Analyzing M/G/1-type Markov chains with low-rank downward transitions

Juan F. Pérez and Benny Van Houdt
Performance Analysis of Telecommunication Systems Research Group
Department of Mathematics and Computer Science, University of Antwerp - IBBT
Middelheimlaan 1, B-2020 Antwerp, Belgium
Email:{juanfernando.perez, benny.vanhoudt}@ua.ac.be

***Abstract*—In this paper we analyze M/G/1-type Markov chains (MCs) where, in a downward transition, the phase variable faces a partial renewal. We refer to this characteristic as low-rank downward transitions. To analyze a chain of this type we define a new M/G/1-type MC, by adding a few artificial states to each level. In this new MC, any downward transition takes the phase to one of the additional states only. In previous works, it has been shown that this type of transitions induces a structure that can be exploited to speed up the computation of the matrix G. However, to find G it is necessary to solve a linear system, which has been solved efficiently for two special cases only. Here we also introduce a method to solve this system in general by means of the Schur decomposition. We illustrate numerically the important reduction in computation time obtained by exploiting the low-rank transitions.**

I. Introduction

A discrete-time M/G/1-type Markov chain (MC) is a two-dimensional process $\{(N_n, X_n), n \geq 0\}$, where N_n is called the *level* variable, and X_n the *phase* variable [1]. The level takes values on $\mathbb{N}$, and the phase does it on a finite set of size m_0 or m depending on whether the level is equal to or greater than 0. In addition, in a single transition the value of the level variable can increase without bounds, but it can decrease either by one or to zero. Also, the transition probabilities are assumed to be level-independent, i.e., the transition probability from a state (i, j) to a state (k, l) may depend on j, l, and the difference $i - k$, but not on the specific values of i and k. By ordering the state space lexicographically, the transition probability matrix P of an M/G/1-type MC can be written as

$$P = \begin{bmatrix} B_0 & B_1 & B_2 & B_3 & B_4 & \cdots \\ C_0 & A_1 & A_2 & A_3 & A_4 & \cdots \\ C_1 & A_0 & A_1 & A_2 & A_3 & \cdots \\ C_2 & 0 & A_0 & A_1 & A_2 & \cdots \\ C_3 & 0 & 0 & A_0 & A_1 & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where $\{A_i, i \geq 0\}$, $\{C_i, i \geq 0\}$, $\{B_i, i \geq 1\}$ and B_0 are non-negative matrices in $\mathbb{R}^{m\times m}$, $\mathbb{R}^{m\times m_0}$, $\mathbb{R}^{m_0\times m}$ and $\mathbb{R}^{m_0\times m_0}$, respectively, such that $\sum_{i=0}^{+\infty} B_i e = e$, $C_0 e + \sum_{i=1}^{+\infty} A_i e = e$, and $C_j e + \sum_{i=0}^{+\infty} A_i e = e$, for $j \geq 1$, where e is a column vector of ones of appropriate size. The stationary probability vector of this chain, i.e., the vector $\pi = [\pi_0, \pi_1, \pi_2, \dots]$ such that $\pi P = \pi$ and $\pi e = 1$, can be found by means of Ramaswami's formula [2], that relies on the matrix G, which is the minimal non-negative solution of

$$G = \sum_{i=0}^{\infty} A_i G^i. \qquad (1)$$

To compute G one can rely on functional iterations [1], Ramaswami's Reduction [3], or Cyclic Reduction (CR) [4]. However, when the block-size m becomes large, even the fastest algorithms may require long computation times. An alternative to overcome this problem is to consider the inner structure of the blocks $\{A_i, i \geq 0\}$. This approach has been taken in [5], where the blocks are assumed to be triangular, allowing a fast computation of G. Another interesting structure arises if the block A_0 is assumed to have a few nonzero columns only. This structure was originally proposed in [6] for Quasi-Birth-and-Death (QBD) MCs, a case that was further analyzed in [7]. There, the matrix G is found by applying a censoring argument and solving a Sylvester matrix equation [8]. Moreover, these results were generalized in [9] to the case of M/G/1-type MCs, where the structure of A_0 is exploited not only to compute G, but also in Ramaswami's formula to compute π. However, in the M/G/1-type case, to find G it is necessary to solve a general linear system, which in [9] is done efficiently for two special cases only.

Our contribution: In this paper we extend the work in [7], [9] by introducing and analyzing the more general type of *low-rank* transitions. An M/G/1-type MC is said to show *low-rank downward* transitions if, during a transition that decreases the value of the level, the phase variable faces a partial renewal. Under a partial renewal, the value of the phase variable after a downward transition can be determined without knowing the precise state, as a partial description suffices. Moreover, this partial description takes values of a finite set of size $r \ll m$. If this property holds, a new M/G/1-type MC can be defined by adding r artificial states in each level. In this chain any downward transition leads to one of the r artificial states, which induces a structure that can be exploited with the methods in [9]. This methodology will be described in detail in Section III. As mentioned above, in the M/G/1-type case, to find the matrix G one must apply a censoring argument, solve a nonlinear equation with matrices of smaller size, and solve a general linear system. However, in [9] this system is solved efficiently for two special cases only. In Section IV we present a method to solve this linear system efficiently in general. Although we focus on M/G/1-type MCs, the methods introduced here can be easily adapted to deal with other structured MCs, such as QBD and GI/M/1-type MCs.

In the next section, we motivate the study of M/G/1-type MCs with low-rank transitions by means of the discrete-time preemptive priority queue with batch arrivals. This example will be used in Section V to illustrate numerically the benefit of exploiting low-rank transitions.

II. The discrete-time preemptive priority queue

Here we consider a BMAP/PH/1 preemptive priority queue in discrete time with two priority classes, high and low. A low-priority priority customer is served only if there are no high-priority customers in the queue. Also, if a low-priority customer is being served, and a high-priority one arrives, the server interrupts its current service and starts serving the high-priority one. When all the high-priority customers in the queue have been attended, the low-priority one that was preempted can (re-)start its service. The service times of the high-priority customers follow a Phase-Type (PH) distribution [10] with parameters (m_s^1, α, T). Here α is a $1 \times m_s^1$ stochastic vector and T is an $m_s^1 \times m_s^1$ sub-stochastic matrix. Also, the service times of the low-priority customers follow a PH distribution characterized by (m_s^2, β, S). The inter-arrival times follow a batch Markovian arrival process (BMAP) characterized by the matrices $\{D_{ij}, 0 \leq i \leq L_1, 0 \leq i \leq L_2\}$, where the matrix D_{ij} holds the transition probabilities associated with the arrival of a batch carrying i high-priority and j low-priority customers. Here L_1 (resp. L_2) is the maximum batch size of high- (resp. low-) priority customers. To model this queue as an M/G/1-type MC one can choose the level to be the number of low-priority customers in the queue, as this ensures that in a single transition the level can decrease by at most one. On the other hand, the phase carries the number of high-priority customers, the state of the arrival process and the state of the current service. If a high priority customer is being served and there is at least one low-priority customer in the queue, the phase also holds the state in which the next-in-line low-priority customer will (re-)start its service whenever it gets access to the server. For the phase space to be finite, the high-priority buffer is assumed to be of size $C < \infty$. This poses no actual limitation to the model since C can be chosen large enough to cause very few high-priority losses. Moreover, high-priority queues are typically fairly short compared to low-priority queues. Hence, there is little difference between a, sufficiently large, finite or an infinite high-priority buffer.

As the exact form of the matrices $\{A_i, i \leq 1\}$, $\{B_i, i \leq 0\}$ and $\{C_i, i \leq 0\}$ is irrelevant for our discussion we limit ourselves to the $m \times m$ matrix A_0:

$$A_0 = \begin{bmatrix} D_{00} \otimes s\beta & D_{10} \otimes s\beta \otimes \alpha & \cdots & D_{L_1 0} \otimes s\beta \otimes \alpha & 0 \cdots 0 \\ 0 & 0 & \cdots & 0 & 0 \cdots 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \ddots \vdots \\ 0 & 0 & \cdots & 0 & 0 \cdots 0 \end{bmatrix},$$

where $s = e - Se$, and $m = m_a m_s^2(1 + C m_s^1)$. The phase space is ordered lexicographically: the first component carries the number of high-priority customers; the second keeps track of the state of the arrival process; the third component holds the state of the current low-priority customer in service, or the state in which the next-in-line customer of this type will re-start its service; and the last component holds the state of the current high-priority service, if any. Since the block A_0 holds the transition probabilities associated with a decrease in the number of low-priority customers, and this can only occur if there are not high-priority customers in the queue, only the first block-row of this matrix is different from zero.

Also, in a single transition at most L_1 high-priority customers may arrive, and therefore the block A_0 has $m_a m_s^2(1 + L_1 m_s^1)$ nonzero columns. Depending on the values of C and L_1, the number of nonzero columns could be small enough, compared to m, for the methods in [9] to provide a significant gain in computation time. However, one can go one step further by noticing that in a downward transition the process only needs to remember the state of the arrival process, since the service processes face a renewal, i.e., the new state of these processes is not influenced by their previous state. In the next section we will show how this M/G/1-type MC can be transformed into a new one where the downward transitions trigger the phase to a set of size m_a only. As a result, in the new chain the number of nonzero columns in A_0 is $m_s^2(1 + C m_s^1)$ times smaller than the original block size, and a significant gain in computation times can be expected by using the methods in [9], even for small values of C.

III. Low-rank transitions

Let us start with an M/G/1-type MC $\{(N_n, X_n), n \geq 0\}$, where the phase space, when $N_n > 0$, is of size m. Now, assume that the variable X_n can be decomposed into two components: $X_n = (Y_n, Z_n)$, for $n \geq 0$. The variable Y_n takes values on the set $\{1, \ldots, r\}$, while Z_n does so on $\{1, \ldots, h\}$, with $rh = m$. Therefore, the $((i,j),(k,l))$-th entry of A_0 can be written as

$$\begin{aligned} P[N_{n+1} = a-1, Y_{n+1} = k, Z_{n+1} = l | N_n = a, Y_n = i, Z_n = j] \\ = P[Y_{n+1} = k, Z_{n+1} = l | N_{n+1} = a-1, N_n = a, Y_n = i, Z_n = j] \\ \times P[N_{n+1} = a-1 | N_n = a, Y_n = i, Z_n = j], \end{aligned}$$

for $a > 1$. The first term on the right-hand side holds the probability that the tuple (Y_{n+1}, Z_{n+1}) takes the values (k, l), given that there is a downward transition at time n and $(Y_n, Z_n) = (i, j)$. We now introduce the low-rank (partial renewal) assumption: the value of (Y_{n+1}, Z_{n+1}) may depend on Y_n, but *not* on the value of Z_n. Still, the probability of having a downward transition at time n may depend on the value of both Y_n and Z_n. Under this assumption the $((i,j),(k,l))$-th entry of A_0 becomes

$$\begin{aligned} P[N_{n+1} = a-1, Y_{n+1} = k, Z_{n+1} = l | N_n = a, Y_n = i, Z_n = j] \\ = P[Y_{n+1} = k, Z_{n+1} = l | N_{n+1} = a-1, N_n = a, Y_n = i] \\ \times P[N_{n+1} = a-1 | N_n = a, Y_n = i, Z_n = j], \end{aligned}$$

which are independent of $a \geq 2$. Therefore, we have decomposed the entries of A_0 in two parts: one regarding the probability that a downward transition occurs at time n, given that $(Y_n, Z_n) = (i, j)$; and the other holding the probability that $(Y_{n+1}, Z_{n+1}) = (k, l)$, given that a downward transition occurs at time n and $Y_n = i$.

We can assemble these probabilities in matrix form by defining the $m \times r$ sub-stochastic matrix V and the $r \times m$ stochastic matrix W. Let the rows of the matrix V be indexed lexicographically as the phase space, and the $((i, j), i)$-th entry of this matrix be equal to the (i, j)-th entry of the column vector $A_0 e$. In other words, the $((i, j), i)$-th entry of V holds the probability that a downward transition occurs at time n, given that $(Y_n, Z_n) = (i, j)$. On the other hand, let the $(i, (k, l))$-th entry of the matrix W be equal to the probability that $(Y_{n+1}, Z_{n+1}) = (k, l)$, given that a downward transition occurs at time n and $Y_n = i$. Therefore, we have $A_0 = VW$.

With this decomposition of A_0 we define a new M/G/1-type MC with $(m + r) \times (m + r)$ blocks $\{\hat{A}_i, i \geq 0\}$ given by

$$\hat{A}_0 = \begin{bmatrix} 0 & 0 \\ V & 0 \end{bmatrix}, \quad \hat{A}_1 = \begin{bmatrix} 0 & W \\ 0 & A_1 \end{bmatrix}, \quad \hat{A}_j = \begin{bmatrix} 0 & 0 \\ 0 & A_j \end{bmatrix}, \ j \geq 2.$$

To complete the description of this chain, let its boundary blocks be given by

$$\hat{B}_0 = B_0, \quad \hat{C}_0 = \begin{bmatrix} 0_{r \times m_0} \\ C_0 \end{bmatrix}, \quad \hat{B}_i = \begin{bmatrix} 0_{m_0 \times r} & B_i \end{bmatrix}, \ i \geq 1,$$

where $0_{m \times n}$ is an $m \times n$ zero matrix. This new MC has two main characteristics that make it attractive. First, its block $\hat{A}_0$ has r nonzero columns only, and therefore the methods in [9] can be applied to speed up the computation of its stationary vector $\hat{\pi} = [\hat{\pi}_0, \hat{\pi}_1, \hat{\pi}_2, \ldots]$. Second, if this MC is observed in the states $\{\{(0, j), 1 \leq j \leq m_0\} \cup \{(i, j), i \geq 1, r + 1 \leq j \leq r + m\}\}$ only, we obtain the original MC. Therefore, we can compute the stationary vector π of the original chain from the vector $\hat{\pi}$. To do this, let the vector $\hat{\pi}_i$ be partitioned as $\hat{\pi}_i = [\hat{\pi}_i^+, \hat{\pi}_i^-]$, for $i \geq 1$, where $\hat{\pi}_i^+$ corresponds to the first (additional) r states in level i. Let K be the probability that the new MC is in any of the additional states in steady state, i.e., $K = \sum_{i \geq 1} \hat{\pi}_i^+ e$. Then, we can find the stationary probability vector of the original chain as $\pi_0 = \hat{\pi}_0/(1 - K)$, and $\pi_i = \hat{\pi}_i^-/(1 - K)$ for $i \geq 1$.

In conclusion, by defining this new M/G/1-type MC of slightly larger block size, we are able to speed up the computation of the stationary vector of the original chain by exploiting the small number of nonzero columns in $\hat{A}_0$. To to this, we follow [7], [9], partitioning the blocks $\{\hat{A}_i, i \geq 0\}$ as

$$\hat{A}_i = \begin{bmatrix} A_i^{++} & 0 \\ A_i^{-+} & 0 \end{bmatrix}, \quad \hat{A}_i = \begin{bmatrix} A_i^{++} & A_i^{+-} \\ A_i^{-+} & A_i^{--} \end{bmatrix}, \ 1 \leq i \leq N,$$

where N is the smallest integer such that $\hat{A}_i = 0$ for $i > N$. Here A_i^{++} and A_i^{--} are square matrices of size r and m, respectively. Due to the structure of $\hat{A}_0$, we know [9] that the matrix $\hat{G}$ has a similar structure, namely

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

where G_+ (resp. G_0) is an $r \times r$ (resp. $m \times r$) matrix. These two sub-matrices can be computed separately, as shown in [9]. First, G_+ is found by censoring the chain, observing it only when its phase variable is in one of the r additional phases. This defines a new M/G/1-type MC with blocks of size r, the G matrix of which is equal to G_+. After finding G_+, we obtain G_0 by solving a linear system, which results by rewriting Equation (1) in block form and considering the structure of $\hat{A}_0$ and $\hat{G}$. Extracting the lower-left block of that equation we find

$$-\sum_{i=1}^{N} A_i^{--} G_0 G_+^{i-1} = \sum_{i=0}^{N} A_i^{-+} G_+^i. \tag{2}$$

As G_0 is the only unknown term, this is a general linear system of the form $\sum_{i=1}^{N} A_i X B_i = C$. This system has mr unknowns and equations, therefore its solution by general procedures has a time complexity of $O(m^3 r^3)$. Assuming a large m, this system can be solved directly if r is very small. Another possibility is to use an iterative approach as those proposed in [11], although these are not guaranteed to converge to the actual solution. Also, in [9] it was shown how to solve this equation in $O(m^3)$ time for two special cases. In the next section we present a method to solve this system in general in $O(m^3 r)$ time.

IV. Solving the general linear system

To solve the system (2) we start by observing that it has a special characteristic: the matrices that post-multiply the unknown matrix G_0 are all powers of the same matrix G_+. The key to exploit this fact is to apply a real Schur decomposition [12] to G_+, i.e., to find an orthogonal matrix $U \in \mathbb{R}^{r \times r}$ such that $U' G_+ U = T$, where $'$ denotes the transpose operator. Recall that a matrix U is called orthogonal if $U'U = UU' = I$. The matrix $T \in \mathbb{R}^{r \times r}$ is upper quasi-triangular, meaning it is block upper triangular and the diagonal blocks are of size one or two [12]. We now post-multiply (2) by U to obtain

$$-\sum_{i=1}^{N} A_i^{--} G_0 U U' G_+^{i-1} U = \sum_{i=0}^{N} A_i^{-+} G_+^i U,$$

which, since $U' G_+^j U = T^j$ for any nonnegative integer j, can be rewritten as

$$-\sum_{i=1}^{N} A_i^{--} G_0 U T^{i-1} = \sum_{i=0}^{N} A_i^{-+} G_+^i U.$$

Now let $Y = \sum_{i=0}^{N} A_i^{-+} G_+^i U$, which is a known matrix, and let $X = G_0 U$, to obtain

$$-\sum_{i=1}^{N} A_i^{--} X T^{i-1} = Y. \tag{3}$$

This system can be equivalently written column-wise as

$$-\sum_{i=1}^{N} A_i^{--} \sum_{j=1}^{r} [T^{i-1}]_{jk} X_j = Y_k, \tag{4}$$

for $k = 1, \dots, r$, where M_k and $[M]_{i,j}$ are the k-th column and the (i,j)-th entry of a matrix M, respectively.

Notice, in Equation (3) the matrices that post-multiply X are all upper quasi-triangular matrices and have the same block structure, as they all are powers of T. Therefore it is possible to iteratively compute the columns X_k, starting with X_1. Let us assume that we have already found $\{X_1, \dots, X_{k-1}\}$ and we want to compute X_k, for some $1 \leq k \leq r$. Given the upper quasi-triangular nature of T, there are two possibilities. The first is that the entry $[T]_{k+1,k}$ is zero, meaning that Equation (4) can be rewritten as

$$-\sum_{i=1}^{N} A_i^{--}[T^{i-1}]_{kk}X_k = Y_k + \sum_{i=1}^{N} A_i^{--} \sum_{j=1}^{k-1} [T^{i-1}]_{jk}X_j.$$

Therefore we can obtain the column X_k by solving a linear system of size m, which requires $O(m^3)$ time. The second case is when $[T]_{k+1,k} \neq 0$, which, due to the upper quasi-triangular structure (the diagonal blocks are at most of size two) implies that $[T]_{k+2,k+1} = 0$. Therefore we can find the columns X_k and X_{k+1} simultaneously by solving the system

$$-\begin{bmatrix} \sum_{i=1}^{N} A_i^{--}[T^{i-1}]_{kk} & \sum_{i=1}^{N} A_i^{--}[T^{i-1}]_{k+1,k} \\ \sum_{i=1}^{N} A_i^{--}[T^{i-1}]_{k,k+1} & \sum_{i=1}^{N} A_i^{--}[T^{i-1}]_{k+1,k+1} \end{bmatrix} \begin{bmatrix} X_k \\ X_{k+1} \end{bmatrix} = \begin{bmatrix} \hat{Y}_k^{k-1} \\ \hat{Y}_{k+1}^{k-1} \end{bmatrix},$$

where $\hat{Y}_k^l = Y_k + \sum_{i=1}^{N} A_i^{--} \sum_{j=1}^{l} [T^{i-1}]_{jk}X_j$, for $1 \leq l \leq k-1$ and $1 \leq k \leq r$. This is a linear system with $2m$ unknowns that requires $O(m^3)$ time to be solved. As a result we can start by finding the first (two) column(s) of X and iteratively compute the others. After we have computed X, G_0 is obtained from $G_0 = XU'$. Since the Schur decomposition of G_+ requires $O(r^2)$ time and $r \ll m$, the computation of G_0 has a time complexity of $O(m^3 r)$.

V. Numerical results

In this section we make use of the discrete-time preemptive priority queue with batch arrivals described in Section II, to illustrate the computational gains obtained by exploiting the low-rank transitions. The load ρ for this queue is given by $\rho = \lambda_1/\mu_1 + \lambda_2/\mu_2$, where λ_1 (resp. λ_2) is the arrival rate and μ_1 (resp. μ_2) is the service rate for the high- (resp. low) priority customers. Table I shows the computation times for various load values, with a high-priority buffer of size 10 and 20. The batch size is uniformly distributed between 1 and L for both customer types, L being equal to 3 or 5. The arrival process for each customer type is a BMAP process with two states: an ON state where batch arrivals occur with geometric inter-arrival times; and an OFF state where no arrivals occur. The sojourn time in these states is geometrically distributed, and the ratio γ between the mean sojourn time in the OFF and the ON states is a measure of the arrival process' burstiness. We set $\gamma = 5$, i.e., a sojourn in the OFF state lasts on average 5 times more than a sojourn in an ON state. The arrival and service processes representations have sizes $m_a = 4$ and $m_s^1 = m_s^2 = 2$. Therefore, for $C = 10$ and $L = 3$, the block size of the original chain is 168, the number of nonzero columns in A_0 is 56, and the number of artificial states is 4. For $C = 20$ and $L = 5$, these figures are 328, 88 and 4. The table shows the total time to compute π applying: CR and a functional iteration (FI), based on the matrix U [10], directly on the chain with block-size m; the methods in [9] exploiting the fact that the block A_0 has $m_a m_s^2(1 + L_1 m_s^1)$ nonzero columns (RT$_\text{o}$); and the methods introduced here combined with those in [9], defining a new chain where the block $\hat{A}_0$ has m_a nonzero columns (RT$_\text{n}$). In the two latter cases the CR algorithm is used to find the matrix G_+. Under this setup, FI requires less computation time than CR, but exploiting the low-rank transitions we obtain a significant gain, comparing with any of the other methods. Moreover, when m is large the CR algorithm on the original chain fails to run due to lack of memory (-), while the method based on the low-rank transition is able to compute π in a few seconds.

Table I
Computation times (sec) to find π

ρ	$C = 10$ - $L = 3$				$C = 20$ - $L = 5$			
	FI	CR	RT$_\text{o}$	RT$_\text{n}$	FI	CR	RT$_\text{o}$	RT$_\text{n}$
0.1	0.9	31.7	3.1	0.2	7.4	519.8	17.5	1.6
0.3	1.5	66.0	4.5	0.3	13.0	-	30.0	2.3
0.5	2.6	80.0	7.3	0.4	22.4	-	61.5	3.4
0.7	4.9	99.6	11.1	0.6	42.2	-	97.2	5.3
0.9	12.1	123.0	19.6	1.0	126.8	-	169.3	12.1

References

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

[2] V. Ramaswami, "A stable recursion for the steady state vector in Markov chains of M/G/1 type," *Stochastic Models*, vol. 4, pp. 183–188, 1988.

[3] ——, *Advances in Matrix Analytic Methods for Stochastic Models.* Notable Publications Inc., 1998, ch. The generality of quasi birth-and-death processes, pp. 93–114.

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

[5] B. Van Houdt and J. S. H. van Leeuwaarden, "Triangular M/G/1-type and tree-like QBD Markov chains," to appear in INFORMS Journal on Computing.

[6] W. K. Grassmann and 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.

[7] J. F. Pérez and B. Van Houdt, "Exploiting restricted transitions in quasi-birth-and-death processes," in *Proceedings of the Sixth International Conference on the Quantitative Evaluation of Systems (QEST)*, 2009.

[8] G. H. Golub, S. Nash, and C. Van Loan, "A Hessenberg-Schur method for the problem AX+XB=C," *IEEE Transactions on Automatic Control*, vol. 24, pp. 909–913, 1979.

[9] J. F. Pérez and B. Van Houdt, "The M/G/1-type Markov chain with restricted transitions and its applications to queues with batch arrivals," under review.

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

[11] D. Bini, G. Latouche, and B. Meini, "Solving nonlinear matrix equations arising in tree-like stochastic processes," *Linear Algebra and its Applications*, vol. 366, pp. 39–64, 2003.

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