Approximate analysis of single-server tandem queues with finite buffers
Bierbooms, J.J.P.H.; Adan, I.J.B.F.; van Vuuren, M.

Published in:
Annals of Operations Research

DOI:
10.1007/s10479-011-1021-1

Published: 01/01/2013

Document Version
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 27. Dec. 2018
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
© The Author(s) 2011. This article is published with open access at Springerlink.com

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).
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 \( \mu_i \) and squared coefficient of variation \( c_{S_i}^2 \).

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.
2 Decomposition

The original tandem queue $L$ is decomposed into $N - 1$ subsystems $L_1, L_2, \ldots, L_{N-1}$. 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 $i = 1, \ldots, N - 1$.

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 $L_1, L_2, \ldots, L_{N-1}$ 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 $L_i$. 

---

![Diagram of tandem queue decomposition](image-url)
(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 $M^a_i$ of subsystem $L_i$ is blocked, i.e., buffer $B_i$ is full, $M^d_i$ is busy, and $M^a_i$ 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 $L_i$,

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

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

$$\mathbb{E}[A^{(k)}_i] = \frac{1 - p^{(k-1)}_{i,b_i+2}}{T^{(k)}_{i-1}},$$  \hfill (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 $T^{(k)}_i$ 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

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

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

\[
A_i = \begin{cases} 
RA_{i-1} + S_{i-1} & \text{with probability } q_{i-1}^e, \\
S_{i-1} & \text{otherwise.}
\end{cases}
\]

This representation is used to determine the second moment of \( A_i \). Based on \( q_{i-1}^e \) 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

\[
\mathbb{E}[A_i^2] = q_{i-1}^e \mathbb{E}[RA_{i-1}^2] + 2q_{i-1}^e \mathbb{E}[RA_{i-1}] \mathbb{E}[S_{i-1}] + \mathbb{E}[S_{i-1}^2].
\] (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 \( M_i^d \) 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 \( M_{i+1}^d \) if \( b_{i+1} = 0 \)). A job served by \( M_i^d \) 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 \( M_i^d \), as will be explained below.

(i) The arrival is triggered by a service completion of departure server \( M_{i+1}^d \) of \( L_{i+1} \), i.e., server \( M_{i+1}^d \) was blocked because the last position in \( B_{i+1} \) was occupied, and waiting for \( M_{i+1}^d \) to complete service. Then the next service of \( M_i^d \) (if there is a job) and \( M_{i+1}^d \) start simultaneously and buffer \( B_{i+1} \) is full. We denote the time elapsing till the next service completion of departure server \( M_{i+1}^d \) by \( D_{i+1}^b \), 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 \( M_i^d \) is equal to the maximum of \( S_i \) and \( D_{i+1}^b \), if \( M_i^d \) can immediately start with the next service. Otherwise, if \( M_i^d \) is starved just after the departure, \( D_i \) is equal to the maximum of \( S_i \) and the residual time of \( D_{i+1}^b \) at the service start of \( M_i^d \).

(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 \( M_{i+1}^d \) by \( D_{i+1}^f \), 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 \( D_{i+1}^f \) at the service start of \( M_i^d \).

(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 in situation (i) and (ii) the next job to be served by \( M_i^d \) 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
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 \( S_i \) 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 \( M^d_i \) will be described by a Markov chain.

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

The transitions are the same from situation (ii), except that \( D^b_{i+1} \) should be replaced by \( D^f_{i+1} \). So, if a departure from \( L_i \) sees situation (ii), and \( M^d_i \) finishes before \( M^d_{i+1} \) (i.e., \( S_i < D^f_{i+1} \)), then the next departure from \( L_i \) certainly sees (i). If \( S_i > R D^f_{i+1} \), then the next departure from \( L_i \) sees (iii) with probability \( p_{i+1}^{n,f} \) and (ii) otherwise.

Finally, in situation (iii), the next departure from \( L_i \) will never see (i). It will see (ii) with probability \( p_{i+1}^{n,f} \) and (iii) otherwise, where \( p_{i+1}^{n,f} \) 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 \( M^d_i \) can immediately start with the next service after a departure. If this is not the case, then \( D^b_{i+1} \) and \( D^f_{i+1} \) should be replaced by their residual times at the start of the next service of \( M^d_i \) (since \( D^b_{i+1} \) and \( D^f_{i+1} \) 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 $M_i^d$

<table>
<thead>
<tr>
<th>Service starts in</th>
<th>Aggregate service time</th>
<th>Next service starts in</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i)</td>
<td>$\max(S_i, D_{i+1}^b)$</td>
<td>(i) $\text{if } S_i &lt; D_{i+1}^b$: $\text{w.p. } 1 - p_{i+1}^{b,nf}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) $\text{if } S_i &gt; D_{i+1}^b$: $\text{w.p. } p_{i+1}^{b,nf}$</td>
</tr>
<tr>
<td>(ii)</td>
<td>$\max(S_i, D_{i+1}^f)$</td>
<td>(i) $\text{if } S_i &lt; D_{i+1}^f$: $\text{w.p. } 1 - p_{i+1}^{f,nf}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ii) $\text{if } S_i &gt; D_{i+1}^f$: $\text{w.p. } p_{i+1}^{f,nf}$</td>
</tr>
<tr>
<td>(iii)</td>
<td>$S_i$</td>
<td>(ii) $\text{w.p. } p_{i+1}^{nf,f}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(iii) $\text{w.p. } 1 - p_{i+1}^{nf,f}$</td>
</tr>
</tbody>
</table>

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

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

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 $k = 2, 3, \ldots$, then the mean and squared coefficient of variation of the Erlang$_{k-1,k}$ distribution with density

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

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

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

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 $\mu$. The phase diagram of the Erlang$_{k-1,k}$ distribution is
illustrated in the left part of Fig. 4. Alternatively, if $c^2 \chi > 1$, then the Hyper-Exponential$_2$ distribution with density

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

matches $\mathbb{E}[X]$ and $c^2 \chi$, provided the parameters $p$, $\mu_1$ and $\mu_2$ are chosen as

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

This means that $X$ can be represented in terms of a probabilistic mixture of two exponential phases with rates $\mu_1$ and $\mu_2$, respectively. The phase diagram of the 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 $\nu_i$, $i = 1, \ldots, k$. 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 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 Hyper-exponential$_2$ distribution is a Coxian$_2$ distribution with

$$v_1 = \mu_1, \quad v_2 = \mu_2, \quad p_1 = (1-p) \frac{\mu_1 - \mu_2}{\mu_1},$$

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
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 $0$, $1 - p^{b, sf}$ and $p^{b, sf}$, 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, $i = 0, \ldots, b + 2$, 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 $Q$ 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 \( Q \) takes the form:

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

Below we specify the submatrices in \( Q \). The transition rates from levels \( 1 \leq i \leq b \) 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 \( Q \).

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 \( \pi_i \). Then \( \pi_i \) has the matrix-geometric form

\[
\pi_i = x_1 R^{i-1} + x_{b+1} \hat{R}^{b+1-i}, \quad i = 1, \ldots, b + 1,
\]

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 \( \pi_0, x_1, x_{b+1} \) and \( \pi_{b+2} \) follow from the balance equations at the boundary levels 0, 1, \( b + 1 \) and \( b + 2 \),

\[
0 = \pi_0 B_{00} + \pi_1 B_{10},
0 = \pi_0 B_{01} + \pi_1 A_1 + \pi_2 A_2,
0 = \pi_b A_0 + \pi_{b+1} A_1 + \pi_{b+2} C_{01},
0 = \pi_{b+1} C_{10} + \pi_{b+2} C_{00}.
\]

Substitution of (6) for \( \pi_1 \) and \( \pi_{b+1} \) in the above equations yields a set of linear equations for \( \pi_0, x_1, x_{b+1} \) and \( \pi_{b+2} \), which together with the normalization equation, has a unique solution. This completes the determination of the equilibrium probabilities vectors \( \pi_i \). 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

\[
T = \pi_1 B_{10} e + \sum_{i=2}^{b+1} \pi_i A_2 e + \pi_{b+2} C_{01} e = \pi_0 B_{01} e + \sum_{i=1}^{b} \pi_i A_0 e + \pi_{b+1} C_{10} e,
\]

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 \( R_A \) 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

\[
q^e = \frac{\pi_1 B_{10} e}{T}.
\]

The moments of \( R_A \) 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 \( \omega_j \) and exit probability \( 1 - p_j \), \( j = 1, \ldots, n_a \), then the first two moments of the residual service time \( R_A \) given that the service time \( A \) is in phase \( l \) are given by

\[
\mathbb{E}[R_A | A \text{ in } l] = \sum_{j=l}^{n_a} \prod_{k=l}^{j-1} p_k (1 - p_j) \sum_{k=l}^{j} \frac{1}{\omega_k},
\]

\[
\mathbb{E}[R_A^2 | A \text{ in } l] = \sum_{j=l}^{n_a} \prod_{k=l}^{j-1} p_k (1 - p_j) \sum_{k=l}^{j} \sum_{m=l}^{j} \frac{2}{\omega_k \omega_m}.
\]
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

\[
p_{b,nf} = \Pr[D_b(2) < A_b(1) | D_b(1) < A_b(1)] = \frac{\Pr[D_b(2) < A_b(1)]}{\Pr[D_b(1) < A_b(1)]}.
\]

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 \( \pi_{b}A_0/\pi_{b}A_0e \). Finally, \( p_{nf,f} \) satisfies

\[
p_{nf,f} = \frac{\pi_{b}A_0e}{\pi_0B_{01}e + \sum_{i=1}^{b} \pi_iA_0e}.
\]

### 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
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

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

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

<table>
<thead>
<tr>
<th>$\tau_i = 1/\mu_i$, $b_i$</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exact</td>
</tr>
<tr>
<td>1, 1, 1, 1</td>
<td>0.710</td>
</tr>
<tr>
<td>1, 1.2, 1.4, 1.6</td>
<td>0.765</td>
</tr>
<tr>
<td>1, 1.5, 2, 2.5</td>
<td>0.861</td>
</tr>
<tr>
<td>1, 2, 3, 4</td>
<td>0.929</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>$\tau_i = 1/\mu_i$, $c_{2i}^2$, $b_i$</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sim</td>
</tr>
<tr>
<td>0.5, 0.5, 0.5, 0.75, 0.75, 2</td>
<td>0.385</td>
</tr>
<tr>
<td>0.5, 0.5, 0.5, 2, 2, 2</td>
<td>0.322</td>
</tr>
<tr>
<td>0.5, 0.5, 0.5, 2, 2, 2</td>
<td>0.360</td>
</tr>
</tbody>
</table>

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 than 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 Altıök (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 Markovian description of the service process of the arrival server is also feasible (and rewarding) seems to be an interesting direction for future research.

Open Access This article is distributed under the terms of the Creative Commons Attribution Noncommercial License which permits any noncommercial use, distribution, and reproduction in any medium, provided the original author(s) and source are credited.

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, $\ldots$, $n_a$; the starting phase is 1, the rate of phase $j$ is $\omega_j$, $p_j$ 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, $\ldots$, $n_a$. 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:

\[
Q_{a0}(j, j) = -\omega_j, \quad j = 1, \ldots, n_a,
\]
\[
Q_{a0}(j, j + 1) = p_j \omega_j, \quad j = 1, \ldots, n_a - 1,
\]
\[
Q_{a1}(j, 1) = (1 - p_j) \omega_j, \quad j = 1, \ldots, n_a.
\]

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:

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

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 $Q^{(i)}_{d0}$ 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:

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

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

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

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

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

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:

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

where, for instance, the rates in $Q^{(iii)}_{d1}$ 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 $Q_{d1}$:

$$Q^{(i) \rightarrow (i)}_{d0} = \begin{pmatrix} 0 & 0 & 0 \\ e_n \otimes Q_{db1} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad Q^{(ii) \rightarrow (i)}_{d0} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ (1 - p^{b,nf})Q_{s1} \otimes e_{ndb} & 0 & 0 \end{pmatrix},$$

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

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

$$Q^{(ii) \rightarrow (i)}_{d0} = \begin{pmatrix} 0 & 0 & 0 \\ e_n \otimes Q_{db1} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad Q^{(ii) \rightarrow (ii)}_{d0} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ (1 - p^{f,nf})Q_{s1} \otimes e_{ndf} & 0 & 0 \end{pmatrix},$$

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

and finally,

$$Q^{(iii) \rightarrow (i)}_{d0} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad Q^{(iii) \rightarrow (ii)}_{d0} = \begin{pmatrix} 0 & 0 & 0 \\ p^{nf,f}Q_{s1} \otimes e_{ndf} & 0 & 0 \end{pmatrix},$$

$$Q^{(iii) \rightarrow (iii)}_{d0} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ (1 - p^{nf,f})Q_{s1} \end{pmatrix}.$$


