Quasi-Birth-and-Death processes with restricted transitions and its applications

Juan F. Pérez, Benny Van Houdt

Performance Analysis of Telecommunication Systems,
Department of Mathematics and Computer Science,
University of Antwerp - IBBT,
Middelheimlaan 1, B-2020 Antwerp, Belgium

Abstract

In this paper we identify a class of Quasi-Birth-and-Death (QBD) processes where the transitions to higher (resp. lower) levels are restricted to occur only from (resp. to) a subset of the phase space. These restrictions induce a specific structure in the R or G matrix of the QBD, which can be exploited to reduce the time required to compute these matrices. We show how this reduction can be achieved by first defining and solving a censored process, and then solving a Sylvester matrix equation. To illustrate the applicability and computational gains obtained with this approach, we consider several examples where the referred structures either arise naturally or can be induced by adequately modeling the system at hand. The examples include the general MAP/PH/1 queue, a priority queue with two customer classes, an overflow queueing system and a wireless relay node.

Keywords: Markov chains, Quasi-Birth-and-Death processes

1. Introduction

Quasi-Birth-and-Death (QBD) processes have played a central role in computational probability for the last thirty years [1, 2]. A QBD Markov chain (MC) is a bi-dimensional process where the first dimension is called the level and the second the phase [2]. The level behaves as in a traditional birth-and-death process, increasing or decreasing its value at most by one at each transition epoch. The phase, on the other hand, allows us to include other information about the system under analysis, opening the way for more general models. In a queuing system the phase

Email addresses: juanfernando.perez@ua.ac.be (Juan F. Pérez), benny.vanhoudt@ua.ac.be (Benny Van Houdt)

typically includes the state of the (Markov) processes underlying the service and inter-arrival time distributions. Apart from its broader applicability, the appeal of QBD processes comes from the matrix-geometric nature of its stationary probability vector, as it can be expressed as a function of a boundary probability vector and a *rate* matrix R. This matrix is found by solving a quadratic matrix equation, a problem that has received much attention and for which many algorithms have been proposed [2, 3, 4].

However, as more information is included in the phase dimension, especially if it comes from multiple variables, the size of the rate matrix increases and even the most efficient algorithms, such as Cyclic Reduction [3] or Logarithmic Reduction [4], require long computation times to find R. One way to deal with this limitation is by exploiting the specific structure in the blocks of the QBD MC. For instance, when these blocks are triangular it is possible to compute R significantly faster than in the general case, as has been shown in [5, 6]. Also, if the blocks are themselves block-circulant, i.e., if each block-row is a right-shifted version of its predecessor [23], then the solution of the matrix equation can also be accelerated [7]. A different block structure arises by restricting the upward or downward transitions, namely the upward (resp. downward) transitions are assumed to only occur in (resp. lead to) a certain subset of the phase space. In this case, Grassmann and Tavakoli [8] have demonstrated how to reduce the time per iteration of the linearly-convergent U-based method [9] by means of an UL decomposition. It is precisely this structure which will be the focus of this paper, although our approach is significantly different from that of [8]. To briefly describe the methodology, consider the case of restricted downward transitions and let $\mathcal{S}^+$ be the subset of the phase space toward which these transitions lead. To determine G, we first define a new process by observing the QBD process when the phase variable is in $\mathcal{S}^+$. The new process is of the M/G/1 type but the size of its blocks is equal to the cardinality of $\mathcal{S}^+$. Therefore, we can use Cyclic Reduction [3] to find its associated matrix G_+, which will be shown to be a sub-matrix of the matrix G of the QBD. After finding G_+ we obtain the remaining entries of G by solving a Sylvester matrix equation. If the QBD has restricted upward transitions, the steps to find R are similar, but in this case the censored process is of the GI/M/1-type. Sections 3 and 4 provide a detailed explanation of our approach for the case of restricted downward and upward transitions, respectively. Section 3 also includes some special cases where additional structure can be exploited to further reduce the computation times.

A QBD MC with restricted transitions is not only computationally appealing, but there are many applications where this property arises naturally or can be induced by adequate modeling. For instance, a preemptive priority queue priority queue [10, 11] with two customer classes can be modeled as a QBD MC with restricted downward transitions. If the level is chosen as the number of low-priority customers, the downward transitions are associated to the service completion of a customer of this class. This only occurs in, and take the chain to, a state where there are no high-priority customers. Another example is an overflow queuing system [12] where the second queue only receives new arrivals when the first queue is full. If the level keeps track of the number of customers in the second queue and the phase holds the number of customers in the first, the upward transitions are restricted to take place in those states with a phase that corresponds to a full first queue. This is similar to the case where the inter-arrival times follow an Erlang distribution, since the arrivals can only happen in the states related to the last phase of this distribution. Another example is the QBD MC used in [13] to compute the waiting time distribution of a type-k customer in an MMAP[K]/PH[K]/c (c = 1, 2) queue, where the downward transitions can only lead to a small subset of the phase space. To illustrate our approach, we consider four different examples in Section 5, including a priority queue with two customer classes and an overflow queueing system. In addition, we show how the restricted-downward-transitions structure can be induced in the QBD MC that describes a general MAP/PH/1 queue. This is achieved by defining a (slightly larger) representation of the service-time distribution, forcing the state of its underlying process to re-start in a specific phase, which occurs whenever there is a service completion (downward transition). Our last example is based on the model introduced in [14] to evaluate the packet delay in a wireless relay node, which actually is a QBD MC with restricted downward transitions. These examples are also used to illustrate the computational gains obtained by using the approach introduced here, compared to the traditional methods and to the methods in [8] and [14]. Our method has been implemented in MATLAB and will be made available online as part of the SMCSolver tool [15]. We now turn to Section 2, where we provide a brief review of QBD processes and show the structures under analysis.

2. QBDs with restricted transitions

A discrete-time QBD MC can be defined as a two-dimensional process $\{(N_t, X_t), t \geq 0\}$, where N_t is called the level variable and takes values on $\mathbb{N}$. The phase variable X_t takes values on the set $\{1, 2, \ldots, m_0\}$ or $\{1, 2, \ldots, m\}$ depending on whether the level is equal to or greater than 0. The level variable can only increase or decrease its value by one at each time epoch, and these transition probabilities are level-independent. Therefore the QBD MC has a transition matrix P of the form

$$P = \begin{bmatrix} B_1 & B_2 & 0 & 0 & \ldots \\ B_0 & A_1 & A_2 & 0 & \ldots \\ 0 & A_0 & A_1 & A_2 & \ldots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where B_1 and A_1 are square matrices of size m_0 and m, respectively. The matrices B_1 and B_2 hold the transition probabilities from level 0 to levels 0 and 1, respectively, and the matrix B_0 contains the transition probabilities from level 1 to level 0. Similarly, the matrices A_0, A_1 and A_2 carry the transition probabilities from level i to levels $i-1$, i and $i+1$, respectively, for $i > 0$. The key when computing the steady state probability vector $\pi = [\pi_0, \pi_1, \pi_2, \ldots]$ of P, if it exists, is to find the minimal nonnegative solution R of the matrix equation

$$R = A_2 + RA_1 + R^2 A_0. \tag{1}$$

The vectors π_i can then be computed as $\pi_i = \pi_1 R^i$, for $i > 1$, where $[\pi_0, \pi_1]$ is the solution of the boundary equation

$$[\pi_0, \pi_1] \begin{bmatrix} B_1 & B_2 \\ B_0 & A_1 + RA_0 \end{bmatrix} = [\pi_0, \pi_1].$$

Another way to find the matrix R is from $R = A_1(I - A_0 - A_1 G)^{-1}$, where G is the minimal nonnegative solution of the matrix equation

$$G = A_0 + A_1 G + A_0 G^2. \tag{2}$$

The method introduced in this paper aims at computing either the matrix R or G, from which the stationary probability vector can be obtained. Although in the presentation above we have assumed a specific boundary behavior, our approach can also be applied under more general boundary conditions, as long as the QBD shows a repeating structure (matrices A_0, A_1 and A_2) from a given level onward.

Many iterative algorithms have been developed to solve equations (1) and (2), including Cyclic Reduction (CR) [3] and Logarithmic Reduction [4], which are quadratically convergent. However, a large block size m may turn the solution of these equations into a lengthy task, as each iteration requires $O(m^3)$ time. In this paper we consider two special cases where the structure of the matrices A_0 and A_2 can be exploited to speed up the computation of the matrix G or R. In both cases we consider a partition of the set $\{1, \ldots, m\}$ into two sets: $\mathcal{S}^+$ containing the first r phases, and $\mathcal{S}^-$ containing the remaining $m-r$ phases. Using this partition, the matrices A_i, for $i = \{0, 1, 2\}$, can be written as

$$A_i = \begin{bmatrix} A_i^{++} & A_i^{+-} \\ A_i^{-+} & A_i^{--} \end{bmatrix}, \tag{3}$$

where A_i^{++} and A_i^{--} are square matrices of size r and $m-r$, respectively. In Section 3 we consider the case where downward transitions can only occur to a state with phase in $\mathcal{S}^+$, hence the matrix A_0 has only $r \ll m$ nonzero columns such that it can be written as

$$A_0 = \begin{bmatrix} A_0^{++} & 0 \\ A_0^{-+} & 0 \end{bmatrix}. \tag{4}$$

When the set $\mathcal{S}^+$ contains only one phase the matrix G can be computed explicitly without the need of resorting to iterative algorithms [16]. Furthermore, this particular case has also been exploited to compute performance measures in an efficient manner without computing all the terms of the vector π [17]. In this paper we consider the more general case where the cardinality of $\mathcal{S}^+$ is greater than one, meaning that the matrix G is not known explicitly from the parameters of the QBD.

The analogous case where upward transitions only occur in a state with phase in $\mathcal{S}^+$ is treated in Section 4. In this case the matrix A_2 has only $r \ll m$ nonzero rows, i.e.,

$$A_2 = \begin{bmatrix} A_2^{++} & A_2^{+-} \\ 0 & 0 \end{bmatrix}. \tag{5}$$

This structure was analyzed by Grassmann and Tavakoli in [8], where it was exploited to reduce the computation time per iteration in the so-called U-algorithm [9], which computes a matrix U such that $R = A_2(I-U)^{-1}$. The algorithm starts with $U_0 = A_1$ and iteratively computes

$U_{k+1} = A_1 + A_2(I - U_k)^{-1}A_0$, such that the iterates converge to the actual value of the matrix U. Even though the approach proposed in [8] provides an important computational gain per iteration, the number of iterations required may be large since this is a linearly-convergent algorithm [18]. In Section 5 we consider an example with the structure described by (5) and compare the performance of our approach with the one proposed in [8]. The Grassmann and Tavakoli method can also be adapted to the case where the matrix A_0 has the form in (4).

2.1. Markov chains of the M/G/1 and GI/M/1 type

An M/G/1-type MC [19] can be seen as a generalization of a QBD MC, where the level is allowed to increase its value by more than one in a single transition. Therefore, the transition matrix $\bar{P}$ of an M/G/1-type MC is of the form

$$\bar{P} = \begin{bmatrix} \bar{B}_0 & \bar{B}_1 & \bar{B}_2 & \bar{B}_3 & \cdots \\ \bar{A}_0 & \bar{A}_1 & \bar{A}_2 & \bar{A}_3 & \cdots \\ & \bar{A}_0 & \bar{A}_1 & \bar{A}_2 & \cdots \\ & & \bar{A}_0 & \bar{A}_1 & \cdots \\ 0 & & & \ddots & \ddots \end{bmatrix},$$

where $\bar{A}_i$, for $i \geq 0$, and $\bar{B}_i$, for $i \geq 0$, are nonnegative matrices in $\mathbb{R}^{b \times b}$ such that $\sum_{i=0}^{+\infty} \bar{A}_i$ and $\sum_{i=0}^{+\infty} \bar{B}_i$ are stochastic. A numerically stable method to find the stationary probability vector of this MC is Ramaswami's formula [20], which depends on the matrix $\bar{G}$, that is the minimal non-negative solution of

$$\bar{G} = \sum_{i=0}^{\infty} \bar{A}_i \bar{G}^i. \tag{6}$$

The quadratically-convergent Cyclic Reduction algorithm can also be applied to solve this equation.

On the other hand, a GI/M/1-type MC [2] can be seen as a QBD where the chain is allowed to decrease several levels in a single transition. The transition matrix for this MC is therefore given by

$$\hat{P} = \begin{bmatrix} \hat{B}_0 & \hat{A}_0 & & & 0 \\ \hat{B}_1 & \hat{A}_1 & \hat{A}_0 & & \\ \hat{B}_2 & \hat{A}_2 & \hat{A}_1 & \hat{A}_0 & \\ \hat{B}_3 & \hat{A}_3 & \hat{A}_2 & \hat{A}_1 & \ldots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix},$$

where $\hat{A}_i$, $i \geq 0$, and $\hat{B}_i$, $i \geq 0$ are nonnegative matrices in $\mathbb{R}^{b \times b}$ such that $\sum_{i=0}^{n} \hat{A}_i + \hat{B}_n$ is stochastic for all $n \geq 0$. In this case the stationary probability vector can be computed as $\pi_i = \pi_0 \hat{R}^i$, where $\hat{R}$ is the minimal non-negative solution to

$$\hat{R} = \sum_{i=0}^{\infty} \hat{R}^i \hat{A}_i. \tag{7}$$

To solve this equation we first compute the dual process, which is of the M/G/1 type, allowing the use of the quadratically-convergent CR algorithm. There are two different duals that can be used for this purpose. A brief description of both is included in Appendix A. Here we have assumed that the boundary level has the same size as all the other levels in both the M/G/1- and GI/M/1-type MCs. A more general boundary can be assumed since our results are related to the behavior of the MCs away from the boundary, which is described by the $(\bar{A}_i)_{i \geq 0}$ or the $(\hat{A}_i)_{i \geq 0}$ matrices.

3. QBDs with restricted downward transitions

In this section we describe how the special structure of the matrix A_0 can be exploited to compute the matrix G. Consider the case where the matrix A_0 has only $r \ll m$ nonzero columns as shown in Equation (4). The (i,j)-th entry of the matrix G holds the probability that the first visit to level $k-1$ occurs by visiting state $(k-1,j)$, starting from state (k,i), for $k > 1$ [16]. Since the downward transitions can only occur to the first r states of any level, the G matrix has the structure

$$G = \begin{bmatrix} G_+ & 0 \\ G_0 & 0 \end{bmatrix},$$

where G_+ (resp. G_0) is an $r \times r$ (resp. $(m-r) \times r$) matrix. The computation of G_+ and G_0 will be split in two steps such that, for $r \ll m$, the total computation time can be significantly reduced.

3.1. Computing G_+

To compute G_+ we define a new process by observing the QBD MC only when the phase variable is in the set $\mathcal{S}^+$. In the original process any transition to a lower level triggers the phase to a state in $\mathcal{S}^+$, therefore the new process can only move one level down at each transition. On the other hand, the original process can move several levels upward while the phase is in $\mathcal{S}^-$, i.e., between two visits to $\mathcal{S}^+$. Therefore the new process can move several levels up in one transition, but only one level down. Hence, the new process is of the M/G/1 type and its behavior away from the boundary is characterized by a set of $r \times r$ matrices $(\bar{A}_i)_{i\geq 0}$. The minimal nonnegative solution $\bar{G}$ of Equation (6) is actually equal to the matrix G_+. This follows from the definition of the matrix $\bar{G}$ as the first passage probability to the state $(k-1,j)$, starting from state (k,i), in the new process, and the fact that in the original process the downward transitions can only lead to $\mathcal{S}^+$. Hence, to compute the matrix G_+ we first need to determine the $r \times r$ blocks $(\bar{A}_i)_{i\geq 0}$ and then solve Equation (6).

To specify the blocks $(\bar{A}_i)_{i\geq 0}$, let the (i,j)-th entry of the $(m-r)\times r$ matrix K_l hold the probability that, given that the original process starts in state (k,i), with $i \in \mathcal{S}^-$, its first transition to a state with phase in $\mathcal{S}^+$ occurs to the state $(k+l,j)$, for $j \in \mathcal{S}^+$, $k > 1$ and $l \in \{-1,0,1,...\}$. Hence, the matrices $(K_i)_{i\geq -1}$ are given by

$$\begin{aligned} K_{-1} &= (I - A_1^{--})^{-1}A_0^{-+}, \\ K_0 &= (I - A_1^{--})^{-1}(A_1^{-+} + A_2^{--}K_{-1}), \\ K_1 &= (I - A_1^{--})^{-1}(A_2^{-+} + A_2^{--}K_0), \\ K_i &= (I - A_1^{--})^{-1}A_2^{--}K_{i-1}, \quad i \geq 2. \end{aligned} \tag{8}$$

To define K_{-1} we observe that the chain starts in level k and spends some time in the states of this level with phase in $\mathcal{S}^-$. Afterward the chain has to move to a state $(k-1,j)$, with $j \in \mathcal{S}^+$. The only other possible state that the chain could visit after its sojourn in level k, avoiding states with phase in $\mathcal{S}^+$, is to move to a state in level $k+1$ and phase in $\mathcal{S}^-$. However, for the chain to visit level $k-1$ it first has to go back from level $k+1$ to level k, and this can only be done through a state with phase in $\mathcal{S}^+$. Therefore, this path is not possible if the first state with phase in $\mathcal{S}^+$ to be visited must be in level $k-1$. The definition of the other matrices can be understood in a similar manner. Now we can define the blocks $(\bar{A}_i)_{i\geq 0}$ in terms of the matrices $(K_i)_{i\geq -1}$ as

$$\begin{aligned} \bar{A}_0 &= A_0^{++} + A_1^{+-}K_{-1}, \\ \bar{A}_1 &= A_1^{++} + A_1^{+-}K_0 + A_2^{+-}K_{-1}, \\ \bar{A}_2 &= A_2^{++} + A_1^{+-}K_1 + A_2^{+-}K_0, \\ \bar{A}_i &= A_1^{+-}K_{i-1} + A_2^{+-}K_{i-2}, \quad i \geq 3. \end{aligned} \tag{9}$$

To define $\bar{A}_0$ we see that the transition from a state (k,i) to a state $(k-1,j)$, with $i,j \in \mathcal{S}^+$, can only occur in two ways: either the chain goes directly to $(k-1,j)$ with transition matrix A_0^{++}; or it moves first to a state in level k with phase in $\mathcal{S}^-$ and, after a sojourn in these states, it moves downward avoiding other states in $\mathcal{S}^+$ (with transition matrix $A_1^{+-}K_{-1}$). A transition to level

$k+1$ is not allowed since the chain cannot return to $k-1$ without passing through a state in level k with phase in $\mathcal{S}^+$. The other matrices can be defined similarly. Notice, to compute the matrices $\bar{A}_i$ it suffices to store two K_i matrices at a time. The $r \times r$ matrices $\bar{A}_i$ are sequentially computed from $i = 0$ to c, where c is the smallest positive integer such that $\sum_{i=0}^{c} \bar{A}_i e > (1-\epsilon)e$, with e a column vector of ones and $\epsilon = 10^{-14}$. These blocks can then be used to compute the matrix G_+ using the CR algorithm [3].

3.2. Computing G_0

Given the structure of the matrices A_0 and G we can rewrite Equation (2) as

$$\begin{bmatrix} G_+ & 0 \\ G_0 & 0 \end{bmatrix} = \begin{bmatrix} A_0^{++} & 0 \\ A_0^{-+} & 0 \end{bmatrix} + \begin{bmatrix} A_1^{++} & A_1^{+-} \\ A_1^{-+} & A_1^{--} \end{bmatrix} \begin{bmatrix} G_+ & 0 \\ G_0 & 0 \end{bmatrix} + \begin{bmatrix} A_2^{++} & A_2^{+-} \\ A_2^{-+} & A_2^{--} \end{bmatrix} \begin{bmatrix} G_+^2 & 0 \\ G_0 G_+ & 0 \end{bmatrix}. \qquad (10)$$

Extracting the lower-left block we find

$$G_0 - (I - A_1^{--})^{-1} A_2^{--} G_0 G_+ = (I - A_1^{--})^{-1} (A_0^{-+} + A_1^{-+} G_+ + A_2^{-+} G_+^2), \qquad (11)$$

which is a Sylvester matrix equation [21, 22] of the type $AXB + X = E$, that can be solved in $O((m-r)^3)$ time with the Hessenberg-Schur method proposed in [21]. A brief description of this method is included in Appendix B together with a discussion on some additional considerations that influence the computation time of G_0. Next, we consider two special cases where additional restrictions on the transition probabilities allow us to limit the number of blocks of the reduced process, further reducing the computation times.

3.3. Restricted downward transitions and $A_2^{--} = 0$

Let the matrix A_0 have the structure shown in Equation (4). Additionally, assume that upward transitions from states with phase in $\mathcal{S}^-$ take the process to a state with phase in $\mathcal{S}^+$, i.e., the matrix A_2 has the form

$$A_2 = \begin{bmatrix} A_2^{++} & A_2^{+-} \\ A_2^{-+} & 0 \end{bmatrix}.$$

With this additional structure, the maximum number of upward transitions between two visits to $\mathcal{S}^+$ is two, since an upward transition from $\mathcal{S}^-$ must end in $\mathcal{S}^+$. Therefore the reduced process of the M/G/1 type, constructed by observing the original process when the phase is in $\mathcal{S}^+$, has only four nonzero blocks defined as

$$\begin{aligned}
\bar{A}_0 &= A_0^{++} + A_1^{+-}(I - A_1^{--})^{-1}A_0^{-+},\\
\bar{A}_1 &= A_1^{++} + A_1^{+-}(I - A_1^{--})^{-1}A_1^{-+} + A_2^{+-}(I - A_1^{--})^{-1}A_0^{-+},\\
\bar{A}_2 &= A_2^{++} + A_1^{+-}(I - A_1^{--})^{-1}A_2^{-+} + A_2^{+-}(I - A_1^{--})^{-1}A_1^{-+},\\
\bar{A}_3 &= A_2^{+-}(I - A_1^{--})^{-1}A_2^{-+}.
\end{aligned}$$

The definition of these blocks can be obtained directly from equations (8) and (9) as follows: $A_2^{--} = 0$ implies that $K_i = 0$ for $i \geq 2$, which therefore means that $\bar{A}_i = 0$ for $i > 3$. Additionally, the fact that $A_2^{--} = 0$ also simplifies the expressions for K_0 and K_1, which are used in the definition of the matrices $\bar{A}_1$, $\bar{A}_2$ and $\bar{A}_3$. This additional structure reduces both the time to compute the blocks and the time to find G_+ using CR. Additionally, to find G_0 we consider again Equation (10) and, by extracting its lower-left block, we find

$$G_0 = (I - A_1^{--})^{-1}\left(A_0^{-+} + A_1^{-+}G_+ + A_2^{-+}G_+^2\right).$$

Therefore, there is no need for solving a Sylvester matrix equation, as was done before, as G_0 can be determined directly from G_+ and other already computed matrices. With this additional constraint the problem of finding the $m \times m$ matrix G is replaced by the determination of just four $r \times r$ matrices and the solution of Equation (6) using these smaller matrices.

3.4. Restricted downward and upward transitions

Now we assume that the matrices A_0 and A_2 of the QBD have the structure described in equations (4) and (5), respectively. In this case, the process obtained by observing the QBD when the phase is in the set $\mathcal{S}^+$ is again a QBD with parameters

$$\begin{aligned}
\bar{A}_0 &= A_0^{++} + A_1^{+-}(I - A_1^{--})^{-1}A_0^{-+},\\
\bar{A}_1 &= A_1^{++} + A_1^{+-}(I - A_1^{--})^{-1}A_1^{-+} + A_2^{+-}(I - A_1^{--})^{-1}A_0^{-+},\\
\bar{A}_2 &= A_2^{++} + A_2^{+-}(I - A_1^{--})^{-1}A_1^{-+}.
\end{aligned}$$

To obtain these expressions, in addition to the simplifications due to $A_2^{--} = 0$ explained above, we notice that K_1 becomes zero since both A_2^{-+} and A_2^{--} are equal to zero. Hence $\bar{A}_3$ also becomes zero and the resulting process is again a QBD (of a smaller block size). Moreover, the matrix G_0 is given by

$$G_0 = (I - A_1^{--})^{-1} \left(A_0^{-+} + A_1^{-+} G_+ \right).$$

The reduction in computation time is evident since now it is enough to find the solution to Equation (2) with matrices of size r instead of m. The number of matrix multiplications required to compute the blocks of the new QBD process and the matrix G_0 is fixed and small compared to the solution of Equation (2).

4. QBDs with restricted upward transitions

We now turn to the case where the matrix A_2 has only $r \ll m$ nonzero rows as in Equation (5), restricting the upward transitions to occur only when the phase variable is in $\mathcal{S}^+$, while A_0 is no longer in the form (4). In a QBD the (i,j)-th entry of the rate matrix R from Equation (1) can be interpreted as the expected number of visits to the state $(k+1, j)$, starting from state (k, i), before visiting any other state at level k [2]. To visit a state in level $k+1$ starting from level k, while avoiding level k, the first transition must take the chain from level k to level $k+1$. However, due to the structure of A_2, no upward transition can be made if the phase variable is in $\mathcal{S}^-$. Hence the last $m - r$ rows of the matrix R are equal to zero, and R can be written as

$$R = \begin{bmatrix} R_+ & R_0 \\ 0 & 0 \end{bmatrix},$$

where R_+ and R_0 are matrices of size $r \times r$ and $r \times (m - r)$, respectively. In a similar way as in the previous case, we define a new process by observing the original QBD MC when the phase variable is in $\mathcal{S}^+$. In this case the level cannot increase in the phases outside $\mathcal{S}^+$, but it can decrease several levels between two visits to $\mathcal{S}^+$. Therefore, the new process is a Markov chain of the GI/M/1 type. Using this process we can find the matrices R_+ and R_0 separately, as shown next.

4.1. Computing R_+

The behavior of the censored process, obtained by observing the original QBD MC when the phase is in $\mathcal{S}^+$, is characterized away from the boundary by the set of $r \times r$ matrices $(\hat{A}_i)_{i \geq 0}$. Let $\hat{R}$ be the minimal nonnegative solution of the Equation (7). Then the (i,j)-th entry of the matrix

$\hat{R}$ can be interpreted as the expected number of visits to state $(k+1,j)$, starting from state (k,i), before the first return to level k [2], for $(i,j) \in \mathcal{S}^+$ and $k > 1$. This is the same interpretation as the (i,j)-th entry of R_+; therefore $R_+ = \hat{R}$. To find $\hat{R}$ we first need to specify the blocks $(\hat{A}_i)_{i \geq 0}$, which is done in terms of the matrices $(W_{-i})_{i \geq 0}$.

Let the entry (i,j) of the $(m-r) \times r$ matrix W_{-l} be the probability that, given that the original process starts in state (k,i) with $i \in \mathcal{S}^-$, its first transition to a state with phase in the set $\mathcal{S}^+$ occurs in the state $(k-l,j)$, for $j \in \mathcal{S}^+$, $k > l \geq 0$. Hence, the matrices $(W_{-i})_{i \geq 0}$ are given by

$$\begin{aligned} W_0 &= (I - A_1^{--})^{-1} A_1^{-+}, \\ W_{-1} &= (I - A_1^{--})^{-1} (A_0^{-+} + A_0^{--} W_0), \\ W_{-i} &= (I - A_1^{--})^{-1} A_0^{--} W_{-(i-1)}, \quad i \geq 2. \end{aligned}$$

The blocks $(\hat{A}_i)_{i \geq 0}$ can be defined in terms of the matrices $(W_{-i})_{i \geq 0}$ as

$$\begin{aligned} \hat{A}_0 &= A_2^{++} + A_2^{+-} W_0, \\ \hat{A}_1 &= A_1^{++} + A_1^{+-} W_0 + A_2^{+-} W_{-1}, \\ \hat{A}_2 &= A_0^{++} + A_0^{+-} W_0 + A_1^{+-} W_{-1} + A_2^{+-} W_{-2}, \\ \hat{A}_i &= A_0^{+-} W_{-i+2} + A_1^{+-} W_{-i+1} + A_2^{+-} W_{-i}, \quad i \geq 3. \end{aligned}$$

The blocks $\hat{A}_i$ are computed from $i = 0$ to c, where c is the smallest positive integer such that $\sum_{i=0}^{c} \hat{A}_i e > (1-\epsilon)e$. In this case it suffices to keep track of the three matrices $\{W_{-i+2}, W_{-i+1}, W_{-i}\}$ when computing the matrix A_i. As stated before, we need to compute the dual process of the GI/M/1-type MC characterized by $(\hat{A}_i)_{i \geq 0}$ in order to apply the CR algorithm. We use the dual relationship to compute the M/G/1-type blocks and, after solving a matrix equation of the type (6), retrieve R_+ from the G matrix of the dual. Since there are two different duals that can be used (see Appendix A), we consider both alternatives and compare their performance in Section 5.

4.2. Computing R_0

By writing Equation (1) in block form and extracting the upper-right corner, we find

$$R_0 - R_+ R_0 A_0^{--} (I - A_1^{--})^{-1} = (A_2^{+-} + R_+ A_1^{+-} + R_+^2 A_0^{+-})(I - A_1^{--})^{-1}. \tag{12}$$

This is also a Sylvester matrix equation of the type $AXB + X = E$, which can be solved in $O((m-r)^3)$ time using the Hessenberg-Schur method proposed in [21] (see Appendix B).

4.3. Restricted upward transitions and $A_0^{--} = 0$

When the matrix A_2 of the QBD MC has only r nonzero rows, as in Equation (5), and additionally the block A_0^{--} is equal to zero, we can further improve the new algorithm in a manner similar to Section 3.3. We omit the details as both cases are analogous.

5. Examples

In this section we consider four different continuous-time queueing systems in which the structures analyzed in the previous sections arise (and a standard uniformization argument is applied to transform the problem to discrete time when necessary). We start by considering a priority queue with two customer classes that can be modeled as a QBD process with restricted downward transitions. Next we present a general MAP/PH/1 queue (see definitions below), which can be modeled as a QBD process that can be induced to have restricted downward transitions. Then we illustrate the case of a QBD process with restricted upward transitions through an overflow queue. Finally, we consider the model of a relay node in a wireless network introduced in [14], where the QBD process used to evaluate the node's performance also falls within our framework. In all these cases cases we compare the times required to compute the R or G matrix using the full-size QBD and the approach proposed in this paper. For the overflow queue we also compare with the approach introduced in [8], while for the relay node model we include a comparison with the method proposed in [14].

Before describing the examples in detail we need to introduce the continuous-time Markovian Arrival Process (MAP) and the Phase-Type (PH) distribution [16], since these are used in our examples to model the arrival processes and the service-time distributions. A MAP is a point process characterized by the parameters (n, D_0, D_1), where n is a positive integer, and D_0 and D_1 are $n \times n$ matrices. This process is driven by an underlying MC with generator matrix $D = D_0 + D_1$, where D_1 and D_0 contain the intensities associated to transitions with and without arrivals, respectively. The off-diagonal entries of D_0 and all the entries of D_1 must be non-negative, while the diagonal entries of D_0 must be negative and such that $(D_0 + D_1)e = 0$. Let γ be the stationary distribution of the underlying MC, i.e., a $1 \times n$ vector such that $\gamma D = 0$ and $\gamma e = 1$. The arrival rate of the MAP is given by $\gamma D_1 e$. This process can be generalized by introducing markings to discriminate among different types of customers. In the forthcoming examples it is enough to consider a marked MAP (MMAP) with two types of customers. In addition to the parameters n and D_0, the MMAP is characterized by the matrices D_1 and D_2, which hold the transition intensities associated with an arrival of type 1 and 2, respectively. In this case the underlying MC has generator matrix $D = D_0 + D_1 + D_2$ and the arrival rate of customers of type i is equal to $\gamma D_i e$, for $i = 1, 2$.

A PH distribution is characterized by the triple (n, α, T), where n is a positive integer, α is a $1 \times n$ vector and T a square matrix of size n. A PH distribution describes the absorption time in an MC where the states $\{1, \ldots, n\}$ are transient and an additional state, say $n+1$, is absorbing. The initial probability distribution of the transient states is given by α, while T is the sub-generator matrix of these states. Therefore, the j-th entry of the vector $t = -Te$ holds the absorption rate from state j, for $1 \leq j \leq n$. The cumulative distribution function of a PH variable is given by $F(x) = 1 - \alpha \exp(Tx)e$, for $x \geq 0$. For further reference recall that the Kronecker product of the matrices A and B, denoted $A \otimes B$, is the block matrix with block (i, j) equal to $A_{ij}B$ [23]. The Kronecker sum $A \oplus B$ is defined as $A \otimes I + I \otimes B$, where I is an identity matrix of appropriate size.

5.1. Priority Queue

Our first example is a continuous-time priority queue with two classes of customers. Class-1 customers have preemptive priority over class-2 customers. Therefore, customers of class 2 can only be served if there are no class-1 customers in the queue, and the service of a class-2 customer is interrupted if a customer of class 1 arrives. The high-priority arrivals are described by a MAP characterized by (m_a^1, C_0^1, C_1^1) while the MAP of the low-priority arrivals has parameters (m_a^2, C_0^2, C_1^2). These two processes can be combined in a single marked MAP with parameters $D_0 = C_0^1 \oplus C_0^2$, $D_1 = C_1^1 \otimes I$ and $D_2 = I \otimes C_1^2$, where D_0, D_1 and D_2 are square matrices of size $m_a = m_a^1 m_a^2$. The service times of class-1 (resp. class-2) customers follow a PH distribution with parameters (m_s^1, α, T) (resp. (m_s^2, β, S)). To model this queue as a QBD with restricted downward transitions we take the level as the number of low-priority customers in the queue, and assume a finite buffer of size C for the class-1 customers. This assumption places no restriction in the analysis since this buffer can be dimensioned such that the blocking probability of the high-priority customers is below a certain threshold, allowing us to truncate its infinite size. Given the preemptive nature of the priority queue, this can be done using a QBD MC that ignores the low-priority customers. The second dimension of the QBD therefore holds the number of class-1 customers, the phase of the arrival process, and the phase of the customer in service. In addition, if there is a class-1 customer in service, the service phase includes both the current phase of the customer in service and the phase in which the next class-2 customer will (re-)start its (possibly preempted) service. This is not necessary if the customer in service is of class 2, since in that case there are zero class-1 customers in the system. Therefore the blocks have size $m = m_a m_s^2(1+Cm_s^1)$ and are given by

$$A_0 = \begin{bmatrix} I \otimes s\beta & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} D_2 \otimes I_{m_s^2} & 0 & \dots & 0 \\ 0 & D_2 \otimes I_{m_s} & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & D_2 \otimes I_{m_s} \end{bmatrix},$$

$$A_1 = \begin{bmatrix} D_0 \oplus S & D_1 \otimes I \otimes \alpha & 0 & \dots & 0 \\ I \otimes t & (D_0 \otimes I) \oplus T & D_1 \otimes I & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & (D_0 \otimes I) \oplus T & D_1 \otimes I \\ 0 & 0 & \dots & I \otimes t\alpha & (D \otimes I) \oplus T \end{bmatrix},$$

where $D = D_0 + D_1$, $t = -Te$, $s = -Se$, $m_s = m_s^1 + m_s^2$, and the size of the identity matrix has been included in those places where it might be unclear from the context. Since low-priority service completions can only occur when there are no high-priority customers in the queue, downward transitions are limited to occur when the process is in one of the first $r = m_a m_s^2$ phases, and these transitions trigger the process to the same set of phases. Therefore the structure of A_0 can be exploited as shown in Section 3.

For the numerical results shown next we consider a high-priority buffer of size $C = 50$ and, for both customer classes, hyper-exponential service times with mean one and squared coefficient of variation (SCV) equal to two. The parameters of the service distribution are computed using the moment-matching method in [24], that results in a PH representation of order 2. The arrival processes are built using the method in [25, 26], which allows the matching of the first two moments of the inter-arrival distribution and the decay rate of the autocorrelation function γ with a MAP of size 2. In this case both MAPs have the same mean, fixed by the load ρ, and SCV equal to five. For this queue the load is given by $\rho = \lambda_1/\mu_1 + \lambda_2/\mu_2$, where λ_i and μ_i are the arrival and service rates of the type-i customers, respectively, for $i = 1, 2$. Since the service rates are equal to one and the arrival rates are equal, then $\lambda_1 = \lambda_2 = \rho/2$. We consider two scenarios, in the first the inter-arrival times are independent ($\gamma = 0$), while in the second γ is equal to 0.9. With this set of parameters the block size is 808 while the number of nonzero columns in A_0 is 8.

Table 1: Computation times (sec) for the priority queue with $\gamma = 0$

ρ	QBD-CR	Bl	# Bl	MG1-CR	Sylv	MG1	Ratio
0.1	35.34	1.33	18	0.09	3.45	4.87	7.3
0.2	35.33	1.41	31	0.11	1.75	3.27	10.8
0.3	40.49	1.53	48	0.11	3.45	5.09	7.9
0.4	40.48	1.67	70	0.14	1.74	3.55	11.4
0.5	45.64	1.84	98	0.17	3.47	5.48	8.3
0.6	45.64	2.09	136	0.31	1.73	4.14	11.0
0.7	50.81	2.42	184	0.25	1.75	4.42	11.5
0.8	50.81	2.84	248	0.28	1.73	4.86	10.5
0.9	61.14	3.42	334	0.48	1.75	5.66	10.8

In Table 1 we show the time required to compute the matrix G using the full-size QBD with the CR algorithm (QBD-CR), the time to compute the M/G/1-type blocks (Bl), the number of those blocks (# Bl), the time to compute the matrix G_+ with CR (MG1-CR) and the time to solve the Sylvester matrix equation to get G_0 (Sylv). The total computation time using the reduced process is shown in column MG1, and the last column has the ratio between the columns QBD-CR and MG1. Clearly, the M/G/1-type based method outperforms the full-size approach, which can take 7 to 11 times longer to compute G. Also, when the load ρ increases, both methods require more computation time, particularly the CR algorithm for the QBD and the computation of the M/G/1-type blocks. A large load has two major effects: first, it increases the rate of upward transitions per time unit; second, since the set $\mathcal{S}^+$ includes only the phases in which there are no high-priority customers in the queue, a larger load increases the likelihood of having long sojourn times in $\mathcal{S}^-$. These two effects together imply that the number of blocks to compute increases, and the CR algorithm requires more time to solve Equation (6). In contrast, the Hessenberg-Schur method to solve Equation (11) seems to be less sensitive to the load of the queue.

Table 2 contains the same information as the previous one, but in this scenario the arrival processes are highly autocorrelated, with decay rate of the autocorrelation function $\gamma = 0.9$. As can be observed, the correlation, together with the load, has a large effect on the number of M/G/1-type blocks that describe the reduced process, and therefore on the time required to compute those blocks and to find G_+. On the other hand, the correlation has little effect on the time to find G_0 with the Hessenberg-Schur method. We see that the reduced process still offers a reduction in computation times but this gain is affected by the system parameters. A similar behavior will be observed in the subsequent examples.

Table 2: Computation times (sec) for the priority queue with $\gamma = 0.9$

ρ	QBD-CR	Bl	# Bl	MG1-CR	Sylv	MG1	Ratio
0.1	35.33	1.34	19	0.08	1.74	3.16	11.2
0.3	40.50	1.61	61	0.13	1.73	3.47	11.7
0.5	50.80	2.28	161	0.25	1.74	4.27	11.9
0.7	61.11	3.95	413	0.52	3.47	7.94	7.7
0.9	71.45	8.24	1032	1.64	1.74	11.61	6.2

In addition to the computation times, it is relevant to consider the behavior of the approach introduced in this paper in terms of the residual error. Let the infinity norm of an $n \times m$ matrix K be given by $||K||_\infty = \max_{i=1}^n \sum_{j=1}^m K_{ij}$. Let $\tilde{G}$ be the matrix that solves Equation (2) obtained with the approach of Section 3. Then the residual error is defined as

$$||\tilde{G} - A_0 + A_1\tilde{G} + A_2\tilde{G}^2||_\infty,$$

which gives a measure of the goodness of $\tilde{G}$ as a solution for Equation (2). In all the instances considered here the residual error was always below 10^{-14}, revealing the good behavior of the approach proposed. This behavior is to be expected since the algorithms on which our method relies (Cyclic Reduction and the Hessenberg-Schur method for the Sylvester equation) are numerically stable. A similar result in terms of the residual error holds for the other examples.

5.2. The MAP/PH/1 queue

The MAP/PH/1 queue receives customers according to a MAP with parameters (m_a, D_0, D_1), which are processed by a single server, and the service time is described by a PH distribution characterized by (m_s, α, T). This queue can be modeled as a QBD MC by choosing the number of customers in the queue to be the level. This selection assures that the level increases and decreases by at most one in a single transition since only one service completion or a single arrival can occur at a time. The state space of this MC can be described as follows: the level zero is the set of states $\Omega_0 = \{(0, j), 1 \leq j \leq m_a\}$, where in state $(0, j)$ the queue is empty and the arrival process is in phase j; the level $k \geq 1$ is the set of states $\Omega_k = \{(k, i, j), 1 \leq i \leq m_s, 1 \leq j \leq m_a\}$, where in state (k, i, j) there are k customers in the queue, the service in progress is in phase i and the arrival process is in phase j. The complete state space is therefore given by $\Omega = \bigcup_{k \geq 0} \Omega_k$. The blocks of this QBD MC are given by

$$A_0 = t\alpha \otimes I_{m_a}, \quad A_1 = T \oplus D_0, \quad A_2 = I_{m_s} \otimes D_1, \tag{13}$$

where $t = -Te$ and I_n is the identity matrix of size n. From this definition it is clear that the block size is $m = m_s m_a$, and that the number of nonzero columns in A_0 depends on the number of nonzero elements in the vector α. In fact, if α has only one nonzero element, then A_0 has only $r = m_a$ nonzero columns, i.e., the block size is m_s times larger than the number of nonzero columns in A_0. This is the case if the service times are described by an Acyclic PH distribution (APH) [27]. This class of distributions (which includes the Erlang and the hyper-exponential distributions as special cases) has a canonical form, introduced in [27], where all the mass of the initial probability vector is concentrated in the first phase. Therefore, in this case the vector α has only one nonzero entry and the matrix A_0 has m_a nonzero columns. In general, the vector α may have any number of nonzero entries, but we can always find a representation of size $m_s + 1$ such that the initial probability vector has only one nonzero entry, as shown in the next theorem.

Theorem 1. *Any continuous PH distribution with representation (m_s, α, T) also has a representation $(m_s + 1, e_1, \bar{T})$, where e_1 and $\bar{T}$ are given by*

$$e_1 = [1 \ 0_{m_s}] \quad \textit{and} \quad \bar{T} = \begin{bmatrix} -c & c\alpha P \\ 0 & T \end{bmatrix},$$

where 0_n is the $1 \times n$ zero vector, c is the diagonal entry of T of largest absolute value, i.e., $c = \max\{|T_{ii}|, 1 \leq i \leq m_s\}$, and P is the uniformized version of the subgenerator matrix T, i.e., $P = \frac{1}{c}T + I_{m_s}$.

Proof. We start by uniformizing the absorbing MC that underlies the PH distribution characterized by (m_s, α, T). Since the rate corresponding to the absorbing state is zero, we can use c to uniformize the chain and, therefore, P holds the transition probabilities among the transient states in the uniformized chain. Also, let $\bar{P}$ be the uniformized version of the subgenerator $\bar{T}$, which is equal to

$$\bar{P} = \frac{1}{c}\bar{T} + I_{m_s+1} = \begin{bmatrix} 0 & \alpha P \\ 0 & P \end{bmatrix}.$$

Now we can write the CDF of the new representation $G(\cdot)$ as

$$\begin{aligned}
G(x) &= 1 - e_1 \exp(\bar{T}x)e = 1 - e_1 \sum_{n\geq 0} \frac{x^n}{n!}\bar{T}^n e = 1 - e_1 \sum_{n\geq 0} \frac{(cx)^n}{n!}(\bar{P} - I_{m_s+1})^n e, \\
&= 1 - e_1 \sum_{n\geq 0} \frac{(cx)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} \bar{P}^k (-I_{m_s+1})^{n-k} e, \\
&= 1 - \sum_{n\geq 0} \frac{(cx)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} e_1 \begin{bmatrix} 0 & \alpha P^k \\ 0 & P^k \end{bmatrix} (-I_{m_s+1})^{n-k} e, \\
&= 1 - \sum_{n\geq 0} \frac{(cx)^n}{n!} \sum_{k=0}^{n} \binom{n}{k} \alpha P^k (-I_{m_s})^{n-k} e, \\
&= 1 - \alpha \sum_{n\geq 0} \frac{(cx)^n}{n!} (P - I_{m_s})^n e, \\
&= 1 - \alpha \exp(Tx)e, \quad x \geq 0,
\end{aligned}$$

which is equal to the CDF of the original representation. Therefore (m_s, α, T) and $(m_s + 1, e_1, \bar{T})$ are two different PH representations of the same distribution. A similar result holds for discrete PH distributions. □

Using this result we can replace α and T by e_1 and $\bar{T}$, respectively, in Equation (13). As a consequence the block A_0 has only $r = m_a$ nonzero columns, and the new block size is $(m_s+1)m_a$, which is exactly the structure we have referred to as restricted downward transitions. To illustrate the applicability of this result we consider a specific case of a MAP/PH/1 queue, namely a system that provides reliable messaging services. In particular, we consider the Web Services Reliable Messaging (WSRM) protocol, which is used to ensure message transmission in web-based service oriented architectures [28]. This protocol has been analyzed in [29], and there the authors have used PH distributions to approximate the effective transmission time in a WSRM implementation. They consider different methods to obtain the PH representation, which is then used as input in an M/PH/1 queue that models the arrival and transmission of messages over WSRM. Here we consider the more general case where the arrivals are modeled as the combination of one, two or three streams, each one represented by a MAP. The transmission times are represented by a hyper-Erlang distribution with $m_s = 153$ phases, which corresponds to the case S_{2JK} considered in [29]. The parameters of this distribution were downloaded from [30]. Although this distribution is acyclic, its initial probability vector has many nonzero entries. As stated before, it is possible to use the results in [27] to obtain a canonical representation where the initial probability vector has a single nonzero entry. However, we have opted for using Theorem 1 to illustrate the computational gains obtained by exploiting the restricted-transitions structure, even if a slightly larger representation of the service process is needed to induce that structure.

Table 3: Computation times (sec) for the MAP/PH/1 queue

	QBD-CR			MG1			Ratio		
r ρ	2	4	8	2	4	8	2	4	8
0.1	2.3	17.8	140.2	0.3	1.5	11.9	8.8	12.0	11.8
0.3	2.6	20.2	157.9	0.3	2.0	15.5	7.7	10.2	10.2
0.5	3.0	22.5	175.5	0.4	2.4	18.1	7.1	9.5	9.7
0.7	3.3	24.8	193.2	0.5	2.7	20.1	6.6	9.3	9.6
0.9	3.9	29.4	228.5	0.6	3.2	24.2	6.8	9.1	9.4

As stated above, the arrivals come from the superposition of one, two or three sources, each one represented by a MAP of size two. This implies that the size m_a of the arrival process representation, and the number of nonzero columns r, is equal to two, four and eight, respectively. The total arrival rate λ is set to match a given load $\rho = \lambda/\mu$, where μ is the mean transmission rate. The total arrival rate is equally divided among all the sources, while each of them has SCV equal to five and the decay of their autocorrelation function is set at 0.5. These characteristics are matched by using the method introduced in [25, 26]. Table 3 shows the times required to compute the matrix G using CR directly on the blocks of size $m = m_s m_a$ (QBD-CR), and using the approach introduced in this paper to exploit the (induced) restricted-transitions structure (MG1). It also shows the ratio between these two times (Ratio), which tells us how many times slower the general approach is compared to our specific method. We observe how in both cases the computation times are affected by the increase in the size of the arrival process representation, as is to be expected since this size affects both the original block size m and the number of nonzero columns. When there is a single source ($r = m_a = 2$), the QBD-CR method is between 6 and 9 times slower than MG1. When the number of sources increases to 2 and 3, this figure increases to between 9 and 12. We also notice that the difference is larger for small loads and, although both methods are negatively affected by the increase of the load, this parameter has a larger effect on the MG1 method. In this method, a larger load implies the computation of a larger number of blocks in the censored process, which also means that it is necessary to solve Equation (6) with a larger number of nonzero coefficients. These procedures are therefore affected by the load, while finding G_0 by solving Equation (11) is almost insensitive to this parameter.

5.3. Overflow Queue

We now consider an overflow queueing system consisting of two queues. The arrival process to the first queue is a MAP characterized by (m_a, D_0, D_1). Customers arriving at the first queue are attended in FCFS order by a single server with service times following a PH distribution characterized by the parameters (m_s^1, α, T). This queue has a finite buffer of size C and a customer that finds the buffer full is sent to the second queue. The second queue receives *only* overflow arrivals from the first queue and attends them in FCFS order with a single server. The service times in this queue follow a PH distribution with parameters (m_s^2, β, S). Hence, the arrival process at the second queue can be described by a MAP with parameters (m_o, C_0, C_1) given by $m_o = (C+1)m_a m_s^1$,

$$C_0 = \begin{bmatrix} D_0 \otimes I & D_1 \otimes I & 0 & \dots & 0 & 0 \\ I \otimes t\alpha & D_0 \oplus T & D_1 \otimes I & \dots & 0 & 0 \\ 0 & I \otimes t\alpha & D_0 \oplus T & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & D_0 \oplus T & D_1 \otimes I \\ 0 & 0 & 0 & \dots & I \otimes t\alpha & D_0 \oplus T \end{bmatrix}, C_1 = \begin{bmatrix} 0 & \dots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & D_1 \otimes I \end{bmatrix},$$

where $t = -Te$. Assuming an infinite buffer at the second queue, we can model the queueing system as a QBD where the level describes the number of customers in the second queue. The second dimension holds the phase of the current customer in service and the phase of the arrival process at the second queue. The parameters of the QBD are $A_0 = I \otimes s\beta$, $A_1 = C_0 \oplus S$, $A_2 = C_1 \otimes I$, with $s = -Se$. In this case, the restricted upward transitions are a result of the overflow process, as can be seen in the structure of C_1, which clearly shows that the arrivals to the second queue can only occur in the last $m_a m_s^1$ phases. The inclusion of a separate arrival stream directed to the second queue would suppress this structure. The block size in this case is $m = m_o m_s^2$ and the number of nonzero rows in A_2 is $r = m_a m_s^1 m_s^2$.

As with the previous examples, we make use of the moment-matching methods in [24, 25, 26] to obtain PH and MAP representations of the service and arrival processes, respectively. The arrival process at the first queue has arrival rate and SCV equal to five, while the service time has mean one and SCV equal to two. Therefore the first queue is heavily loaded and many customers are overflowed to the second queue. The arrival rate at the second queue (λ_2) is the arrival rate of the MAP with parameters (C_0, C_1). Therefore for a given load at the second queue (ρ_2) the service rate at this queue is fixed by the relation $\rho_2 = \lambda_2/\mu_2$. In this queue the service times have SCV equal to two, as in the first queue. The results are presented for different values of ρ_2 and a buffer size of 100 in the first queue. With these parameters the block size is $m = 808$ while the number of nonzero rows in A_2 is $r = 8$.

Table 4: Computation times (sec) for the overflow queue with $C = 100$

ρ_2	QBD-CR	Bl	# Bl	G-CR-R	G-CR-B	Sylv	GM1-R	GM1-B	Ratio-R	Ratio-B
0.1	31.91	24.92	2791	17.11	8.14	2.25	44.28	35.31	0.72	0.90
0.3	42.3	8.63	980	2.33	0.53	2.25	13.20	11.41	3.20	3.71
0.5	47.38	5.63	601	1.17	0.64	3.94	10.73	10.2	4.41	4.64
0.7	52.47	4.41	442	0.81	0.58	2.24	7.45	7.22	7.04	7.27
0.9	62.67	3.70	350	0.66	0.45	2.23	6.59	6.39	9.51	9.81

Table 4 shows the computation times in a similar fashion as in Section 5.1, the main difference being that the column G-CR-R (resp. G-CR-B) includes the time to compute the blocks of the Ramaswami (resp. Bright) dual process and the time to solve the dual with the CR algorithm. The columns GM1-R and GM1-B show the total computation times to find R using the two different duals, while the columns Ratio-R and Ratio-B hold the ratio between the QBD-CR and the GM1-R and GM1-B columns, respectively. Again, the load has an important effect on the computation times, but in this case the consequences are reversed. When the load is low the original process can make many downward transitions between two visits to the set $\mathcal{S}^+$, increasing the number of GI/M/1-type blocks. As before, a large number of blocks increases the computation time of the CR algorithm for the M/G/1-type MC (the dual process), but it has little effect on the solution of the Sylvester equation and the full-size QBD. For loads between 0.2 and 0.9 in this scenario, the solution of the full-size QBD may take between 2 and 10 times as long as the solution of the reduced process. When the load is one the process is null recurrent and the QBD-CR takes a much longer time than for lower loads. This effect can be reduced by using the shift technique [18], resulting in times similar to those shown for loads up to 0.9. When comparing the two alternative duals, it is clear how the Bright dual outperforms the Ramaswami dual, being specially effective when the load is low, i.e., when the number of GI/M/1-type blocks is large. This effect is to be expected since for $\rho_2 < 1$ the GI/M/1-type MC is positive recurrent, and therefore the Ramaswami dual is transient, while the Bright dual is positive recurrent (see [31]).

In Figure 1 we include, for the full-size QBD, the computation times of CR (QBD-CR), the original U-based algorithm (QBD-U), and the modified version of the U-based algorithm (QBD-GT) proposed by Grassman and Tavakoli [8] to exploit the special structure of A_2. We also include the total time required to solve the reduced process using the Ramaswami dual (GM1-R) and the Bright dual (GM1-B). The scenario is the same as in the previous case, with the only exception that the buffer size in the first queue is $C = 50$. This means that the block size is $m = 408$ while the number of nonzero rows in A_2 does not change. This reduction is done because of the long computation times experienced with the U-based method, as can be observed in the figure. From these results the substantial gain obtained by the QBD-GT method compared to the original QBD-U is evident, as the latter requires about 5 times as much computation time. In spite of this gain, the QBD-GT method performs better than the QBD-CR only for small values of ρ_2. In contrast, the GI/M/1-type-based approach performs better than CR on the full-size QBD, except for low values of ρ_2. For the remaining part of the load range (except $\rho_2 = 1$) the QBD-CR takes up to 6 times as much time as the GI/M/1-type based approach. In this case the time to compute R is smaller using the Bright dual than the Ramaswami dual. The difference is significant for low loads, when the number of blocks is large (2800 for $\rho_2 = 0.1$), and it vanishes as the load increases. Therefore, the use of the Bright dual implies an important reduction in computation times in the range of the load that is more critical for the reduced process.

5.4. Wireless Relay node

Our last example is a model introduced in [14] to evaluate the packet-level performance in a wireless node implementing user relaying. In a wireless network, one of the source nodes can hear

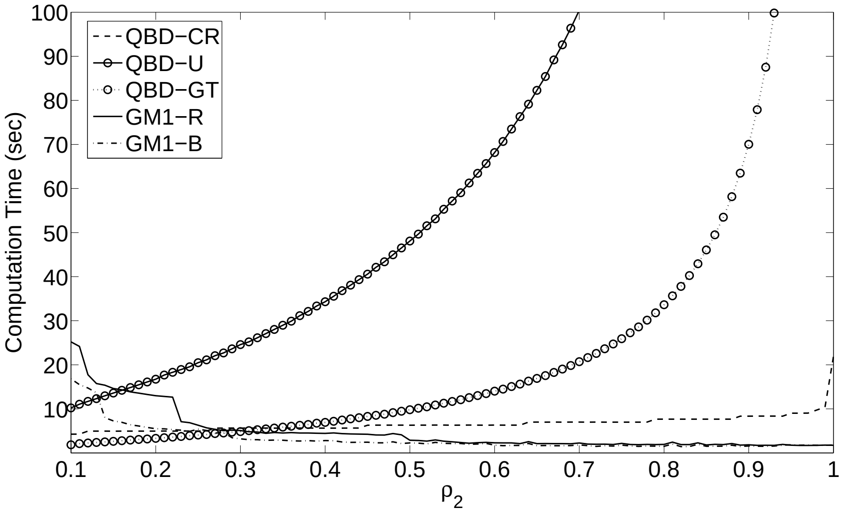

Figure 1: Computation times (sec) for the overflow queue with $C = 50$

what another source node is sending to the destination node, and it could therefore retransmit the information to the destination node. The node that retransmits the packets sent by another source node is called the relay node. The main purpose of enabling the source nodes to act as relay nodes is to provide multiple channels to transmit the same information, helping to alleviate the network's capacity loss caused by, for instance, channel fading [14]. To model a single relay node, the authors in [14] start by setting up an M/G/1-type MC where the level is the number of packets waiting for transmission in this node. The model is in discrete time. In a single slot, only one packet can be transmitted, and the number of packets arriving at the relay node can be between zero and $K < \infty$. However, taking advantage of the finite nature of K the authors propose a re-blocking of the MC's transition matrix to transform it into a QBD with block size Km, where m is the block size of the original M/G/1-type MC. The re-blocking operation has two main advantages: first, it is computationally less expensive to determine the stationary probability vector of a QBD MC than that of an M/G/1-type MC, once the matrix G of each of them has been computed, due to the matrix-geometric property; and second, in this particular case the matrix A_0 of the QBD MC has a few nonzero columns, a structure that can be exploited with the methods introduced in this paper. More specifically, the model in [14] considers three possible cases for the behavior of the relay node: in the first case the matrix A_0 has only one nonzero column, allowing the matrix G to be expressed directly in terms of the system parameters; in the other two cases the matrix A_0 has two nonzero columns, which prevents expressing G in closed form. Regarding the operation of the relay node, the three cases differ on how the node is allowed to participate in relaying the transmission of another source, and when it can transmit its own packets.

Table 5: Computation times (sec) for the wireless relay node - Case 2

K	100					200				
ρ	FI	QBD-CR	MG1	Ratio-F	Ratio-C	FI	QBD-CR	MG1	Ratio-F	Ratio-C
0.1	2.4	0.3	0.1	17.2	2.4	13.5	2.5	0.6	23.3	4.4
0.3	4.1	0.4	0.1	33.0	3.4	22.8	3.2	0.6	40.5	5.7
0.5	6.6	0.5	0.1	52.6	4.0	36.3	3.2	0.6	62.7	5.5
0.7	11.8	0.5	0.1	83.3	3.5	64.8	3.8	0.6	112.2	6.7
0.9	34.5	0.7	0.2	219.7	4.2	189.7	5.2	0.6	337.0	9.2

Table 6: Computation times (sec) for the wireless relay node - Case 3

K	100					200				
ρ	FI	QBD-CR	MG1	Ratio-F	Ratio-C	FI	QBD-CR	MG1	Ratio-F	Ratio-C
0.1	5.2	2.5	0.4	12.3	6.0	32.8	19.3	2.7	12.1	7.1
0.3	8.9	3.2	0.4	21.2	7.6	55.4	24.3	2.8	20.0	8.8
0.5	14.0	3.8	0.4	33.1	9.1	88.4	24.3	2.8	32.2	8.8
0.7	24.8	3.8	0.4	58.9	9.1	158.2	29.3	2.8	57.5	10.6
0.9	72.4	5.2	0.4	171.1	12.2	460.1	39.2	2.8	163.6	13.9

We consider the two cases where G cannot be found explicitly, to compare the performance of our method (MG1) not only with the CR algorithm on the full-size QBD (QBD-CR), but also with the approach proposed in [14], which also exploits the structure of A_0. This latter approach relies on a functional iteration to find the components of the G matrix, and therefore it has been labeled FI. We assume uncorrelated arrivals, as in [14], although the model can be easily generalized to allow for MAP arrivals while preserving the restricted-downward-transitions structure. The distribution of the number of packets arriving in a single slot is assumed to be uniformly distributed between one and K, and the probability of having zero arrivals is set according to the load ρ of the node. The results for the two cases are shown in Tables 5 and 6, which are labeled Case 2 and Case 3, respectively, as in [14]. Recall that the size of the QBD blocks is Km, where m is equal to 2 and 4 for cases 2 and 3, respectively. From these results we observe that in both cases there is a significant gain that can be obtained by exploiting the restricted-transitions structure with the methods introduced here. Moreover, the FI method is dramatically slower than both QBD-CR and MG1, even though this method takes advantage of the special structure of the block A_0. The main reason for this behavior, as with the QBD-GT method in the previous example, is that the gain obtained at each iteration of the algorithm is not enough to compensate for the large number of iterations required to find G. Additionally, the number of iterations, and therefore the computation time, is badly affected by the load of the node. For instance, for the third case and $K = 100$ the FI algorithm requires around 5 seconds to run if the load is 0.1, but more than one minute if the load equals 0.9. The computation time therefore increases by a factor of fourteen while, for the same scenario, the QBD-CR method requires about twice as much time when $\rho = 0.9$ compared to the case when $\rho = 0.1$. Furthermore, for the same instance the MG1 method is almost insensitive to the load and requires less than half a second to complete the computation of G.

Notice that during the numerical examples we have encountered three different behaviors of the MG1 method in relation to the load of the system under analysis: for the MAP/PH/1 and the priority queues the computation time increases with the load, for the overflow queue it decreases and for the relay node it is almost unaffected by the load. For the current example we see that the level of the QBD is a set of values for the number of packets waiting to be transmitted. In each level, the set $\mathcal{S}^+$ holds the phases where the number of waiting packets has one particular value (the largest in the level). When the load increases we expect the chain to make longer excursions toward higher levels, meaning the censored process has more blocks, but there is no particular reason for the chain to avoid or prefer the states with phase in $\mathcal{S}^+$. This is in contrast with the priority and the overflow queues, where a larger load implies that visits to the states with phase in $\mathcal{S}^+$ become less and more likely, respectively. Additionally, when comparing the performance of the various methods in the two cases, we observe that all of them require significantly more time in Case 3 than in Case 2. However, the difference between these two cases is not proportional for all the methods. In fact, for the FI method the average ratio between the times for Case 3 to those for Case 2 is slightly above two. For QBD-CR this figure is well above seven, while for MG1 it is around four. In spite of this relatively better behavior of the FI method, in absolute terms it requires substantially more time than MG1. Actually, for Case 3 the FI approach is between 12 and 170 times slower than our method. For the QBD-CR approach, this figure ranges between 6 and 14, confirming the benefits of exploiting the restricted-transitions structure with the approach introduced in this paper.

6. Conclusion

By means of the examples considered in the previous section, we have shown that the computation times to find the R or G matrix of the QBD MC can be substantially reduced with the approach proposed in this paper. As expected, for every case the gain increases with the ratio m/r, but it also depends on other factors related to the parameters of the system modeled. Even though for some cases the reduced-process approach may take longer than solving the full-size QBD, exploiting the structure of the matrices A_0 or A_2 may reduce the computation times substantially. To determine whether the reduced process can be useful for a particular system or not, attention must be paid to the expected sojourn times in $\mathcal{S}^-$ related to those in $\mathcal{S}^+$. If the sojourn times in $\mathcal{S}^-$ are too long compared to the sojourn times in $\mathcal{S}^+$, the reduced process will need many blocks to be described. This increases both the time required to compute the blocks and the time to find G_+ or R_+. However, to analyze the performance of a particular system it is usual to consider a broad range of conditions (load, variability, etc.), and it is likely that for a considerable part of this range the reduced process provides important reductions in computation times. An additional gain can be obtained for the GI/M/1-type case by using the Bright dual, which helps to reduce the computation times specially in those cases where the reduced process requires more time, i.e., when the number of blocks is large.

Appendix A. Dual processes

In this section we describe two dual relationships between discrete-time M/G/1- and GI/M/1-type processes. In both cases the dual process can be seen as the time-reverse of the original process with respect to an invariant measure [32, 33]. We consider the computation of an M/G/1-type MC as the dual of a GI/M/1-type MC, but the opposite relationship can be defined in a similar manner. The Ramaswami dual was introduced in [34] and its probabilistic interpretation given in [32]. Let the set of matrices $(A_i)_{i\geq 0}$ describe a GI/M/1-type MC, such that $A = \sum_{i=0}^{\infty} A_i$ is stochastic and irreducible. Then A is the transition matrix of a discrete-time MC with stationary probability vector α, i.e., $\alpha A = \alpha$ and $\alpha e = 1$. The Ramaswami dual is an M/G/1-type MC characterized by the set of matrices $(A_i^R)_{i\geq 0}$ given by $A_i^R = \Delta_R^{-1} A_i' \Delta_R$, where $\Delta_R = diag(\alpha)$. The G matrix of this process, denoted G_R, is related to the R matrix of the original process by $G_R = \Delta_R^{-1} R' \Delta_R$. Let $\rho(M)$ denote the spectral radius of a matrix M. Since the matrix G_R has the same eigenvalues as R, if the original GI/M/1-type MC is positive recurrent ($\rho(R) < 1$) the dual process is transient ($\rho(G_R) < 1$), and vice versa. The dual process will be null recurrent if and only if the original process is also null recurrent. In this case the dual process is the time-reverse process with respect to the invariant measure α.

We now turn to the Bright dual [33], which is defined as the time-reverse process with respect to a different invariant measure. If the GI/M/1-type MC is positive recurrent, the eigenvalue of maximum real part of R is $\eta = \rho(R) < 1$. It has been shown that the spectral radius of the matrix $\sum_{i=0}^{\infty} A_i \eta^i$ is equal to one [19]. Therefore there exists a positive vector w_η such that

$$w_\eta \left(\sum_{i=0}^{\infty} A_i \eta^i \right) = w_\eta.$$

The Bright dual is an M/G/1-type MC characterized by the matrices $(A_i^B)_{i\geq 0}$ defined as $A_i^B = \eta^{i-1} \Delta_B^{-1} \hat{A}_i' \Delta_B$, where $\Delta_B = diag(w_\eta)$. The matrix R of the original GI/M/1-type MC and the matrix G_B of the dual process are related by $G_B = \eta^{-1} \Delta_B^{-1} R' \Delta_B$. In this case the eigenvalues of the matrix G_B are the eigenvalues of R divided by η. Hence the spectral radius of G_B is equal to one and the dual process is positive recurrent [33]. When the process is positive recurrent, as in the examples shown in Section 5 for loads less than one, the Ramaswami dual will be transient while the Bright dual will be positive recurrent. As explained in detail in [31], the Bright dual can therefore reduce the computation times achieved by the Ramaswami dual considerably. This is confirmed numerically in Section 5, especially when the load of the overflow queue is small, which results in a large number of blocks for the GI/M/1-type MC and a small value of η. The computation time for the reduced process increases with the number of blocks, but the gain that can be realized by using the Bright dual is larger when η is smaller [31]. Therefore, the Bright dual becomes especially useful in this case as it compensates the larger computation times caused by the number of blocks.

Appendix B. Solving Sylvester matrix equations

In this section we describe how to solve the matrix equations (11) and (12) using the Hessenberg-Schur decomposition proposed in [21]. As noted before, these are Sylvester matrix equations of the form $AXB + X = E$. Consider Equation (11) and let $n = m - r$, then X and E are $n \times r$ matrices, while A and B are square matrices of size n and r, respectively. The first step to solve this linear system is to find orthogonal matrices U and V such that $U'AU = P$ and $V'BV = R$, where P is an upper-Hessenberg matrix, R is a quasi-upper triangular matrix and $'$ denotes the transpose operator. A matrix P is upper-Hessenberg if its entries $P_{ij} = 0$ for $i > j + 1$. A quasi-upper triangular matrix, also called real Schur form, is block-triangular with 1×1 (resp. 2×2) diagonal blocks that correspond to the real (resp. complex) eigenvalues [23]. While the Hessenberg decomposition to obtain U can be done using Householder transformations, the real Schur decomposition to compute V makes use of the QR algorithm, see [23, Chapter 7]. Let $F = U'EV$ and $Y = U'XV$, then the linear system becomes $PYR + Y = F$. Therefore, to find Y_k, the k-th column of the matrix Y, we need to solve the system

$$P \sum_{j=1}^{\max(k+1,r)} R_{jk} Y_j + Y_k = F_k,$$

for $1 \leq k \leq r$. However, the quasi-upper triangular form of R greatly simplifies this system. For $k < r$ there are two possible cases, either $R_{k+1,k} = 0$ or not. If $R_{k+1,k} = 0$, then Y_k is the solution to the $n \times n$ Hessenberg system

$$(PR_{k,k} + I)Y_k = F_k - \sum_{j=1}^{k-1} R_{jk} P Y_j, \tag{B.1}$$

which can be solved in $O(n^2)$ time. On the other hand, $R_{k+1,k} \neq 0$ implies $R_{k+2,k+1} = 0$, and hence we need to solve

$$\begin{bmatrix} PR_{k,k} + I & PR_{k+1,k} \\ PR_{k,k+1} & PR_{k+1,k+1} + I \end{bmatrix} \begin{bmatrix} Y_k \\ Y_{k+1} \end{bmatrix} = \begin{bmatrix} \hat{F}_k^{k-1} \\ \hat{F}_{k+1}^{k-1} \end{bmatrix}, \tag{B.2}$$

where $\hat{F}_k^l = F_k - \sum_{j=1}^{l} R_{jk} P Y_j$, for $1 \leq l \leq k-1$ and $1 \leq k \leq r$. This $2n \times 2n$ linear system is upper-triangular with two nonzero subdiagonals that can be solved in $O(n^2)$ time [21]. Notice, to determine Y_k it is necessary to know $Y_1, \ldots, Y_{k-1}$. Therefore, the algorithm starts by computing the first (or first two) column(s), and then works forward until the last column of Y has been computed. After finding Y, the matrix X can be computed as $X = UYV'$.

It is possible to apply this procedure to either the original or the transpose $B'X'A' + X' = E'$ system. In the first case A is transformed into Hessenberg form and B into real Schur form, while the opposite happens in the second case. The choice directly affects the computation times since for a matrix of size b the Schur decomposition can be done in $10b^3$ operations, while it takes $\frac{5}{3}b^3$ operations to compute the Hessenberg decomposition using Householder transformations [21, 23]. Therefore, to solve Equation (11) it is better to use the original system since the Hessenberg decomposition is applied on the $n \times n$ matrix A, which is larger than the $r \times r$ matrix B under the assumption that $r \ll m$. On the other hand, to solve Equation (12) it is preferable to first transpose the system since in that case B is an $n \times n$ matrix given by $B = A_0^{--}(I - A_1^{--})^{-1}$.

An additional issue to take into account when solving equations (11) and (12) is the actual computation of the matrices A, B and E. Take for example Equation (11), where $A = (I - A_1^{--})^{-1}A_2^{--}$, $B = G_+$ and $E = (I - A_1^{--})^{-1}(A_0^{-+} + A_1^{-+}G_+ + A_2^{-+}G_+^2)$. Although all the matrices involved are already computed it is still necessary to perform two matrix multiplications to determine A and E. In the examples shown in this paper the A_i^{-+} blocks are sparse or even zero. In some cases however these blocks can be dense and therefore these matrix multiplications may require considerably more time. A way to avoid this is to solve the slightly different equation

$$(I - A_1^{--})G_0 - A_2^{--}G_0G_+ = A_0^{-+} + A_1^{-+}G_+ + A_2^{-+}G_+^2,$$

which is a Sylvester matrix equation of the type $AXB + CX = E$. The procedure to solve this equation is very similar to the one shown above, but in this case the first step of the QZ algorithm [23] is applied to the pair (A, C). As a result A is reduced to Hessenberg form while C is transformed into upper-triangular form. This, together with a reduction of B to quasi-upper triangular form, allows the solution of this equation in a similar way as done in (B.1) and (B.2). A detailed explanation can be found in [22]. Since the matrices A, B, C and E are already computed, this algorithm may perform better when the blocks A_i^{-+} are dense. We have found instances of random QBDs with dense blocks where this last algorithm outperforms the one based on the equation $AXB + X = E$.

[1] V. Wallace, The solution of quasi birth and death processes arising from multiple access computer systems, Ph.D. thesis, Systems Engineering Laboratory, University of Michigan (1969).

[2] M. F. Neuts, Matrix-Geometric Solutions in Stochastic Models, The Johns Hopkins University Press, Baltimore, 1981.

[3] D. A. Bini, B. Meini, On the solution of a nonlinear matrix equation arising in queueing problems, SIAM Journal of Matrix Analysis and Applications 17 (1996) 906–926.

[4] G. Latouche, V. Ramaswami, A logarithmic reduction algorithm for quasi-birth-and-death processes, Journal of Applied Probability 30 (1993) 650–674.

[5] J. S. H. van Leeuwaarden, M. S. Squillante, E. M. M. Winands, Quasi-birth-and-death processes, lattice path counting and hypergeometric functions, Journal of Applied Probability 46 (2009) 507–520.

[6] B. Van Houdt, J. S. H. van Leeuwaarden, Triangular M/G/1-type and tree-like QBD Markov chains, under review.

[7] K. De Turck, S. De Vuyst, D. Fiems, S. Wittevrongel, H. Bruneel, Performance of the sleep-mode mechanism of the new IEEE 802.16m proposal for correlated downlink traffic, in: NET-COOP, 2009.

[8] W. K. Grassmann, J. Tavakoli, Solving QBD processes when levels can increase only in certain phases, manuscript in preparation, presented at the MAM6 conference, Beijing (China), June 2008.

[9] G. Latouche, Algorithms for infinite Markov chains with repeating columns, in: C. D. Meyer, R. J. Plemmons (Eds.), Linear Algebra, Markov chains and Queueing Models, Springer-Verlag, 1993, pp. 231–265.

[10] J. W. Cohen, The Single Server Queue, North-Holland, 1969.

[11] N. K. Jaiswal, Priority Queues, Academic Press, 1968.

[12] K. S. Meier-Hellstern, The analysis of a queue arising in overflow models, IEEE Transactions on Communications 37 (1989) 367–372.

[13] B. Van Houdt, C. Blondia, The waiting time distribution of a type k customer in a discrete-time MMAP[K]/PH[K]/c (c = 1, 2) queue using QBDs, Stochastic Models 20 (2004) 55–69.

[14] J. Cai, A. S. Alfa, P. Ren, X. Shen, J. W. Mark, Packet level performance analysis in wireless user-relaying networks, IEEE Transactions on Wireless Communications 7 (2008) 12.

[15] D. A. Bini, B. Meini, S. Steffé, B. Van Houdt, Structured Markov chains solver: software tools, in: SMCtools Workshop, ACM Press, Pisa, Italy, 2006.

[16] G. Latouche, V. Ramaswami, Introduction to Matrix Analytic Methods in Stochastic Modeling, ASA-SIAM Series on Statistics and Applied Probability, SIAM, Philadelphia, PA, 1999.

[17] G. Ciardo, E. Smirni, ETAQA: an efficient technique for the analysis of QBD-processes by aggregation, Performance Evaluation 36-37 (1999) 71–93.

[18] D. Bini, G. Latouche, B. Meini, Numerical Methods for Structured Markov Chains, Oxford University Press, 2005.

[19] M. F. Neuts, Structured stochastic matrices of M/G/1 type and their applications, Marcel Dekker Inc., 1989.

[20] V. Ramaswami, A stable recursion for the steady state vector in Markov chains of M/G/1 type, Stochastic Models 4 (1988) 183–188.

[21] G. H. Golub, S. Nash, C. Van Loan, A Hessenberg-Schur method for the problem AX+XB=C, IEEE Transactions on Automatic Control 24 (1979) 909–913.

[22] J. D. Gardiner, A. J. Laub, J. J. Amato, C. B. Moler, Solution of the Sylvester matrix equation AXBT + CXDT = E, ACM Transactions on Mathematical Software 18 (1992) 223–231.

[23] G. H. Golub, C. Van Loan, Matrix Computations, The Johns Hopkins University Press, 1996.

[24] W. Whitt, Approximating a point process by a renewal process, I: Two basic methods, Operations Research 30 (1982) 125–147.

[25] A. Heindl, Inverse characterization of hyperexponential MAP(2)s, in: Proc. 11th Int. Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA), 2004.

[26] J. E. Diamond, A. S. Alfa, On approximating higher order MAPs with MAPs of order two, Queueing Systems 34 (2000) 269–288.

[27] A. Cumani, On the canonical representation of homogeneous Markov processes modeling failure-time distributions, Microelectronics Reliability 22 (1982) 583–602.

[28] BEA Systems, IBM, Microsoft Corporation Inc, TIBCO Software Inc, Web services reliable messaging protocol (WS-ReliableMessaging) (2005).

[29] P. Reinecke, K. Wolter, Phase-type approximations for message transmission times in web services reliable messaging, in: SIPEW '08: Proceedings of the SPEC international workshop on Performance Evaluation, 2008.

[30] P. Reinecke, K. Wolter, Acyclic phase-type distribution models for WSRM, `http://www2.informatik.hu-berlin.de/~preineck/acphmodels/`.

[31] P. G. Taylor, B. Van Houdt, On the dual relationship between Markov chains of GI/M/1 and M/G/1 type, to appear in Advances in Applied Probability.

[32] S. Asmussen, V. Ramaswami, Probabilistic interpretations of some duality results for the matrix paradigms in queueing theory, Communications in Statistics Stochastic Models 6 (1990) 715–733.

[33] L. Bright, Matrix-analytic methods in applied probability, Ph.D. thesis, Department of Applied Mathematics, University of Adelaide (1996).

[34] V. Ramaswami, A duality theorem for the matrix paradigms in queueing theory, Communications in Statistics Stochastic Models 6 (1990) 151–161.