p)\mu_2 e^{-\mu_2 x}, \quad x \geq 0, \tag{5}$$

matches $\mathbb{E}[X]$ and c_X^2, provided the parameters p, μ_1 and μ_2 are chosen as

$$p = \frac{1}{2}\left(1 + \sqrt{\frac{c_X^2 - 1}{c_X^2 + 1}}\right), \qquad \mu_1 = \frac{2p}{\mathbb{E}[X]}, \qquad \mu_2 = \frac{2(1-p)}{\mathbb{E}[X]}.$$

This means that X can be represented in terms of a probabilistic mixture of two exponential phases with rates μ_1 and μ_2, respectively. The phase diagram of the $\text{Hyper-exponential}_2$ distribution is illustrated in the right part of Fig. 4.

A unified representation of Erlang and Hyper-exponential distributions is provided by the family of Coxian distributions (cf. Cumani 1982). A random variable X is said to have a Coxian_k distribution if it has to go through at most k exponential phases, where phase i has rate ν_i, $i = 1, \ldots, k$. It starts in phase 1 and after phase i, $i = 1, \ldots, k-1$, it enters phase $i+1$ with probability p_i, and otherwise, it exits with probability $1 - p_i$. Phase k is the last phase, so $p_k = 0$. Clearly, the $\text{Erlang}_{k-1,k}$ distribution is a Coxian_k distribution with $\nu_i = \mu$ for all i and $p_i = 1$ for $i = 1, \ldots, k-2$ and $p_{k-1} = 1 - p$. The $\text{Hyper-exponential}_2$ distribution is a Coxian_2 distribution with

$$\nu_1 = \mu_1, \qquad \nu_2 = \mu_2, \qquad p_1 = (1-p)\frac{\mu_1 - \mu_2}{\mu_1},$$

where, without loss of generality, $\mu_1 \geq \mu_2$. This representation of Erlang and Hyper-exponential distributions in terms of Coxians will be convenient for the description of the service processes of the arrival and departure server in Appendices A and B.

It is also possible to use phase-type distributions matching the first three (or even higher) moments; see e.g. van der Heijden (1993), Osogami and Harchol-Balter (2003). Obviously, there exist many phase-type distributions matching the first two moments. However, numerical experiments suggest that the use of other distributions does not essentially affect the results, cf. Johnson (1993).

6.2 Subsystem analysis

We apply the recipe of Sect. 6.1 to represent each of the random variables A, S, D^b and D^f in terms of exponential phases. The status of the service process of the arrival server M^a can be easily described by the service phase of A. The description of the service process of the departure server M^d is more complicated. Here we need to keep track of the phase of S and the phase of D^b or D^f, depending on situation (i), (ii) or (iii). The description of this service process is illustrated in the following example.

Example Suppose that S can be represented by two successive exponential phases, D^b by three phases and D^f by a single phase, where each phase possibly has a different rate. Then

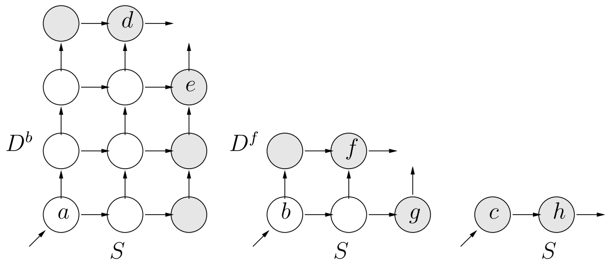

Fig. 5 Phase diagram for the service process of the departure server

the phase-diagram for each situation (i), (ii), and (iii) is sketched in Fig. 5. States a, b and c are the initial states for each situation. The gray states indicate that either S, D^b or D^f has completed all phases. A transition from one of the states d, e, f, g and h corresponds to a service completion of departure server M^d (i.e., a departure from subsystem L); the other transitions correspond to a phase completion, and do not trigger a departure. The probability that a transition from state e is directed to initial state a is equal to 1; the probability that a transition from state d is directed to initial state a, b and c is equal to 0, $1 - p^{b,nf}$ and $p^{b,nf}$, respectively. The transition probabilities from the other states f, g and h can be found similarly.

In Fig. 5 it is assumed that M^d can immediately start with the next service S after a departure. However, if M^d is starved, then S will not immediately start but has to wait for the next arrival at L (i.e., service completion of the arrival server M^a). However, D^b or D^f will immediately start completing their phases, and may even have completed all their phases at the start of S.

From the example above, it will be clear that the service process of M^d can be described by a Markovian Arrival Process (MAP): a finite-state Markov process with generator Q_d. This generator can be decomposed as $Q_d = Q_{d0} + Q_{d1}$, where the transitions of Q_{d1} correspond to service completions (i.e., departures from L) and the ones of Q_{d0} correspond to transitions not leading to departures. The dimension n_d of Q_d can be large, depending on the number of phases required for S, D^b and D^f. Similarly, the service process of M^a can be described by a Markovian Arrival Process with generator $Q_a = Q_{a0} + Q_{a1}$ of dimension n_a. For an extensive treatment of MAPs, we refer the reader to Neuts (1989). The specification of the generators Q_a and Q_d is deferred to Appendices A and B, respectively.

Subsystem L can be described by a QBD with states (i, j, l), where i denotes the number of jobs in subsystem L, excluding the one at the arrival server M^a. Clearly, $i = 0, \dots, b+2$, where $i = b + 2$ indicates that the arrival server is blocked because buffer B is full. The state variables j and l denote the state of the arrival and departure process, respectively. To specify the generator $\mathbf{Q}$ of the QBD we use the Kronecker product: If A is an $n_1 \times n_2$ matrix and B is an $n_3 \times n_4$ matrix, the Kronecker product $A \otimes B$ is defined as

$$A \otimes B = \begin{pmatrix} A(1,1)B & \cdots & A(1,n_2)B \\ \vdots & & \vdots \\ A(n_1,1)B & \cdots & A(n_1,n_2)B \end{pmatrix}.$$

We order the states lexicographically and partition the state space into *levels*, where level $i = 0, 1, \ldots, b+2$ is the set of all states with i jobs in the system. Then $\mathbf{Q}$ takes the form:

$$\mathbf{Q} = \begin{pmatrix} B_{00} & B_{01} & & & & \\ B_{10} & A_1 & A_0 & & & \\ & A_2 & \ddots & \ddots & & \\ & & \ddots & \ddots & A_0 & \\ & & & A_2 & A_1 & C_{10} \\ & & & & C_{01} & C_{00} \end{pmatrix}.$$

Below we specify the submatrices in $\mathbf{Q}$. The transition rates from levels $1 \le i \le b$ are given by

$$A_0 = Q_{a1} \otimes I_{n_d},$$
$$A_1 = Q_{a0} \otimes I_{n_d} + I_{n_a} \otimes Q_{d0},$$
$$A_2 = I_{n_a} \otimes Q_{d1},$$

where I_n is the identity matrix of size n. The transition rates are different for the levels $i = 0$ and $i = b+2$. At level $b+2$ the arrival server M^a is blocked, so

$$C_{10} = Q_{a1}(:, 1) \otimes I_d,$$
$$C_{00} = Q_{d0},$$
$$C_{01} = I_{n_a}(1, :) \otimes Q_{d1},$$

where $P(x, :)$ is the x-th row of matrix P and $P(:, y)$ is the y-th column of P. To specify the transition rates to level 0, we introduce the transition rate matrix Q_s of dimension n_s, describing the progress of the phases of D^b or D^f while the departure server M^d is starved. Further, the $n_d \times n_s$ matrix $\bar{Q}_{d1}$ contains the transition rates from states in Q_d, that correspond to a departure, to the initial states in Q_s. Finally, $\bar{I}_{n_s,n_d}$ is the 0-1 matrix of size $n_s \times n_d$ that preserves the phase of Q_s (i.e., the phase of D^b or D^f) when the departure server M^d starts serving the next job after having been starved. Then we obtain

$$B_{10} = I_{n_a} \otimes \bar{Q}_{d1},$$
$$B_{00} = Q_{a0} \otimes I_{n_s} + I_{n_a} \otimes Q_s,$$
$$B_{01} = Q_{a1} \otimes \bar{I}_{n_s,n_d}.$$

This concludes the specification of $\mathbf{Q}$.

The steady-state distribution of the QBD can be determined by the matrix-geometric method; see e.g Latouche and Ramaswami (1999), Naoumov et al. (1997), van Vuuren and Adan (2009). We denote the equilibrium probability vector of level i by π_i. Then π_i has the matrix-geometric form

$$\pi_i = x_1 R^{i-1} + x_{b+1} \hat{R}^{b+1-i}, \quad i = 1, \ldots, b+1, \tag{6}$$

where R is the minimal nonnegative solution of the matrix-quadratic equation

$$A_0 + RA_1 + R^2 A_2 = 0,$$

and $\hat{R}$ is the minimal nonnegative solution of

$$A_2 + \hat{R} A_1 + \hat{R}^2 A_0 = 0.$$

The matrices R and $\hat{R}$ can be efficiently determined by using an iterative algorithm developed in Naoumov et al. (1997). The vectors π_0, x_1, x_{b+1} and π_{b+2} follow from the balance equations at the boundary levels 0, 1, $b+1$ and $b+2$,

$$
\begin{aligned}
0 &= \pi_0 B_{00} + \pi_1 B_{10}, \\
0 &= \pi_0 B_{01} + \pi_1 A_1 + \pi_2 A_2, \\
0 &= \pi_b A_0 + \pi_{b+1} A_1 + \pi_{b+2} C_{01}, \\
0 &= \pi_{b+1} C_{10} + \pi_{b+2} C_{00}.
\end{aligned}
$$

Substitution of (6) for π_1 and π_{b+1} in the above equations yields a set of linear equations for π_0, x_1, x_{b+1} and π_{b+2}, which together with the normalization equation, has a unique solution. This completes the determination of the equilibrium probabilities vectors π_i. Once these probability vectors are known, we can easily derive performance measures and quantities required to describe the service times of the arrival and departure server.

Throughput:
The throughput T satisfies

$$
\begin{aligned}
T &= \pi_1 B_{10} e + \sum_{i=2}^{b+1} \pi_i A_2 e + \pi_{b+2} C_{01} e \\
&= \pi_0 B_{01} e + \sum_{i=1}^{b} \pi_i A_0 e + \pi_{b+1} C_{10} e,
\end{aligned} \tag{7}
$$

where e is the all-one vector.

Service process of the arrival server:
To specify the service time of the arrival server we need the probability q^e that the system is empty just after a departure and the first two moments of the residual service time RA of the arrival server at the time of such an event. The probability q^e is equal to the mean number of departures per time unit leaving behind an empty system divided by the mean total number of departures per time unit. So

$$
q^e = \pi_1 B_{10} e / T. \tag{8}
$$

The moments of RA can be easily obtained, once the distribution of the phase of the service time of the arrival server, just after a departure leaving behind an empty system, is known. Note that component (j,k) of the vector $\pi_1 B_{10}$ is the mean number of transitions per time unit from level 1 entering state (j,k) at level 0. By adding all components with $j = l$ and dividing by $\pi_1 B_{10} e$, i.e., the mean total number of transitions per time unit from level 1 to 0, we obtain the probability that the arrival server is in phase l just after a departure leaving behind an empty system. Further, if the service time A of the arrival server is represented by a Coxian distribution with n_a phases, where phase j has rate ω_j and exit probability $1 - p_j$, $j = 1, \ldots, n_a$, then the first two moments of the residual service time RA given that the service time A is in phase l are given by

$$
\begin{aligned}
\mathbb{E}[RA | A \text{ in } l] &= \sum_{j=l}^{n_a} \prod_{k=l}^{j-1} p_k (1 - p_j) \sum_{k=l}^{j} \frac{1}{\omega_k}, \\
\mathbb{E}\big[RA^2 | A \text{ in } l\big] &= \sum_{j=l}^{n_a} \prod_{k=l}^{j-1} p_k (1 - p_j) \sum_{k=l}^{j} \sum_{m=l}^{j} \frac{2}{\omega_k \omega_m}.
\end{aligned}
$$

Summation of the conditional moments multiplied by the probability of being in phase l yields the moments of RA.

Service process of the departure server:
We need to calculate the first two moments of D^b and D^f and the transition probabilities $p^{b,nf}$, $p^{f,nf}$ and $p^{nf,f}$. This requires the distribution of the initial phase upon entering level $b+1$ due to a departure (or arrival). Clearly, component (j,k) of $\pi_{b+2}C_{01}$ is equal to the number of transitions per time unit from level $b+2$ entering state (j,k) at level $b+1$. Hence, $\pi_{b+2}C_{01}/\pi_{b+2}C_{01}e$ yields the distribution of the initial phase upon entering level $b+1$ due to a departure. Defining $D^b(1)$ and $D^b(2)$ as the time till the first, respectively second, departure and $A^b(1)$ as the time till the first arrival, from the moment of entering level $b+1$, it is straightforward to calculate the moments of $D^b(1) \equiv D^b$ and the probabilities $\Pr[D^b(1) < A^b(1)]$ and $\Pr[D^b(2) < A^b(1)]$. Transition probability $p^{b,nf}$ now follows from

$$p^{b,nf} = \Pr\big[D^b(2) < A^b(1) | D^b(1) < A^b(1)\big] = \frac{\Pr[D^b(2) < A^b(1)]}{\Pr[D^b(1) < A^b(1)]}.$$

Calculation of the moments of D^f and transition probability $p^{f,nf}$ proceeds along the same lines, where the distribution of the initial phase upon entering level $b+1$ due to an arrival is given by $\pi_b A_0/\pi_b A_0 e$. Finally, $p^{nf,f}$ satisfies

$$p^{nf,f} = \frac{\pi_b A_0 e}{\pi_0 B_{01} e + \sum_{i=1}^{b} \pi_i A_0 e}.$$

7 Numerical results

In order to investigate the quality of the current method we evaluate a large set of examples and compare the results with discrete-event simulation. We also compare the results with the approximation of van Vuuren and Adan (2009), a recent and accurate approximation. The crucial difference between the two methods lies in the modeling of the departure service process: the current method attempts to take into account dependencies between successive service times. In each example we assume that only mean and squared coefficient of variation of the service times at each server are known, and we match, both in the approximation and discrete-event simulation, mixed Erlang or Hyper-exponential distributions to the first two moments of the service times, depending on whether the coefficient of variation is less or greater than 1; see (4) and (5) in Sect. 6. Then we compare the throughput and the mean sojourn time (i.e., the mean time that elapses from the service start at server M_0 until service completion at server M_{N-1}) produced by the current approximation and the ones in van Vuuren and Adan (2009) with the ones produced by discrete-event simulation. Each simulation run is sufficiently long such that the widths of the 95% confidence intervals of the throughput and mean sojourn time are smaller than 1%.

We use the following set of parameters for the tests. The mean service times of the servers are all set to 1. We vary the number of servers in the tandem queue between 4, 8, 16, 24 and 32. The squared coefficient of variation (SCV) of the service times of each server is the same and is varied between 0.5, 1, 2, 3 and 5. The buffer sizes between the servers are the same and varied between 0, 1, 3 and 5. We will also test three kinds of *imbalance* in the tandem queue. We test imbalance in the mean service times by increasing the average service time of the 'even' servers from 1 to 1.2. The effect of imbalance in the SCV is tested by increasing the SCV of the service times of the 'even' servers by 0.5. Finally, imbalance in the buffer

Table 2 Overall results for tandem queues with different buffer sizes

Buffer sizes	Error (%) in the throughput					Error (%) in mean sojourn time				
	Avg.	0–2	2–4	>4	VA	Avg.	0–2	2–4	>4	VA
0, 0, ...	1.41	88	10	2	7.22	3.94	53	19	28	11.66
1, 1, ...	3.99	46	36	18	4.60	2.89	59	30	11	7.14
3, 3, ...	3.32	56	28	16	3.85	2.03	75	25	0	4.61
5, 5, ...	2.23	75	20	5	3.69	2.25	66	32	2	3.89
0, 2, ...	1.56	89	11	0	4.70	2.38	75	23	2	6.76
1, 3, ...	3.36	58	27	15	3.95	2.41	69	31	0	4.94
3, 5, ...	2.71	66	21	13	3.63	1.88	77	22	1	3.88
5, 7, ...	1.88	79	21	0	3.53	2.50	66	30	4	3.70

Table 3 Overall results for tandem queues with different SCVs of the service times

SCVs	Error (%) in the throughput					Error (%) in mean sojourn time				
	Avg.	0–2	2–4	>4	VA	Avg.	0–2	2–4	>4	VA
0.5, 0.5, ...	0.77	100	0	0	1.03	2.37	70	21	9	2.67
1, 1, ...	1.22	100	0	0	1.27	2.18	75	19	6	2.83
2, 2, ...	1.85	90	10	0	3.09	2.24	76	20	4	4.95
3, 3, ...	2.90	51	49	0	5.53	2.40	75	23	2	7.20
5, 5, ...	5.58	15	45	40	9.64	3.29	48	45	7	10.31
0.5, 1, ...	0.88	100	0	0	1.60	2.53	70	23	7	3.08
1, 1.5, ...	1.22	100	0	0	1.85	2.26	73	21	6	3.20
2, 2.5, ...	2.00	85	15	0	3.74	2.23	76	20	4	5.60
3, 3.5, ...	3.19	41	59	0	6.18	2.40	75	23	2	7.75
5, 5.5, ...	5.96	14	40	46	10.06	3.43	38	51	11	10.63

Table 4 Overall results for tandem queues with different mean service times

Mean service times	Error (%) in the throughput					Error (%) in mean sojourn time				
	Avg.	0–2	2–4	>4	VA	Avg.	0–2	2–4	>4	VA
1, 1, ...	2.65	68	23	9	4.23	2.50	69	26	5	5.71
1, 1.2, ...	2.46	71	21	8	4.57	2.57	67	27	6	5.93

sizes is tested by increasing the buffers size of the ‘even’ buffers by 2. This leads to a total of 800 test cases.

The results for each category are summarized in Tables 2, 3, 4 and 5. Each table lists the average error in the throughput and the mean sojourn time compared with simulation results. Each table also gives for three error-ranges the percentage of the cases that fall in that range, and the average error of the approximation of van Vuuren and Adan (2009), denoted by VA.

From the tables we can conclude that the current method performs well and better than van Vuuren and Adan (2009). The overall average error in the throughput is 2.56% and the overall average error in the mean sojourn time is 2.54%, while the corresponding percentages for van Vuuren and Adan (2009) are 4.40% and 5.82%, respectively.

Table 5 Overall results for tandem queues of different length

Servers in line	Error (%) in the throughput					Error (%) in mean sojourn time				
	Avg.	0–2	2–4	>4	VA	Avg.	0–2	2–4	>4	VA
4	2.26	69	29	2	0.57	1.77	83	17	0	0.95
8	2.68	66	27	7	2.87	1.82	81	18	1	2.80
16	2.68	68	21	11	5.30	1.63	88	9	3	5.39
24	2.55	72	17	11	6.41	2.65	66	26	8	8.38
32	2.61	73	16	11	6.84	4.80	19	62	19	11.59

Table 6 Throughput for four-server lines with exponential service times

$\tau_i = 1/\mu_i$				b_i			Throughput			
0	1	2	3	1	2	3	Exact	App	Buz	Per
1	1.1	1.2	1.3	1	1	1	0.710	0.702	0.700	0.694
1	1.2	1.4	1.6	1	1	1	0.765	0.759	0.756	0.751
1	1.5	2	2.5	1	1	1	0.861	0.858	0.855	0.853
1	2	3	4	1	1	1	0.929	0.929	0.927	0.927

Table 7 Throughput for three-server lines with general service times

$\tau_i = 1/\mu_i$			$c^2_{S_i}$			b_i		Throughput			
0	1	2	0	1	2	1	2	Sim	App	Buz	Alt
0.5	0.5	0.5	0.75	0.75	0.75	2	2	0.385	0.383	0.385	0.368
0.5	0.5	0.5	2	2	2	2	2	0.322	0.317	0.312	0.338
0.5	0.5	0.5	2	2	2	2	9	0.360	0.353	0.349	0.368

In Table 2 it is striking that in case of zero buffers the current method produces the most accurate estimates, while the method of van Vuuren and Adan (2009) produces the least accurate results. A possible explanation is that for each subsystem the current method keeps track of the status of the downstream server while its departure server is starved; this is not done in van Vuuren and Adan (2009). Both methods seem to be robust to variations in buffer sizes along the line. Table 3 convincingly demonstrates that especially in case of service times with high variability the current approximation performs much better that van Vuuren and Adan (2009). Remarkably, Table 5 shows that, while van Vuuren and Adan (2009) performs better for short lines, the average error in the throughput of the current method does not seem to increase for longer lines, a feature not shared by the approximation of van Vuuren and Adan (2009).

Lastly, we compare the current method to other approaches reported in the literature. In Table 6, results are listed for tandem lines with four servers and exponential service times (used in Buzacott et al. 1995). The columns Exact, App, Buz, and Per list the exact results, results of the current approximation, the approximation of Buzacott et al. (1995), and the one of Perros and Altiok (1986). Table 7 lists results for tandem lines with three servers and non-exponential service times. In this table, the columns Sim, App, Buz, and Alt show results of simulation, the current approximation, the approximation of Buzacott et al. (1995), and the one of Altiok (1989). Both tables show that the methods perform well on these cases.

8 Conclusions

In this paper, we developed an approximate analysis of single-server tandem queues with finite buffers, based on decomposition into single-buffer subsystems. The distinguishing feature of the analysis is that dependencies between successive aggregate service times (including starvation and blocking) are taken into account. Numerical results convincingly demonstrated that it pays to include such dependencies, especially in case of longer tandem queues and service times with a high variability. The price to be paid, of course, is that the resulting subsystems are more complex and computationally more demanding.

We conclude with a remark on the subsystems. There seems to be an asymmetry in the modeling of the service processes of the arrival and departure server. The service times of the arrival server are assumed to be independent and identically distributed, whereas the service times of the departure server are modeled by a Markovian arrival process, carefully taking into account dependencies between successive service times. Investigating whether a similar Makovian description of the service process of the arrival server is also feasible (and rewarding) seems to be an interesting direction for future research.

Appendix A: Arrival service generator $Q_{a0} + Q_{a1}$

This appendix is devoted to the specification of generator $Q_a = Q_{a0} + Q_{a1}$ of the MAP describing the service process of the arrival server M^a. Using the first two moments of A from (2) and (3) we can fit a phase-type distribution as described in Sect. 6.1. We can equivalently represent this distribution as a Coxian distribution with n_a phases, numbered $1, \dots, n_a$; the starting phase is 1, the rate of phase j is ω_j, p_j is the probability to proceed to the next phase $j+1$, and $1-p_j$ is the probability that A is completed. Since n_a is the last phase, we have $p_{n_a} = 0$. Then the states of the MAP are numbered $1, \dots, n_a$. Its generator can be expressed as $Q_{a0} + Q_{a1}$, where the transition rates in Q_{a1} are the ones corresponding to a service completion, i.e., an arrival in the buffer. The non-zero elements of Q_{a0} and Q_{a1} are presented below:

$$
\begin{aligned}
Q_{a0}(j,j) &= -\omega_j, && j = 1, \dots, n_a, \\
Q_{a0}(j,j+1) &= p_j\omega_j, && j = 1, \dots, n_a - 1, \\
Q_{a1}(j,1) &= (1-p_j)\omega_j, && j = 1, \dots, n_a.
\end{aligned}
$$

Appendix B: Departure service generator $Q_{d0} + Q_{d1}$

In this appendix we specify the generator $Q_d = Q_{d0} + Q_{d1}$ of the service process of the departure server M^d. The generator Q_d applies to the situation that M^d can immediately start with the next service after a departure (see Table 1). In case M^d is starved, the arrival and departure processes are specified by B_{10}, B_{00} and B_{01} in Sect. 6.

First, we fit a phase-type distribution on the random variables S, D^b, and D^f as described in Sect. 6.1. In the same way as in Appendix A, we construct a MAP for each of the three random variables with the following generators:

- $Q_{db} = Q_{db0} + Q_{db1}$ of size $n_{db} \times n_{db}$ for D^b,
- $Q_{df} = Q_{df0} + Q_{df1}$ of size $n_{df} \times n_{df}$ for D^f,
- $Q_s = Q_{s0} + Q_{s1}$ of size $n_s \times n_s$ for S.

These generators can be used to construct Q_{d0} and Q_{d1}. We start with specifying the transition rates in Q_{d0}, which are the phase transitions not corresponding to a service completion (i.e. not corresponding to a departure from the subsystem). We divide the states of D in three groups, each group corresponding to one of the situations in Sect. 5. The matrix Q_{d0} can be divided accordingly into blocks:

$$Q_{d0} = \begin{pmatrix} Q_{d0}^{(i)} & 0 & 0 \\ 0 & Q_{d0}^{(ii)} & 0 \\ 0 & 0 & Q_{d0}^{(iii)} \end{pmatrix}.$$

Note that once we start service in either of the three situations, we cannot move to another situation without having a departure first. This explains why the non-diagonal blocks in Q_{d0} consist of zeros only. The matrix $Q_{d0}^{(i)}$ corresponds to situation (i) in Sect. 5 and is again divided in three parts. In the first part of the process, the phases of both S and D^b are uncompleted, in the second part the phases of S are completed and the phases of D^b are uncompleted, and in the third part the phases of S are uncompleted and the phases of D^b are completed. Using the Kronecker product as defined in Sect. 6.2 we get:

$$Q_{d0}^{(i)} = \begin{pmatrix} Q_{s0} \otimes I_{n_{db}} + I_{n_s} \otimes Q_{db0} & Q_{s1}(:,1) \otimes I_{n_{db}} & I_{n_s} \otimes Q_{db1}(:,1) \\ 0 & Q_{db0} & 0 \\ 0 & 0 & Q_{s0} \end{pmatrix},$$

where I_n is the identity matrix of size n and $P(:,1)$ is the first column of the matrix P. Transition matrix $Q_{d0}^{(ii)}$ corresponds to situation (ii) in Sect. 5 and can be obtained in a similar way as $Q_{d0}^{(i)}$:

$$Q_{d0}^{(ii)} = \begin{pmatrix} Q_{s0} \otimes I_{n_{df}} + I_{n_s} \otimes Q_{df0} & Q_{s1}(:,1) \otimes I_{n_{df}} & I_{n_s} \otimes Q_{df1}(:,1) \\ 0 & Q_{df0} & 0 \\ 0 & 0 & Q_{s0} \end{pmatrix}.$$

The matrix $Q_{d0}^{(iii)}$ corresponds to situation (iii) in Sect. 5 where D is equal to S, so

$$Q_{d0}^{(iii)} = Q_{s0}.$$

Next, we obtain the transition rates in Q_{d1} corresponding to departures from the subsystem. As for Q_{d0}, we divide the states in three groups corresponding to the three situations in Sect. 5, and we adjust Q_{d1} accordingly:

$$Q_{d1} = \begin{pmatrix} Q_{d1}^{(i)\to(i)} & Q_{d1}^{(i)\to(ii)} & Q_{d1}^{(i)\to(iii)} \\ Q_{d1}^{(ii)\to(i)} & Q_{d1}^{(ii)\to(ii)} & Q_{d1}^{(ii)\to(iii)} \\ Q_{d1}^{(iii)\to(i)} & Q_{d1}^{(iii)\to(ii)} & Q_{d1}^{(iii)\to(iii)} \end{pmatrix},$$

where, for instance, the rates in $Q^{(iii)\to(i)}$ correspond to completions of services which started in situation (iii), and which encounter situation (i) on completion. Recall that we divided the states corresponding to situation (i) in three parts: phases of both S and D^b uncompleted, only phases of S completed, and only phases of D^b completed. In Sect. 5, we argued that if $S < D^b$, the next service starts in situation (i). However, if $S > D^b$, the next service starts in situation (ii) with probability $1 - p^{b,nf}$ and in situation (iii) with probability $p^{b,nf}$. Based on this information, we can construct the first rows of Q_{d1}:

$$Q_{d0}^{(i)\to(i)} = \begin{pmatrix} 0 & 0 & 0 \\ e_{n_s} \otimes Q_{db1} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad Q_{d0}^{(i)\to(ii)} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ (1 - p^{b,nf}) Q_{s1} \otimes e_{n_{db}} & 0 & 0 \end{pmatrix},$$

$$Q_{d0}^{(i)\to(iii)} = \begin{pmatrix} 0 \\ 0 \\ p^{b,nf} Q_{s1} \end{pmatrix},$$

where e_n is the first row of the identity matrix of size n. Similarly we get:

$$Q_{d0}^{(ii)\to(i)} = \begin{pmatrix} 0 & 0 & 0 \\ e_{n_s} \otimes Q_{db1} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad Q_{d0}^{(ii)\to(ii)} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ (1 - p^{f,nf}) Q_{s1} \otimes e_{n_{df}} & 0 & 0 \end{pmatrix},$$

$$Q_{d0}^{(ii)\to(iii)} = \begin{pmatrix} 0 \\ 0 \\ p^{f,nf} Q_{s1} \end{pmatrix},$$

and finally,

$$Q_{d0}^{(iii)\to(i)} = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}, \qquad Q_{d0}^{(iii)\to(ii)} = \begin{pmatrix} p^{nf,f} Q_{s1} \otimes e_{n_{df}} & 0 & 0 \end{pmatrix},$$
$$Q_{d0}^{(iii)\to(iii)} = \left(1 - p^{nf,f}\right) Q_{s1}.$$

References

Altiok, T. (1989). Approximative analysis of queues in series with phase-type service times and blocking. *Operations Research*, *37*, 601–610.

Buzacott, J. A., Liu, X. G., & Shanthikumar, J. G. (1995). Multistage flow line analysis with the stopped arrival queue mode. *IIE Transactions*, *27*, 444–455.

Colledani, M., & Tolio, T. (2011). Performance evaluation of transfer lines with general repair times and multiple failure modes. *Annals of Operations Research*, *182*, 31–65.

Cumani, A. (1982). On the canonical representation of Markov processes modeling failure time distributions. *Microelectronics and Reliability*, *22*, 583–602.

Dallery, Y., & Gershwin, B. (1992). Manufacturing flow line systems: a review of models and analytical results. *Queueing Systems*, *12*, 3–94.

Gershwin, S. B. (1987). An efficient decomposition algorithm for the approximate evaluation of tandem queues with finite storage space and blocking. *Operations Research*, *35*, 291–305.

van der Heijden, M. C. (1993). *Performance analysis for reliability and inventory models*. PhD Thesis, Vrije Universiteit, Amsterdam.

Helber, S. (2005). Analysis of flow lines with Cox-2-distributed processing times and limited buffer capacity. *OR-Spektrum*, *27*, 221–242.

Johnson, M. A. (1993). An empirical study of queueing approximations based on phase-type distributions. *Communications in Statistics. Stochastic Models*, *9*, 531–561.

Kerbache, L., & MacGregor Smith, J. (1987). The generalized expansion method for open finite queueing networks. *The European Journal of Operations Research*, *32*, 448–461.

Latouche, G., & Ramaswami, V. (1999). *ASA-SIAM series on statistics and applied probability: Vol. 5. Introduction to matrix analytic methods in stochastic modeling*

Naoumov, V., Krieger, U. R., & Wagner, D. (1997). Analysis of a multiserver delay-loss system with a general Markovian Arrival Process. In A. S. Alfa & S. R. Chakravarthy (Eds.), *Lecture notes in pure and applied mathematics. Matrix-analytic methods in stochastic models* (pp. 43–66). New York: Marcel Dekker.

Neuts, M. F. (1989). *Structured stochastic matrices of M/G/1-type and their applications*. New York: Marcel Dekker.

Osogami, T., & Harchol-Balter, M. (2003). Closed form solutions for mapping general distributions to minimal PH distributions. *Performance Evaluation*, *63*, 524–552.

Perros, H. G., & Altiok, T. (1986). Approximate analysis of open networks of queues with blocking: tandem configurations. *IEEE Transactions on Software Engineering*, 450–461.

Perros, H. G. (1989). A bibliography of papers on queueing networks with finite capacity queues. *Performance Evaluation*, *10*, 255–260.

Perros, H. G. (1994). *Queueing networks with blocking*. Oxford: Oxford University Press.

Tijms, H. C. (1994). *Stochastic models: an algorithmic approach*. Chichester: Wiley.

van Vuuren, M., Adan, I. J. B. F., & Resing-Sassen, S. A. (2005). Performance analysis of multi-server tandem queues with finite buffers and blocking. *OR-Spektrum*, *27*, 315–339.

van Vuuren, M., & Adan, I. J. B. F. (2009). Performance analysis of tandem queues with small buffers. *IIE Transactions*, *41*, 882–892.