# Etaqa-MG1: An efficient technique for the analysis of a class of M/G/1-type processes by aggregation*

Gianfranco Ciardo    Weizhen Mao    Alma Riska    Evgenia Smirni

Department of Computer Science

College of William and Mary

Williamsburg, VA 23187-8795

{ciardo,wm,riska,esmirni}@cs.wm.edu

December 18, 2001

## Abstract

We extend the Etaqa approach, initially proposed for the efficient numerical solution of a class of quasi birth-death processes, to a more complex class of M/G/1-type Markov processes, where arbitrary forward transitions are allowed, but backward transitions must be to a single state to the previous level. The new technique can be used for the exact solution of this class of M/G/1-type models by computing the solution of a finite linear system. We demonstrate the utility of our method by describing the exact computation of an extensive set of Markov reward functions, which include the expected queue length or its higher moments. We also provide an algorithm that finds an appropriate state reordering satisfying our applicability conditions, if one such order exists. We illustrate the method, discuss its complexity and numerical stability, and present comparisons with other traditional techniques.
**Keywords:** M/G/1-type Markov chain; matrix-analytic solution; bulk arrival.

## 1  Introduction

We consider Markov chains on an infinite state space having an M/G/1-type structure. Such processes are often the modeling tool of choice for modern computer and communication systems [13]. As a consequence, considerable effort has been placed into the development of exact analysis techniques for them. In the continuous-time case, the infinitesimal generator of such processes is upper

---

1

block Hessenberg with a repetitive structure, and matrix-analytic methods have been proposed for their solution [15]. The key in the matrix-analytic solution is the computation of an auxiliary matrix, $\mathbf{G}$. Similarly, for Markov chains of the GI/M/1-type, which have a lower block Hessenberg form, matrix-geometric solutions have been proposed [14]. Again, the key in the matrix-geometric solution is the computation of an auxiliary matrix, $\mathbf{R}$. The traditional solution algorithms compute the stationary probability vector with a recursive function based on $\mathbf{G}$ (for the case of M/G/1-type processes) or $\mathbf{R}$ (for the case of GI/M/1-type processes), and iterative procedures are used for determining $\mathbf{G}$ or $\mathbf{R}$. Various algorithms for the computation of $\mathbf{G}$ or $\mathbf{R}$ have been proposed (e.g., the work of Latouche [8] for the efficient computation of $\mathbf{R}$ and of Meini [12] for the efficient computation of $\mathbf{G}$). See also [9] for a discussion of the fundamental aspects of matrix-analytic methods.

The primary distinction between our research and the above works is that we restrict our attention to a family of M/G/1-type processes with a specific form, for which "returns" from a higher level of states to the immediately lower level are always directed toward a single state (for such a subclass, the computation of the matrix $\mathbf{G}$ is trivial, but the remaining solution steps using traditional methods are still expensive). We instead recast the problem into that of solving a finite linear system in $m + 2n$ unknowns, where $m$ is the number of states in the boundary portion of the process and $n$ is the number of states in each of the repetitive "levels" of the state space, and are able to obtain exact results.

Since the method that we propose applies only to a restrictive class of M/G/1-type processes, we further investigate an algorithm to repartition the state space such that the condition for backward transitions to a single state hold. This algorithm is guaranteed to find such a repartitioning, if one exists; hence our technique does not apply to all M/G/1-type processes. However, we argue that our method applies to an important class of processes. For example, any queueing system with a coxian service process and bulk poisson arrival process results in an M/G/1-type chain with backward transitions directed to a single state, and it is a known fact that the class of coxian distributions is dense [4], thus can approximate any distribution, given enough phases. Also, [3] presents a survey of applications from the literature whose Markov chains exhibit the required structure required.

Our approach is an extension of the ETAQA method we introduced for the efficient solution of quasi-birth-death processes with matrix-geometric form [3]. The proposed methodology uses basic results for Markov chains. Assuming that the state space $\mathcal{S}$ is partitioned into the sets $\mathcal{S}^{(j)}$, for $j \geq 0$, where $\mathcal{S}^{(0)} = \{s_1^{(0)}, \ldots, s_m^{(0)}\}$ contains $m$ boundary states and, for $j \geq 1$, $\mathcal{S}^{(j)}$ contains the same number $n$ of states, $\mathcal{S}^{(j)} = \{s_1^{(j)}, \ldots, s_n^{(j)}\}$, we exploit the structure of the repetitive portion of the chain and instead of evaluating the probability distribution of *all* states in the chain, we calculate the *aggregate* probability distribution of $n$ equivalence classes $\mathcal{T}_i = \{s_i^{(j)} : j \geq 2\}$, for $1 \leq i \leq n$ (corresponding to a specific partition of the repetitive portion of the chain, see Fig. 1), as well as the stationary probability of each state in $\mathcal{S}^{(0)} \cup \mathcal{S}^{(1)}$.

The new ETAQA-MG1 approach is both exact and efficient. Indeed, our method does not explicitly
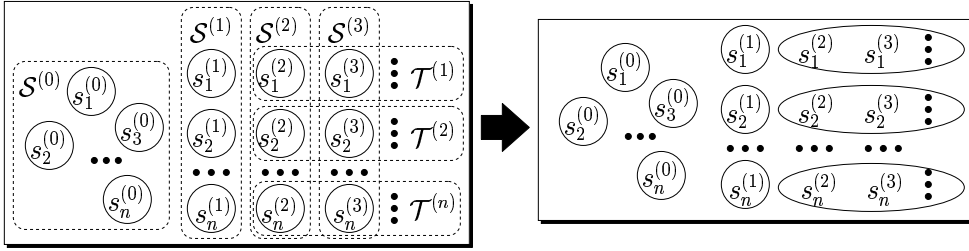
Figure 1: Aggregation of an infinite $\mathcal{S}$ into a finite number of states.

compute the *entire* stationary probability vector but only the part of it that is related to the boundary states, and the aggregate probability vector for the repetitive states. The traditional matrix-analytic method uses a recursive formula for the computation of the stationary probability vector. If this formula entails subtractions, then there is the possibility of numerical instability [15, 17]. The preferred method for the recursive computation of the stationary probability vector uses Ramaswami's recursive formula [17], which entails only additions. Thus, we compare experimentally the computation and storage complexity and the stability of our methodology with that of the matrix-analytic method based on Ramaswami's formula.

Our paper is organized as follows. Section 2 contains terminology and related work. In Section 3, we present the basic theorem that extends ETAQA to M/G/1-type processes. We demonstrate how the methodology can be used for the computation of Markov reward functions in Section 4. We continue by showing the applicability of ETAQA-MG1 to bounded bulk arrivals in Section 5. In Section 6, we compare the computation and storage complexity of ETAQA-MG1 with the matrix-analytic method. We briefly refer in Section 7 to an algorithm that can repartition an infinitesimal generator into a form such that ETAQA-MG1 can be applied and elaborate on this algorithm in Appendix A. Section 8 presents an applications from the computer systems area that can be analyzed using the ETAQA-MG1 approach, and numerical results. Section 9 discusses the numerical stability of our approach. Finally, Section 10 summarizes our contributions.

## 2   Background

We briefly review the terminology used to describe the class of processes we consider. We restrict ourselves to the case of continuous-time Markov chains (hence we refer to the infinitesimal generator matrix $\mathbf{Q}$), but the theory can just as well be applied to the discrete case.

Neuts [14] defines various classes of infinite-state Markov chains with a repetitive structure. In all cases, the state space is partitioned into the boundary states $\mathcal{S}^{(0)} = \{s_1^{(0)}, \ldots, s_m^{(0)}\}$ and the sets of states $\mathcal{S}^{(j)} = \{s_1^{(j)}, \ldots, s_n^{(j)}\}$, for $j \geq 1$. For the class of M/G/1-type Markov chains, the

3

infinitesimal generator $\mathbf{Q}$ is block-partitioned as:

$$
\mathbf{Q} = \begin{bmatrix}
\widehat{\mathbf{L}}^{(0)} & \widehat{\mathbf{F}}^{(1)} & \widehat{\mathbf{F}}^{(2)} & \widehat{\mathbf{F}}^{(3)} & \widehat{\mathbf{F}}^{(4)} & \widehat{\mathbf{F}}^{(5)} & \cdots \\
\widehat{\mathbf{B}} & \mathbf{L} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \mathbf{F}^{(3)} & \mathbf{F}^{(4)} & \cdots \\
\mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \mathbf{F}^{(3)} & \cdots \\
\mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \cdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix}, \tag{1}
$$

(we use the letters "L", "F", and "B" according to whether they describe "local", 'forward", and "backward" transition rates, respectively, and a "$\frown$" for matrices related to $\mathcal{S}^{(0)}$).

For the solution of M/G/1-type processes, several recursive algorithms exist [5, 12, 15]. Here, we outline an algorithm that provides a stable calculation for the values of $\boldsymbol{\pi}^{(j)}$, the stationary probability vector for states in $\mathcal{S}^{(j)}$. Using Ramaswami's recursive formula [17] we have:

$$
\forall j \geq 1, \quad \boldsymbol{\pi}^{(j)} = - \left( \boldsymbol{\pi}^{(0)} \widehat{\mathbf{S}}^{(j)} + \sum_{l=1}^{j-1} \boldsymbol{\pi}^{(l)} \mathbf{S}^{(j-l)} \right) \mathbf{S}^{(0)^{-1}} \tag{2}
$$

where $\widehat{\mathbf{S}}^{(j)}$ and $\mathbf{S}^{(j)}$ are defined as follows:

$$
\widehat{\mathbf{S}}^{(j)} = \sum_{l=j}^{\infty} \widehat{\mathbf{F}}^{(l)} \mathbf{G}^{l-j}, \;\; j \geq 1, \qquad \mathbf{S}^{(j)} = \sum_{l=j}^{\infty} \mathbf{F}^{(l)} \mathbf{G}^{l-j}, \;\; j \geq 0 \;\; (\text{letting } \mathbf{F}^{(0)} = \mathbf{L}),
$$

where $\mathbf{G}$ is the solution of the matrix equation:

$$
\mathbf{B} + \mathbf{L}\mathbf{G} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)} \mathbf{G}^{j+1} = \mathbf{0}. \tag{3}
$$

Several iterative algorithms for the computation of $\mathbf{G}$ exist [5, 15], the most efficient being cyclic-reduction [12]. See [18] for cases where $\mathbf{G}$ can be explicitly defined and does not require any calculation.

Given the above expression for $\boldsymbol{\pi}^{(j)}$ and the normalization condition, a unique vector $\boldsymbol{\pi}^{(0)}$ can be obtained by solving the following system of $m$ equations

$$
\boldsymbol{\pi}^{(0)} \left[ \left( \widehat{\mathbf{L}}^{(0)} - \widehat{\mathbf{S}}^{(1)} \mathbf{S}^{(0)^{-1}} \widehat{\mathbf{B}} \right)^{\diamond} \;\middle|\; \mathbf{1}^T - \left( \sum_{j=1}^{\infty} \widehat{\mathbf{S}}^{(j)} \right) \left( \sum_{j=0}^{\infty} \mathbf{S}^{(j)} \right)^{-1} \mathbf{1}^T \right] = [\mathbf{0} \mid 1], \tag{4}
$$

where the symbol "$\diamond$" indicates that we discard one (any) column of the corresponding matrix, since we added a column representing the normalization condition. Once $\boldsymbol{\pi}^{(0)}$ is known, we can then iteratively compute $\boldsymbol{\pi}^{(j)}$ for $j = 1, 2, \ldots$, stopping when the accumulated probability mass is close to one.

# 3    Extending Etaqa to M/G/1-type processes

We consider the following structure in the infinitesimal generator matrix $\mathbf{Q}$:

$$\mathbf{Q} = \begin{bmatrix} \widehat{\mathbf{L}}^{(0)} & \widehat{\mathbf{F}}^{(1)} & \widehat{\mathbf{F}}^{(2)} & \widehat{\mathbf{F}}^{(3)} & \widehat{\mathbf{F}}^{(4)} & \cdots \\ \widehat{\mathbf{B}} & \mathbf{L}^{(1)} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \mathbf{F}^{(3)} & \cdots \\ \mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{5}$$

which differs from that of Eq. 1 only in block $\mathbf{L}^{(1)}$, at the intersection of the rows and columns for $\mathcal{S}^{(1)}$, which is not restricted to be equal to $\mathbf{L}$. This allows us to have (a) local transitions from $\mathcal{S}^{(1)}$ to $\mathcal{S}^{(1)}$ with a pattern other than those from $\mathcal{S}^{(j)}$ to $\mathcal{S}^{(j)}$, $j \geq 2$, and (b) a total rate from each state $s_i^{(1)}$ to the states in $\mathcal{S}^{(0)}$ other than the total rate from state $s_i^{(j)}$ to the states in $\mathcal{S}^{(j-1)}$, i.e., the row sums of $\widehat{\mathbf{B}}$ and $\mathbf{B}$ can be different. For $\mathbf{Q}$ to be an infinitesimal generator, the infinite sets of matrices $\{\widehat{\mathbf{F}}^{(j)} : j \geq 1\}$ and $\{\mathbf{F}^j : j \geq 1\}$ must be summable. If we also partition the stationary probability vector satisfying $\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}$ as $\boldsymbol{\pi} = \begin{bmatrix} \boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(2)}, \ldots \end{bmatrix}$ with $\boldsymbol{\pi}^{(0)} \in \mathbb{R}^m$ and $\boldsymbol{\pi}^{(j)} \in \mathbb{R}^n$ for $j \geq 1$, we can then write $\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}$ as:

$$\begin{cases} \boldsymbol{\pi}^{(0)} \ \widehat{\mathbf{L}}^{(0)} \ + \ \boldsymbol{\pi}^{(1)} \ \widehat{\mathbf{B}} & = \mathbf{0} \\ \boldsymbol{\pi}^{(0)} \ \widehat{\mathbf{F}}^{(1)} \ + \ \boldsymbol{\pi}^{(1)} \ \mathbf{L}^{(1)} \ + \ \boldsymbol{\pi}^{(2)} \ \mathbf{B} & = \mathbf{0} \\ \boldsymbol{\pi}^{(0)} \ \widehat{\mathbf{F}}^{(2)} \ + \ \boldsymbol{\pi}^{(1)} \ \mathbf{F}^{(1)} \ + \ \boldsymbol{\pi}^{(2)} \ \mathbf{L} \ + \ \boldsymbol{\pi}^{(3)} \ \mathbf{B} & = \mathbf{0} \\ \boldsymbol{\pi}^{(0)} \ \widehat{\mathbf{F}}^{(3)} \ + \ \boldsymbol{\pi}^{(1)} \ \mathbf{F}^{(2)} \ + \ \boldsymbol{\pi}^{(2)} \ \mathbf{F}^{(1)} \ + \ \boldsymbol{\pi}^{(3)} \ \mathbf{L} \ + \ \boldsymbol{\pi}^{(4)} \ \mathbf{B} = \mathbf{0} \\ \qquad\qquad\qquad\qquad\qquad \cdots \end{cases} \tag{6}$$

## 3.1    Conditions for stability

We briefly review the conditions that enable us to assert that the CTMC described by the infinitesimal generator $\mathbf{Q}$ in (5) is stable, that is, admits a probability vector satisfying $\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}$ and $\boldsymbol{\pi}\mathbf{1}^T = 1$.

First, observe that the matrix $\widetilde{\mathbf{Q}} = \mathbf{B} + \mathbf{L} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)}$ is an infinitesimal generator, since it has zero row sums and non-negative off-diagonal entries. If $\widetilde{\mathbf{Q}}$ is irreducible, any state $s_i^{(j)}$ in the original process can reach any state $s_{i'}^{(j')}$, for $2 \leq j \leq j'$ and $1 \leq i, i' \leq n$, without having to go through states in $\mathcal{S}^{(l)}$, $l < j$. In this case, for "large values of $j$", the conditional probability of being in $s_i^{(j)}$ given that we are in $\mathcal{S}^{(j)}$ tends to $\widetilde{\pi}_i$, where $\widetilde{\boldsymbol{\pi}}$ is the unique solution of $\widetilde{\boldsymbol{\pi}}\widetilde{\mathbf{Q}} = \mathbf{0}$, subject to $\widetilde{\boldsymbol{\pi}}\mathbf{1}^T = 1$, that is, the stationary solution of the ergodic CTMC having $\widetilde{\mathbf{Q}}$ as the infinitesimal generator[1].

---

[1] For a proof of this statement in the case when only the first $p$ matrices $\mathbf{F}^{(1)}, \ldots, \mathbf{F}^{(p)}$ can be nonzero, simply observe that, if the process is in one of the states of $\mathcal{S}^{(j)}$, the last $j/p - 1$ state transitions (at least) *must* have been taken according to the rates specified by $\mathbf{B}$, $\mathbf{L}$, and $\mathbf{F}^{(1)}, \ldots, \mathbf{F}^{(p)}$. Thus, for large $j$, the conditional probability of

Then, the M/G/1-type process is stable as long as, for large values of $j$, the forward drift from $\mathcal{S}^{(j)}$ is less than the backward drift from it:

$$\widetilde{\boldsymbol{\pi}} \left( \sum_{j=1}^{\infty} j \mathbf{F}^{(j)} \right) \mathbf{1}^T < \widetilde{\boldsymbol{\pi}} \mathbf{B} \mathbf{1}^T.$$

This stability condition can be verified numerically, and it is easy to see that it is equivalent to the one given by Neuts in [15], $\widetilde{\boldsymbol{\pi}} \boldsymbol{\beta} < 1$, where, in our terminology, the column vector $\boldsymbol{\beta}$ is given by $\boldsymbol{\beta} = \left( \mathbf{L} + \sum_{j=1}^{\infty} j \mathbf{F}^{(j)} \right) \mathbf{1}^T$. As in the scalar case, the condition where $\widetilde{\boldsymbol{\pi}} \boldsymbol{\beta}$ is exactly equal to 1 results in a null-recurrent CTMC. If $\widetilde{\mathbf{Q}}$ is instead reducible, this stability condition must be applied to each subset of $\{1, \ldots, n\}$ corresponding to a recurrent class in the CTMC described by $\widetilde{\mathbf{Q}}$.

## 3.2 Main theorem

As done in [3], we require that $\mathbf{B}$ contains nonzero entries only in one column, which we assume to be the last one, column $n$, without loss of generality. The main idea behind our approach is to transform the infinitely countable set of linear equations $\boldsymbol{\pi} \mathbf{Q} = \mathbf{0}$, subject to $\boldsymbol{\pi} \mathbf{1}^T = 1$, into $m + 2n$ equations in $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and the new aggregate vector of $n$ unknowns $\boldsymbol{\pi}^{(*)} = \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)}$. This finite system can then be solved using ordinary numerical techniques, i.e., either direct methods such as Gaussian elimination, if $m + 2n$ is not too large, or indirect iterative methods such as Gauss-Seidel, which can better exploit the sparsity of the system, thus are applicable even when $m + 2n$ is of the order $10^4$ or $10^5$, provided the sparsity of the original matrices describing $Q$ is sufficiently high, as it is normally the case. Once $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)}$ have been obtained, very general measures of interest can be computed by solving further linear systems, as discussed in Section 4.

We now show the derivation of our $m + 2n$ equations.

(i) The normalization condition offers one equation:

$$\boldsymbol{\pi}^{(0)} \mathbf{1}^T + \boldsymbol{\pi}^{(1)} \mathbf{1}^T + \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \mathbf{1}^T = 1. \tag{7}$$

(ii) The next $m$ equations can be obtained by the first line in (6):

$$\boldsymbol{\pi}^{(0)} \widehat{\mathbf{L}}^{(0)} + \boldsymbol{\pi}^{(1)} \widehat{\mathbf{B}} = \mathbf{0}. \tag{8}$$

(iii) The third set of $n - 1$ equations is from the second line in (6), which defines $n$ equations, fortunately only the last one actually containing a contribution from $\boldsymbol{\pi}^{(2)}$, due to the structure we

---

being in $s_i^{(j)}$ tends to the probability of being in state $i$ in the CTMC with generator $\widetilde{\mathbf{Q}}$. For the case where $\mathbf{F}^{(j)}$ can be nonzero for arbitrarily large $j$, recall that its rates must tend to zero faster than $1/j$ (since these matrices are summable), thus a "truncation" of this "exact" process to a sufficiently large maximum value $p$ for the forward jumps will behave in essentially the same way. In other words, in the case of unbounded forward jumps, the last $j/p - 1$ transitions will have been taken according to the rates in $\widetilde{\mathbf{Q}}$ with a probability, dependent on $p$, which can be made arbitrarily close to one.

require in $\mathbf{B}$. This is of fundamental importance, since $\boldsymbol{\pi}^{(2)}$ is not one of our variables. Hence, we consider only the first $n-1$ equations and write:

$$\boldsymbol{\pi}^{(0)}\widehat{\mathbf{F}}^{(1)}_{1:m,1:n-1} + \boldsymbol{\pi}^{(1)}\mathbf{L}^{(1)}_{1:n,1:n-1} = \mathbf{0}. \tag{9}$$

(iv) Another set of $n-1$ equations is obtained as follows. First, the sum of the remaining lines in (6) gives

$$\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty}\widehat{\mathbf{F}}^{(j)} + \boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}\mathbf{F}^{(j)} + \left(\sum_{j=2}^{\infty}\boldsymbol{\pi}^{(j)}\right)\left(\mathbf{L} + \sum_{j=1}^{\infty}\mathbf{F}^{(j)}\right) + \sum_{j=3}^{\infty}\boldsymbol{\pi}^{(j)}\mathbf{B} = \mathbf{0}.$$

Since $\sum_{j=3}^{\infty}\boldsymbol{\pi}^{(j)} = \boldsymbol{\pi}^{(*)} - \boldsymbol{\pi}^{(2)}$, we again remove the explicit mention of $\boldsymbol{\pi}^{(2)}$ by considering only the first $n-1$ equations:

$$\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty}\widehat{\mathbf{F}}^{(j)}_{1:m,1:n-1} + \boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}\mathbf{F}^{(j)}_{1:n,1:n-1} + \boldsymbol{\pi}^{(*)}\left(\mathbf{L}_{1:n,1:n-1} + \sum_{j=1}^{\infty}\mathbf{F}^{(j)}_{1:n,1:n-1}\right) = \mathbf{0}. \tag{10}$$

(v) For the last equation, consider the flow balance equations between the states in $\cup_{l=0}^{j}\mathcal{S}^{(l)}$ and those in $\cup_{l=j+1}^{\infty}\mathcal{S}^{(l)}$, for $j \geq 1$,

$$\begin{cases} \boldsymbol{\pi}^{(0)}\sum_{l=2}^{\infty}\widehat{\mathbf{F}}^{(l)}\mathbf{1}^T + \boldsymbol{\pi}^{(1)}\sum_{l=1}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T & = \boldsymbol{\pi}^{(2)}\mathbf{B}\mathbf{1}^T \\ \boldsymbol{\pi}^{(0)}\sum_{l=3}^{\infty}\widehat{\mathbf{F}}^{(l)}\mathbf{1}^T + \boldsymbol{\pi}^{(1)}\sum_{l=2}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T + \boldsymbol{\pi}^{(2)}\sum_{l=1}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T & = \boldsymbol{\pi}^{(3)}\mathbf{B}\mathbf{1}^T \\ \qquad\qquad\qquad\qquad\cdots & \\ \boldsymbol{\pi}^{(0)}\sum_{l=j+1}^{\infty}\widehat{\mathbf{F}}^{(l)}\mathbf{1}^T + \boldsymbol{\pi}^{(1)}\sum_{l=j}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T + \boldsymbol{\pi}^{(2)}\sum_{l=j-1}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T + & \\ \qquad\cdots \qquad + \boldsymbol{\pi}^{(j)}\sum_{l=1}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T & = \boldsymbol{\pi}^{(j+1)}\mathbf{B}\mathbf{1}^T \\ \qquad\qquad\qquad\qquad\cdots & \end{cases} \tag{11}$$

and sum these equations by parts:

$$\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty}\sum_{l=j}^{\infty}\widehat{\mathbf{F}}^{(l)}\mathbf{1}^T + \boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}\sum_{l=j}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T + \sum_{j=2}^{\infty}\boldsymbol{\pi}^{(j)}\sum_{h=1}^{\infty}\sum_{l=h}^{\infty}\mathbf{F}^{(l)}\mathbf{1}^T = \sum_{j=2}^{\infty}\boldsymbol{\pi}^{(j)}\mathbf{B}\mathbf{1}^T.$$

which can be rewritten as:

$$\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty}(j-1)\widehat{\mathbf{F}}^{(j)}\mathbf{1}^T + \boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}j\mathbf{F}^{(j)}\mathbf{1}^T + \boldsymbol{\pi}^{(*)}\left(\sum_{j=1}^{\infty}j\mathbf{F}^{(j)} - \mathbf{B}\right)\mathbf{1}^T = \mathbf{0}. \tag{12}$$

The following theorem states that these equations are sufficient to compute $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)}$.

**Theorem 1 (Etaqa-MG1).** Given an ergodic CTMC with infinitesimal generator $\mathbf{Q}$ having the

| $\mathbf{v}^{[1]}$ | $\mathbf{v}^{[2]}$ through $\mathbf{v}^{[m+1]}$ | $\mathbf{v}^{[m+2]}$ through $\mathbf{v}^{[m+n]}$ | $\mathbf{v}^{[m+n+1]}$ through $\mathbf{v}^{[m+2n]}$ | $\mathbf{v}^{[m+2n+1]}$ through $\mathbf{v}^{[m+3n]}$ | $\cdots$ | $\mathbf{z}^{[1]}$ through $\mathbf{z}^{[n-1]}$ |
|---|---|---|---|---|---|---|
| $\mathbf{1}^T$ | $\widehat{\mathbf{L}}^{(0)}$ | $\widehat{\mathbf{F}}^{(1)}_{1:m,1:n-1}$ | $\widehat{\mathbf{F}}^{(2)}$ | $\widehat{\mathbf{F}}^{(3)}$ | $\cdots$ | $\left(\sum_{j=2}^\infty \widehat{\mathbf{F}}^{(j)}\right)_{1:m,1:n-1}$ |
| $\mathbf{1}^T$ | $\widehat{\mathbf{B}}$ | $\mathbf{L}^{(1)}_{1:n,1:n-1}$ | $\mathbf{F}^{(1)}$ | $\mathbf{F}^{(2)}$ | $\cdots$ | $\left(\sum_{j=1}^\infty \mathbf{F}^{(j)}\right)_{1:n,1:n-1}$ |
| $\mathbf{1}^T$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{L}$ | $\mathbf{F}^{(1)}$ | $\cdots$ | $\left(\sum_{j=1}^\infty \mathbf{F}^{(j)} + \mathbf{L}\right)_{1:n,1:n-1}$ |
| $\mathbf{1}^T$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{B}$ | $\mathbf{L}$ | $\cdots$ | $\left(\sum_{j=1}^\infty \mathbf{F}^{(j)} + \mathbf{L}\right)_{1:n,1:n-1}$ |
| $\mathbf{1}^T$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{B}$ | $\cdots$ | $\left(\sum_{j=1}^\infty \mathbf{F}^{(j)} + \mathbf{L}\right)_{1:n,1:n-1}$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\cdots$ | $\vdots$ |

$$\mathbf{z}^{[n]} = \sum_{j=1}^\infty \sum_{l=j}^\infty \sum_{i=1}^n \mathbf{v}^{[m+ln+i]} = \begin{bmatrix} \sum_{j=1}^\infty \sum_{l=j}^\infty \widehat{\mathbf{F}}^{(l+1)} \mathbf{1}^T \\ \sum_{j=1}^\infty \sum_{l=j}^\infty \mathbf{F}^{(l)} \mathbf{1}^T \\ \left(\mathbf{L} + \sum_{j=1}^\infty \mathbf{F}^{(j)} + \sum_{j=1}^\infty \sum_{l=j}^\infty \mathbf{F}^{(l)}\right)\mathbf{1}^T \\ \left(\mathbf{B} + 2\left(\mathbf{L} + \sum_{j=1}^\infty \mathbf{F}^{(j)}\right) + \sum_{j=1}^\infty \sum_{l=j}^\infty \mathbf{F}^{(l)}\right)\mathbf{1}^T \\ \left(2\mathbf{B} + 3\left(\mathbf{L} + \sum_{j=1}^\infty \mathbf{F}^{(j)}\right) + \sum_{j=1}^\infty \sum_{l=j}^\infty \mathbf{F}^{(l)}\right)\mathbf{1}^T \\ \vdots \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^\infty j\widehat{\mathbf{F}}^{(j+1)} \mathbf{1}^T \\ \sum_{j=1}^\infty j\mathbf{F}^{(j)} \mathbf{1}^T \\ \left(\sum_{j=1}^\infty j\mathbf{F}^{(j)} - \mathbf{B}\right)\mathbf{1}^T \\ \left(\sum_{j=1}^\infty j\mathbf{F}^{(j)} - \mathbf{B}\right)\mathbf{1}^T \\ \left(\sum_{j=1}^\infty j\mathbf{F}^{(j)} - \mathbf{B}\right)\mathbf{1}^T \\ \vdots \end{bmatrix}$$

Figure 2: The column vectors used to prove linear independence.

structure shown in (5), such that the first $n-1$ columns of $\mathbf{B}$ are null, $\mathbf{B}_{1:n,1:n-1} = \mathbf{0}$, and with stationary probability vector $\boldsymbol{\pi} = \left[\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(2)}, \ldots\right]$, the system of linear equations

$$\mathbf{xM} = [1, \mathbf{0}] \tag{13}$$

where $\mathbf{M} \in \mathbb{R}^{(m+2n)\times(m+2n)}$ is defined as follows:

$$\mathbf{M} = \begin{bmatrix} \mathbf{1}^T & \widehat{\mathbf{L}}^{(0)} & \widehat{\mathbf{F}}^{(1)}_{1:m,1:n-1} & \left(\sum_{j=2}^\infty \widehat{\mathbf{F}}^{(j)}\right)_{1:m,1:n-1} & \left(\sum_{j=2}^\infty (j-1)\widehat{\mathbf{F}}^{(j)}\right)\mathbf{1}^T \\ \mathbf{1}^T & \widehat{\mathbf{B}} & \mathbf{L}^{(1)}_{1:n,1:n-1} & \left(\sum_{j=1}^\infty \mathbf{F}^{(j)}\right)_{1:n,1:n-1} & \left(\sum_{j=1}^\infty j\mathbf{F}^{(j)}\right)\mathbf{1}^T \\ \mathbf{1}^T & \mathbf{0} & \mathbf{0} & \left(\sum_{j=1}^\infty \mathbf{F}^{(j)} + \mathbf{L}\right)_{1:n,1:n-1} & \left(\sum_{j=1}^\infty j\mathbf{F}^{(j)} - \mathbf{B}\right)\mathbf{1}^T \end{bmatrix} \tag{14}$$

admits a unique solution $\mathbf{x} = [\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(*)}]$ where $\boldsymbol{\pi}^{(*)} = \sum_{j=2}^\infty \boldsymbol{\pi}^{(j)}$.

**Proof**: The vector $[\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \sum_{j=2}^\infty \boldsymbol{\pi}^{(j)}]$ satisfies (7), (8), (9), (10), and (12), hence it is a solution of (13). To show that this solution is unique, it is enough to prove that the rank of $\mathbf{M}$ is $m + 2n$, i.e., its $m + 2n$ rows are linearly independent.

Since $\mathbf{Q}$ is ergodic, we know that the vector $\mathbf{1}^T$ and the set of vectors corresponding to all the columns of $\mathbf{Q}$ except one (any one of them) are linearly independent. In particular, we choose to

8

remove from $\mathbf{Q}$ the $n^{\text{th}}$ column of the second block of columns in (5), corresponding to the transitions into state $s_n^1$. The result is then the countably infinite set of linearly independent column vectors of $\mathbb{R}^{\mathbb{N}}$, $\{\mathbf{v}^{[1]}, \mathbf{v}^{[2]}, \dots\}$, shown in Fig. 2. We then define $n$ new vectors of $\mathbb{R}^{\mathbb{N}}$, $\{\mathbf{z}^{[1]}, \dots, \mathbf{z}^{[n]}\}$, as follows (see Fig. 2):

- For $i = 1, \dots, n-1$, let $\mathbf{z}^{[i]} = \sum_{j=1}^{\infty} \mathbf{v}^{[m+jn+i]}$, that is we sum the $i^{\text{th}}$ column of each block of columns, starting from the block corresponding to transitions into level $\mathcal{S}^{(2)}$. $\mathbf{B}$ doesn't appear in the expression of these vectors because $\mathbf{B}_{1:n,1:n-1} = \mathbf{0}$.

- Let $\mathbf{z}^{[n]} = \sum_{j=1}^{\infty} \sum_{l=j}^{\infty} \sum_{i=1}^{n} \mathbf{v}^{[m+ln+i]}$. Fig. 2 shows the steps required to obtained a closed-form expression for $\mathbf{z}^{[n]}$, which are proved by recalling that, since $\mathbf{Q}$ is an infinitesimal generator, $(\mathbf{B} + \mathbf{L} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)})\mathbf{1}^T = \mathbf{0}$, which also implies $(\mathbf{L} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)})\mathbf{1}^T = -\mathbf{B}\mathbf{1}^T$.

Then, we can show that the $m + 2n$ vectors $\{\mathbf{v}^{[1]}, \dots, \mathbf{v}^{[m+n]}, \mathbf{z}^{[1]}, \dots, \mathbf{z}^{[n]}\}$ are linearly independent, since the original set $\{\mathbf{v}^{[1]}, \mathbf{v}^{[2]}, \dots\}$ is linearly independent and the vectors $\{\mathbf{z}^{[1]}, \dots, \mathbf{z}^{[n]}\}$ are obtained as linear combinations of different subsets of vectors from $\{\mathbf{v}^{[m+n+1]}, \mathbf{v}^{[m+n+2]}, \dots\}$:

- Disjoint subsets of vectors are used to build $\{\mathbf{z}^{[1]}, \dots, \mathbf{z}^{[n-1]}\}$.

- $\mathbf{z}^{[n]}$ is built using vectors already used for $\{\mathbf{z}^{[1]}, \dots, \mathbf{z}^{[n-1]}\}$, but also vectors of the form $\mathbf{v}^{[m+jn]}$, which are not used to build any of the vectors in $\{\mathbf{z}^{[1]}, \dots, \mathbf{z}^{[n-1]}\}$.

Hence, the matrix having as column the vectors $\{\mathbf{v}^{[1]}, \dots, \mathbf{v}^{[m+n]}, \mathbf{z}^{[1]}, \dots, \mathbf{z}^{[n]}\}$ has rank $m + 2n$, which implies that we must be able to find $m + 2n$ linearly independent rows in it. Since row $m + jn + i$ is identical to row $m + n + i$, for $j > 1$ and $i = 1, \dots, n$, the first $m + 2n$ rows must be linearly independent. These are also the rows of our matrix $\mathbf{M}$, and so the proof is complete. $\square$

# 4 Computing the measures of interest

We now consider the problem of obtaining stationary measures of interest once $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)}$ have been computed. We consider measures that can be expressed as the expected reward rate

$$r = \sum_{j=0}^{\infty} \sum_{i \in \mathcal{S}^{(j)}} \boldsymbol{\rho}_i^{(j)} \boldsymbol{\pi}_i^{(j)},$$

where $\boldsymbol{\rho}_i^{(j)}$ is the *reward rate* of state $s_i^{(j)}$. For example, if we wanted to compute the expected queue length in steady state for a model where $\mathcal{S}^{(j)}$ contains the system states with $j$ customers in the queue, we would let $\boldsymbol{\rho}_i^{(j)} = j$, while, to compute the second moment of the queue length, we would let $\boldsymbol{\rho}_i^{(j)} = j^2$.

Since our solution approach computes $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)} = \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)}$, we rewrite $r$ as

$$r = \boldsymbol{\pi}^{(0)} \boldsymbol{\rho}^{(0)T} + \boldsymbol{\pi}^{(1)} \boldsymbol{\rho}^{(1)T} + \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \boldsymbol{\rho}^{(j)T},$$

9

where $\boldsymbol{\rho}^{(0)} = [\boldsymbol{\rho}_1^{(0)}, \ldots, \boldsymbol{\rho}_m^{(0)}]$ and $\boldsymbol{\rho}^{(j)} = [\boldsymbol{\rho}_1^{(j)}, \ldots, \boldsymbol{\rho}_n^{(j)}]$, for $j \geq 1$. Then, we must show how to compute the above summation without explicitly using the values of $\boldsymbol{\pi}^{(j)}$ for $j \geq 2$. We can do so if the reward rate of state $s_i^{(j)}$, for $j \geq 2$ and $i = 1, \ldots, n$, is a polynomial of degree $k$ in $j$ with arbitrary coefficients $\mathbf{a}_i^{[0]}, \mathbf{a}_i^{[1]}, \ldots, \mathbf{a}_i^{[k]}$:

$$\forall j \geq 2, \ \forall i \in \{1, 2, \ldots, n\}, \qquad \boldsymbol{\rho}_i^{(j)} = \mathbf{a}_i^{[0]} + \mathbf{a}_i^{[1]} j + \cdots + \mathbf{a}_i^{[k]} j^k. \tag{15}$$

The above equation allows the reward rate for each state $s_i^{(j)}$ in the set $\mathcal{S}^{(j)}$ to be a general polynomial function with arbitrary coefficient that may differ from those for other states in $\mathcal{S}^{(j)}$. We emphasize this because, in Section 7, the states within $\mathcal{S}^{(j)}$ may need to have different reward rates, and the method we describe below is general enough to handle these cases as well.

Using the representation of Eq.(15) we write

$$\begin{aligned}
\sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \boldsymbol{\rho}^{(j)T} &= \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \left( \mathbf{a}^{[0]} + \mathbf{a}^{[1]} j + \cdots + \mathbf{a}^{[k]} j^k \right)^T \\
&= \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \mathbf{a}^{[0]T} + \sum_{j=2}^{\infty} j \boldsymbol{\pi}^{(j)} \mathbf{a}^{[1]T} + \cdots + \sum_{j=2}^{\infty} j^k \boldsymbol{\pi}^{(j)} \mathbf{a}^{[k]T} \\
&= \mathbf{r}^{[0]} \mathbf{a}^{[0]T} + \mathbf{r}^{[1]} \mathbf{a}^{[1]T} + \cdots + \mathbf{r}^{[k]} \mathbf{a}^{[k]T},
\end{aligned}$$

and the problem is reduced to the computation of $\mathbf{r}^{[l]} = \sum_{j=2}^{\infty} j^l \boldsymbol{\pi}^{(j)}$, for $l = 0, \ldots, k$ (for an interpretation of $\mathbf{r}^{[l]}$, one can identify it except for the missing term $\boldsymbol{\pi}^{(1)}$, with the $l^{\text{th}}$ vector moment of the random variable representing the level $j$ at which the chain resides).

We show how $\mathbf{r}^{[k]}$, for $k > 0$, can be computed recursively, starting from $\mathbf{r}^{[0]}$, which is simply $\boldsymbol{\pi}^{(*)}$. We first define the following sums:

$$\widehat{\mathbf{F}}^{[j]} = \sum_{l=j}^{\infty} \widehat{\mathbf{F}}^{(l)}, \quad \mathbf{F}^{[j]} = \sum_{l=j}^{\infty} \mathbf{F}^{(l)}, \quad \widehat{\mathbf{F}}^{[c,i]} = \sum_{j=1}^{\infty} (j+c)^i \widehat{\mathbf{F}}^{(j)}, \quad \text{and} \quad \mathbf{F}^{[c,i]} = \sum_{j=1}^{\infty} (j+c)^i \mathbf{F}^{(j)}$$

Multiplying the equations in (6) from the second line on by the appropriate factor $j^k$ results in

$$\left\{ \begin{array}{llllllll}
2^k \boldsymbol{\pi}^{(0)} & \widehat{\mathbf{F}}^{(1)} & + & 2^k \boldsymbol{\pi}^{(1)} & \mathbf{L}^{(1)} & + & 2^k \boldsymbol{\pi}^{(2)} & \mathbf{B} & & & = \mathbf{0} \\
3^k \boldsymbol{\pi}^{(0)} & \widehat{\mathbf{F}}^{(2)} & + & 3^k \boldsymbol{\pi}^{(1)} & \mathbf{F}^{(1)} & + & 3^k \boldsymbol{\pi}^{(2)} & \mathbf{L} & + & 3^k \boldsymbol{\pi}^{(3)} & \mathbf{B} & = \mathbf{0} \\
& & & & \cdots
\end{array} \right. \tag{16}$$

Summing the equations in (16) by parts we obtain

$$\boldsymbol{\pi}^{(0)} \underbrace{\sum_{j=1}^{\infty} (j+1)^k \widehat{\mathbf{F}}^{(j)}}_{=\widehat{\mathbf{F}}^{[1,k]}} + \boldsymbol{\pi}^{(1)} 2^k \mathbf{L}^{(1)} + \boldsymbol{\pi}^{(1)} \underbrace{\sum_{j=1}^{\infty} (j+2)^k \mathbf{F}^{(j)}}_{=\mathbf{F}^{[2,k]}} +$$

$$\sum_{h=2}^{\infty} \boldsymbol{\pi}^{(h)} \left[ \sum_{j=1}^{\infty} (j+h+1)^k \mathbf{F}^{(j)} + (h+1)^k \mathbf{L} + h^k \mathbf{B} \right] = \mathbf{0},$$

10

which can be rewritten as:

$$\boldsymbol{\pi}^{(0)}\widehat{\mathbf{F}}^{[1,k]}+\boldsymbol{\pi}^{(1)}\left(2^k\mathbf{L}^{(1)}+\mathbf{F}^{[2,k]}\right)+\sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}\sum_{j=1}^{\infty}(j+h+1)^k\mathbf{F}^{(j)}+\sum_{l=0}^{k-1}\binom{k}{l}\mathbf{r}^{[l]}\mathbf{L}+\mathbf{r}^{[k]}(\mathbf{L}+\mathbf{B})=\mathbf{0}. \quad (17)$$

We can express the third term on the left-hand-side of the above equation as:

$$
\begin{aligned}
\sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}\sum_{j=1}^{\infty}(j+h+1)^k\mathbf{F}^{(j)} &= \sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}\sum_{l=0}^{k}\binom{k}{l}(h+1)^l\sum_{j=1}^{\infty}j^{k-l}\mathbf{F}^{(j)} \\
&= \sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}\sum_{l=0}^{k}\binom{k}{l}(h+1)^l\mathbf{F}^{[0,k-l]} \\
&= \sum_{l=0}^{k}\binom{k}{l}\sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}(h+1)^l\mathbf{F}^{[0,k-l]} \\
&= \sum_{l=0}^{k}\binom{k}{l}\sum_{t=0}^{l}\binom{l}{t}\sum_{h=2}^{\infty}h^t\boldsymbol{\pi}^{(h)}\mathbf{F}^{[0,k-l]} \\
&= \sum_{l=0}^{k}\binom{k}{l}\sum_{t=0}^{l}\binom{l}{t}\mathbf{r}^{[t]}\mathbf{F}^{[0,k-l]} \\
&= \sum_{l=0}^{k-1}\binom{k}{l}\sum_{t=0}^{l}\mathbf{r}^{[t]}\mathbf{F}^{[0,k-l]} + \sum_{t=0}^{k-1}\binom{k}{t}\mathbf{r}^{[t]}\mathbf{F}^{[0,0]} + \mathbf{r}^{[k]}\mathbf{F}^{[0,0]}
\end{aligned}
$$

and substituting the third term in (17) with the above expression we obtain:

$$
\begin{aligned}
\mathbf{r}^{[k]}(\mathbf{F}^{[0,0]}+\mathbf{L}+\mathbf{B}) = \quad &- \left(\boldsymbol{\pi}^{(0)}\widehat{\mathbf{F}}^{[1,k]}+\boldsymbol{\pi}^{(1)}(2^k\mathbf{L}^{(1)}+\mathbf{F}^{[2,k]})+\sum_{l=0}^{k-1}\binom{k}{l}\mathbf{r}^{[l]}\mathbf{L}\right) \\
&- \left(\sum_{l=0}^{k-1}\binom{k}{l}\sum_{t=0}^{l}\mathbf{r}^{[t]}\mathbf{F}^{[0,k-l]} + \sum_{t=0}^{k-1}\binom{k}{t}\mathbf{r}^{[t]}\mathbf{F}^{[0,0]}\right)
\end{aligned}
$$

which is a linear system of the form $\mathbf{r}^{[k]}(\mathbf{F}^{[0,0]}+\mathbf{L}+\mathbf{B}) = \mathbf{b}^{[k]}$ where $\mathbf{b}^{[k]}$ is an expression that can be computed from $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and the vectors $\mathbf{r}^{[0]}$ through $\mathbf{r}^{[k-1]}$. Since we know that the rank of $\mathbf{F}^{[0,0]}+\mathbf{L}+\mathbf{B}$ (alternatively, $\sum_{j=1}^{\infty}\mathbf{F}^{(j)}+\mathbf{L}+\mathbf{B}$) is $n-1$, we remove the equation corresponding to the last column, resulting in $n-1$ equations

$$\mathbf{r}^{[k]}(\mathbf{F}^{[0,0]}+\mathbf{L})_{1:n,1:n-1} = \mathbf{b}^{[k]}_{1:n-1}. \quad (18)$$

One additional equation is then required. We obtain it from the equations in (12), again by multiplying them by the appropriate $j^k$ and summing them:

$$\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty}j^k\underbrace{\sum_{h=j}^{\infty}\widehat{\mathbf{F}}^{(h)}}_{=\,\widehat{\mathbf{F}}^{[j]}}\mathbf{1}^T+\boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}(j+1)^k\underbrace{\sum_{h=j}^{\infty}\mathbf{F}^{(h)}}_{=\,\mathbf{F}^{[j]}}\mathbf{1}^T+\sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}\sum_{j=1}^{\infty}(j+h)^k\underbrace{\sum_{h=j}^{\infty}\mathbf{F}^{(h)}}_{=\,\mathbf{F}^{[j]}}\mathbf{1}^T = \underbrace{\sum_{j=2}^{\infty}\boldsymbol{\pi}^{(j)}j^k\mathbf{B}\mathbf{1}^T}_{=\,\mathbf{r}^{[k]}\mathbf{B}\mathbf{1}^T},$$

which therefore can be rewritten as:

$$\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty}j^k\widehat{\mathbf{F}}^{[j]}\mathbf{1}^T + \boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}(j+1)^k\mathbf{F}^{[j]}\mathbf{1}^T + \sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}\sum_{j=1}^{\infty}(j+h)^k\mathbf{F}^{[j]}\mathbf{1}^T = \mathbf{r}^{[k]}\mathbf{B}\mathbf{1}^T. \quad (19)$$

11

The third term of the above summation can then be expressed as:

$$
\sum_{h=2}^{\infty} \boldsymbol{\pi}^{(h)} \sum_{j=1}^{\infty} \sum_{l=0}^{k} \binom{k}{l} j^k h^{k-l} \mathbf{F}^{[j]} \mathbf{1}^T = \sum_{h=2}^{\infty} \boldsymbol{\pi}^{(h)} \sum_{t=0}^{k} \binom{k}{t} h^{k-t} \sum_{j=1}^{\infty} j^k \mathbf{F}^{[j]} \mathbf{1}^T =
$$

$$
\sum_{t=0}^{k} \binom{k}{t} \sum_{h=2}^{\infty} \boldsymbol{\pi}^{(h)} h^{k-t} \sum_{j=1}^{\infty} j^k \mathbf{F}^{[j]} \mathbf{1}^T = \sum_{t=0}^{k} \binom{k}{t} \mathbf{r}^{[k-t]} \sum_{j=1}^{\infty} j^k \mathbf{F}^{[j]} \mathbf{1}^T.
$$

Substituting the above in (19) we obtain

$$
\mathbf{r}^{[k]} \left( \sum_{j=1}^{\infty} j^k \mathbf{F}^{[j]} - \mathbf{B} \right) \mathbf{1}^T = - \left( \boldsymbol{\pi}^{(0)} \sum_{j=2}^{\infty} j^k \widehat{\mathbf{F}}^{[j]} \mathbf{1}^T + \boldsymbol{\pi}^{(1)} \sum_{j=1}^{\infty} (j+1)^k \mathbf{F}^{[j]} \mathbf{1}^T + \sum_{t=1}^{k} \binom{k}{t} \mathbf{r}^{[k-t]} \sum_{j=1}^{\infty} j^k \mathbf{F}^{[j]} \right)
$$

$$(20)$$

The right-hand side of the above equation is an expression containing $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and the vectors $\mathbf{r}^{[0]}$ through $\mathbf{r}^{[k-1]}$. The $n \times n$ matrix $\left[ \left( \mathbf{F}^{[0,0]} + \mathbf{L} \right)_{1:n,1:n-1} \middle| \left( \sum_{j=1}^{\infty} j^k \mathbf{F}^{[j]} - \mathbf{B} \right) \mathbf{1}^T \right]$ required to compute $\mathbf{r}^{[k]}$ has full rank; the proof is analogous to that of Theorem 1. The above result can be used to find any moment $k$, provided that the first $k-1$ moments have been previously computed.

As an example, we consider $\mathbf{r}^{[1]}$, which is used to compute measures such as the first moment of the queue length. In this case,

$$
\mathbf{b}^{[1]} = -\left( \boldsymbol{\pi}^{(0)} \sum_{j=1}^{\infty} (j+1) \widehat{\mathbf{F}}^{(j)} + \boldsymbol{\pi}^{(1)} (2\mathbf{L}^{(1)} + \sum_{j=1}^{\infty} (j+2) \mathbf{F}^{(j)}) + \boldsymbol{\pi}^{(*)} (\mathbf{L} + \sum_{j=1}^{\infty} (j+1) \mathbf{F}^{(j)}) \right)
$$

$$
c^{[1]} = -\left( \boldsymbol{\pi}^{(0)} \sum_{j=2}^{\infty} j \widehat{\mathbf{F}}^{[j]} + \boldsymbol{\pi}^{(1)} \sum_{j=1}^{\infty} (j+1) \mathbf{F}^{[j]} + \boldsymbol{\pi}^{(*)} \sum_{j=1}^{\infty} j \mathbf{F}^{[j]} \right) \mathbf{1}^T.
$$

We conclude by observing that some measures might be infinite. For example, if the matrices $\mathbf{F}^{(j)}$ are summable but decrease only like $1/j^h$ for some $h > 1$, then the moments of order $h-1$ or higher for the queue length do not exist (are infinite).

# 5 Bounded bulk arrivals

In this section, we consider processes with bounded bulk arrivals. Such processes can be solved using the original ETAQA [3] or matrix-geometric [14] methods, by merging every $p$ levels into a single larger level, but the results of this section show how to compute the solution without the corresponding increase in complexity (a factor of $p$ for both execution time and storage in ETAQA, a factor of $p^3$ for execution time and $p^2$ for storage in the matrix-geometric method).

If we restrict the process so that it is allowed to jump forward by at most $p$ levels, its infinitesimal

generator $\mathbf{Q}$ still has the structure of (5), except that $\widehat{\mathbf{F}}^{(j)}$ and $\mathbf{F}^{(j)}$ are zero for $j > p$:

$$
\mathbf{Q} = \begin{bmatrix}
\widehat{\mathbf{L}}^{(0)} & \widehat{\mathbf{F}}^{(1)} & \widehat{\mathbf{F}}^{(2)} & \cdots & \widehat{\mathbf{F}}^{(p)} & \mathbf{0} & \mathbf{0} & \cdots \\
\widehat{\mathbf{B}} & \mathbf{L}^{(1)} & \mathbf{F}^{(1)} & \cdots & \mathbf{F}^{(p-1)} & \mathbf{F}^{(p)} & \mathbf{0} & \cdots \\
\mathbf{0} & \mathbf{B} & \mathbf{L} & \cdots & \mathbf{F}^{(p-2)} & \mathbf{F}^{(p-1)} & \mathbf{F}^{(p)} & \cdots \\
\mathbf{0} & \mathbf{0} & \mathbf{B} & \cdots & \mathbf{F}^{(p-3)} & \mathbf{F}^{(p-2)} & \mathbf{F}^{(p-1)} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix}, \tag{21}
$$

We can then formulate the following lemma.

**Lemma 1 (bulk arrivals).** Given an ergodic CTMC with infinitesimal generator $\mathbf{Q}$, having the structure shown in (21) such that the first $n - 1$ columns of $\mathbf{B}$ are null, $\mathbf{B}_{1:n,1:n-1} = \mathbf{0}$, and with stationary probability vector $\boldsymbol{\pi} = \left[ \boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(2)}, \ldots \right]$, the system of linear equations

$$
\mathbf{x} \begin{bmatrix}
\mathbf{1}^T & \widehat{\mathbf{L}}^{(0)} & \widehat{\mathbf{F}}^{(1)}_{1:m,1:n-1} & \left( \sum_{j=2}^{p} \widehat{\mathbf{F}}^{(j)} \right)_{1:m,1:n-1} & \left( \sum_{j=2}^{p} (j-1)\widehat{\mathbf{F}}^{(j)} \right) \mathbf{1}^T \\
\mathbf{1}^T & \widehat{\mathbf{B}} & \mathbf{L}^{(1)}_{1:n,1:n-1} & \left( \sum_{j=1}^{p} \mathbf{F}^{(j)} \right)_{1:n,1:n-1} & \left( \sum_{j=1}^{p} j\mathbf{F}^{(j)} \right) \mathbf{1}^T \\
\mathbf{1}^T & \mathbf{0} & \mathbf{0} & \left( \sum_{j=1}^{p} \mathbf{F}^{(j)} + \mathbf{L} \right)_{1:n,1:n-1} & \left( \sum_{j=1}^{p} j\mathbf{F}^{(j)} - \mathbf{B} \right) \mathbf{1}^T
\end{bmatrix} = [1, \mathbf{0}]
$$

admits a unique solution $\mathbf{x} = [\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(*)}]$ where $\boldsymbol{\pi}^{(*)} = \sum_{i=2}^{\infty} \boldsymbol{\pi}^{(i)}$.

**Proof:** The steps of the proof are exactly the same as those of the theorem introduced in Section 3, hence they are omitted. $\square$

The measure of interest $r$ can also be derived as done in Section 4. Using the same definitions for $r$, $\boldsymbol{\rho}$, and $\mathbf{r}^{[l]}$, we need to show how to express the vectors $\mathbf{r}^{[l]}$, for $l = 0, \ldots, k$. Here, $\mathbf{r}^{[k]}$ can be computed recursively by solving the equation

$$
\mathbf{r}^{[k]} \left[ \left( \sum_{j=1}^{p} \mathbf{F}^{(j)} + \mathbf{L} \right)_{1:n,1:n-1} \, \middle| \, \left( \sum_{j=1}^{p} j^k \mathbf{F}^{[j]} - \mathbf{B} \right) \mathbf{1}^T \right] = \left[ \mathbf{b}^{[k]} \, \middle| \, c^{[k]} \right],
$$

where $\mathbf{b}^{[k]}$ and $c^{[k]}$ are computed using $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\mathbf{r}^{[0]}$ through $\mathbf{r}^{[k-1]}$, following the exact same steps as for the case of unbounded bulks described in Section 4, but adjusted so as to consider bounded bulks only.

## 6 Theoretical complexity

In this section, we present a detailed complexity comparison of ETAQA-MG1 with the matrix-analytic method outlined in Section 2 (in Section 8, we present actual runtimes on an application). First, we note that, under our condition that $\mathbf{B}$ has a single nonzero column, $\mathbf{G}$ does not need

to be computed [18], since it is a zero matrix except for the last column, which contains all ones. This observation is further confirmed by the probabilistic interpretation of $\mathbf{G}$: an entry $(k, l)$ in $\mathbf{G}$ expresses the conditional probability of the process first entering $\mathcal{S}^{(j-1)}$ through state $l$, given that it starts from state $k$ of $\mathcal{S}^{(j)}$ [15][2]. Thus, for the special case of $\mathbf{Q}$ that we consider in this paper, there is no cost associated for computing and storing $\mathbf{G}$.

To obtain $\boldsymbol{\pi}^{(0)}$, the matrix-analytic algorithm requires the computation of the inverse of $\mathbf{S}^{(0)}$, which is, in general, a full matrix. This inverse must be stored for the duration of the entire computation, increasing the storage complexity for the matrix-analytic approach. During the construction of the system of linear equations of Eq. (4), we also need to multiply this (usually full) inverse by sparse $m \times n$ and $n \times m$ matrices, for an overall complexity of $O(n^3 + n\eta\{\widehat{\mathbf{B}}, \sum_{j=1}^{\infty} \widehat{\mathbf{F}}^{(j)}\})$, where $\eta\{\cdot\}$ denotes the total number of nonzero entries in the argument matrices. Finally, the resulting linear system is described by an $m \times m$ matrix, which is, in general, full. Instead, for ETAQA-MG1, we only require the solution of a *sparse* system of $m + 2n$ linear equations, plus some matrix summations and scalar multiplications.

Although there are closed form formulas for the first and the second moment of the queue length, based on the matrix-analytic approach [11], the entire stationary probability vector is in principle required to compute the measures of interest. Using Ramaswami's recursive formula, the vectors $\boldsymbol{\pi}^{(j)}$ can be computed only up to an index $s$ such that $\sum_s^{j=0} \boldsymbol{\pi}^{(j)} \mathbf{1}^T$ is very close to one.

We also note that, in practice, use of either ETAQA-MG1 or the matrix-analytic method requires storing only a finite number of matrices $\widehat{\mathbf{F}}^{(j)}$ and $\mathbf{F}^{(j)}$, enough to accurately describe the M/G/1-type process. In our study, we assume that matrices $\widehat{\mathbf{F}}^{(j)}$ and $\mathbf{F}^{(j)}$ for $j > p$ are zero, as discussed in Section 5. However ETAQA's superiority is more obvious in this case. The complexity of the matrix-analytic algorithm has now an additional cost due to the fact that the matrices $\widehat{\mathbf{S}}^{(j)}$ and $\mathbf{S}^{(j)}$ do not have a closed-form expression, therefore they must be computed for all $j \geq 1$, up to the forward matrix truncation point $p$. Fortunately, computing these matrices involves only full vector operations, and their storage requires only a full vector each (plus the storage for the forward matrices themselves), due to the structure of $\mathbf{G}$ and to the fact that, in our case, $\mathbf{GG} = \mathbf{G}$. Since $\widehat{\mathbf{S}}^{(j)}$ and $\mathbf{S}^{(j)}$ are zero for $j > p$, only $p$ full vectors are additionally required to store these matrices, and, using Eq. 2, we only need to keep a sliding window of the $p + 1$ most recently computed vectors $\boldsymbol{\pi}^{(j)}$. ETAQA-MG1 requires instead only sparse matrix additions and the solution of a sparse system of $m + 2n$ linearly independent equations.

Table 1 summarizes the computational and storage complexity of ETAQA-MG1 and the matrix-analytic method. Since that ETAQA-MG1 is an aggregation-based technique that truly exploits the sparsity of the matrices describing the original infinitesimal generator, its storage and time complexity are superior to the matrix-analytic method. Note that the contribution of $\mathbf{B}$ to the time complexity can be ignored, since it contains at most $n$ nonzero entries.

---

[2]The probabilistic interpretation of $\mathbf{G}$ is the same for both DTMCs and CTMCs. The interpretation in [15, page 81] is consistent with the discussion in [9, page 142], where CTMCs are taken into consideration.

|  | Time complexity | Additional storage |
|---|---|---|
| **Computation of $\boldsymbol{\pi}^{(0)}$ (matrix-analytic) or $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)}$ (Etaqa)** | | |
| ETAQA-MG1 | $O^L(\eta\{\widehat{\mathbf{B}}, \widehat{\mathbf{L}}^{(0)}, \mathbf{L}^{(1)}, \mathbf{L}, \widehat{\mathbf{F}}, \mathbf{F}\})$ | none |
| Matrix-analytic | $O^L(m^2) + O(n\eta\{\widehat{\mathbf{F}}, \widehat{\mathbf{B}}\} + \eta\{\mathbf{F}\} + n^3)$ | $O(m^2 + n^2)$ |
| **First moment measures** | | |
| ETAQA-MG1 | $O^L(\eta\{\mathbf{L}, \mathbf{F}\})$ | none |
| Matrix-analytic | $O(s(p\eta\{\mathbf{F}\} + n^2) + p\eta\{\widehat{\mathbf{F}}\})$ | $O(pn + n^2)$ |

Table 1: Computational and storage complexity of ETAQA-MG1 and the matrix-analytic method. $O^L(x)$ denotes the time complexity of solving a linear system described by $x$ nonzero entries. The expressions $\eta\{\widehat{\mathbf{F}}\}$ and $\eta\{\mathbf{F}\}$ should be taken to mean $\eta\{\sum_{j=1}^{p} \widehat{\mathbf{F}}^{(j)}\}$ and $\eta\{\sum_{j=1}^{p} \mathbf{F}^{(j)}\}$, respectively.

# 7 Repartitioning Q

The approach we introduced in Sections 3 and 5 is very efficient but its applicability is limited by the requirement that $\mathbf{B}$ has only one nonzero column. There are models where this condition on $\mathbf{B}$ is not immediately satisfied yet it can be achieved through an appropriate *repartitioning* of the states. For illustration simplicity, Fig. 3 shows an example of state repartitioning applied to a QBD process, but exactly the same idea would apply to that model structure if it had a more complex forward arc structure, i.e., if it were an M/G/1-type process.
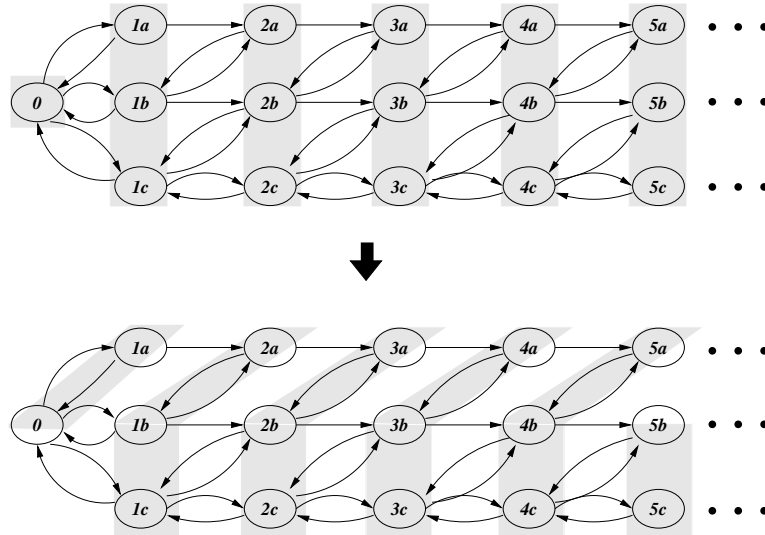


Figure 3: State repartitioning.

Fig. 3 (top) shows a QBD process where the transitions from $\mathcal{S}^{(j)}$ to $\mathcal{S}^{(j-1)}$ have two destinations, $s_b^{(j-1)}$ and $s_c^{(j-1)}$. Clearly, the single column condition for $\mathbf{B}$ is not immediately satisfied. Fig. 3 (bottom) shows instead that, by redefining the sets $\mathcal{S}^{(j)}$ in such a way that $\mathcal{S}^{(1)} = \{s_a^{(2)}, s_b^{(1)}, s_c^{(1)}\}$,

$\mathcal{S}^{(2)} = \{s_a^{(3)}, s_b^{(2)}, s_c^{(2)}\}$, etc., the transitions from $\mathcal{S}^{(j)}$ to $\mathcal{S}^{(j-1)}$ now go to a single state, $s_c^{(j-1)}$. The condition on **B** is then satisfied.

**Lemma 2 (Repartitioning).** An algorithm exists that accepts as input the (finite) block description of the (infinite) infinitesimal generator **Q** of an M/G/1-type process and determines a repartitioning of the state space so that the single-non-zero-column condition for **B** is satisfied, if such a partition exists. The computational complexity of this algorithm is $O(n^2 e \log e)$ and its space complexity is $O(n + e)$, where $e$ is the total number of nonzero entries in $\widetilde{\mathbf{Q}} = \mathbf{B} + \mathbf{L} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)}$.
**Proof:** See Appendix A.  □

We observe that, as a result of this repartitioning, at most $n(n-1)/2$ states might be shifted from the repetitive portion of the chain into $\mathcal{S}^{(0)}$. This is because, among the $n$ sets of states $\mathcal{T}_i = \{s_i^{(j)} : j \geq 2\}$, $k_0 \geq 1$ of them will remain unchanged, $k_1 \leq n - k_0$ of them will have to be shifted one position to the left (i.e., state $s_i^{(j)}$ becomes state $s_i^{(j-1)}$ (adding $k_1$ states to $\mathcal{S}^{(0)}$), $k_2 \leq n - k_0 - k_1$ of them will have to be shifted two positions to the left (adding $2k_2$ states to $\mathcal{S}^{(0)}$), and so on. In the worst case, this adds $1 + 2 + \cdots + n - 1$ states to $\mathcal{S}^{(0)}$.

Furthermore, this change in the definition of the sets $\mathcal{S}^{(j)}$ does not affect our ability to compute the measures of interest. Assume that we have defined the reward rate $\boldsymbol{\rho}_i^{(j)} = \mathbf{a}_i^{[0]} + \mathbf{a}_i^{[1]} j + \cdots + \mathbf{a}_i^{[k]} j^k$ for state $s_i^{(j)}$ in the original partition of the state space. After repartitioning, the "old" state $s_i^{(j)}$ may now be state $\overline{s}_i^{(j-c)}$, for some fixed $c$ (we use a "—" for quantities referring to the new partition). The same expected reward rate will still be computed, that is,

$$r = \sum_{j=0}^{\infty} \sum_{i \in \mathcal{S}^{(j)}} \boldsymbol{\rho}_i^{(j)} \boldsymbol{\pi}_i^{(j)} = \sum_{j=0}^{\infty} \sum_{i \in \overline{\mathcal{S}}^{(j)}} \overline{\boldsymbol{\rho}}_i^{(j)} \overline{\boldsymbol{\pi}}_i^{(j)},$$

provided the new reward rate for state $\overline{s}_i^{(j-c)}$ satisfies $\overline{\boldsymbol{\rho}}_i^{(j-c)} = \boldsymbol{\rho}_i^{(j)}$. In other words, we simply require that $\overline{\mathbf{a}}_i^{[0]} + \overline{\mathbf{a}}_i^{[1]}(j-c) + \cdots + \overline{\mathbf{a}}_i^{[k]}(j-c)^k = \mathbf{a}_i^{[0]} + \mathbf{a}_i^{[1]} j + \cdots + \mathbf{a}_i^{[k]} j^k$, and this can be obviously achieved through an appropriate definition of the coefficients $\overline{\mathbf{a}}_i^{[l]}$.

# 8    Application: multiprocessor scheduling

In this section, we describe an application that can be studied through the ETAQA-MG1 approach. We concentrate on presenting the CTMCs that models the application of interest, focusing on the form of the repeating matrix pattern, and we present numerical results to compare ETAQA-MG1 with the matrix-analytic approach.

Modeling the behavior of scheduling policies in parallel systems often results in CTMCs with matrix-geometric form [21]. Here, we present a CTMC that models the behavior of a class of dynamic scheduling policies where the reconfiguration cost is reduced by limiting how and when processor reallocations can occur [1]. Fig. 4 illustrates the CTMC modeling a dynamic scheduling policy that can only reduce the number of processor allocated to an application, and only immediately after a service completion. Fig. 4 shows the system behavior under bursty arrival conditions modeled

as bulks of finite size (see [19] for an analysis of adaptive space-sharing policies under similar assumptions for the arrival process).
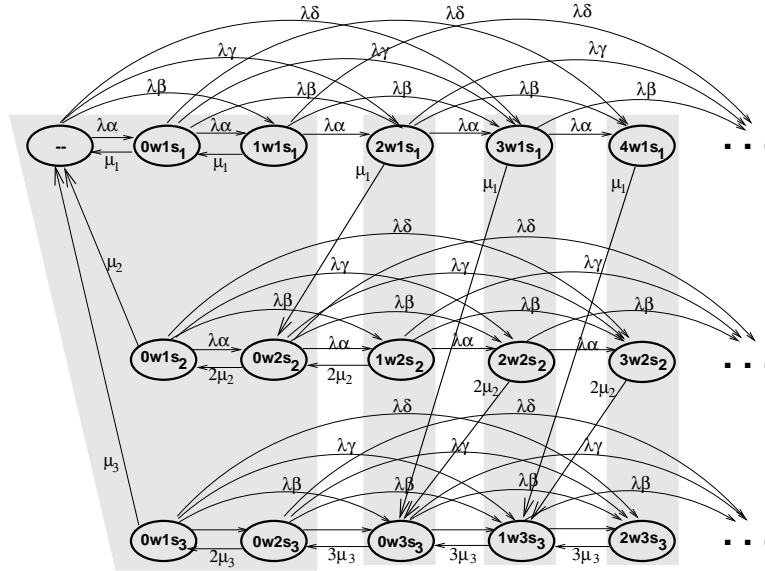


Figure 4: The CTMC that models a dynamic multiprocessor scheduling policy.

The system state is described by the number of applications that are waiting for service in the queue and the number of applications that are in service. $NwMs_r$ denotes that there are $N$ applications in the queue, waiting to be executed, and $M$ applications in service, each having a fraction $\frac{1}{r}$ of the total number of processors allocated to it. Thus, the service rate of an application is $\mu_r$, a function of $r$. The arrival rate for bulks is $\lambda$, and the probability that the size of the bulk is 1, 2, 3, or 4 is $\alpha$, $\beta$, $\gamma$, and $\delta$, respectively.

Our lemma for bounded bulk arrivals can be readily applied to study the performance of this policy. Since the behavior of this dynamic scheduling policy under different workloads is outside the scope of this paper, we do not present numerical results that illustrate the performance of the policy per se. Instead, we present the time for the solution needed by ETAQA-MG1 and the most efficient matrix-analytic method (as presented in Section 2) on a 1Ghz Pentium IV with 512KB cache and 1GB memory. We point out that, for simplicity, Fig. 4 shows bulk arrivals of maximum size four and the multiprocessor partitioned in up to three classes, but our experiments consider maximum bulks of sizes 50 or 100, and several values for the maximum number of classes. Figures 5(a) and 5(b) report the significant run-time improvements achieved when computing $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)}$ using ETAQA-MG1, as opposed to computing $\boldsymbol{\pi}^{(0)}$, and $\boldsymbol{\pi}^{(j)}$, for $1 \leq j \leq s$, using the matrix-analytic method, as a function of the number of multiprocessor partitions and the truncation point $s$ in the computation of the probability vector, i.e., $\sum_{i=0}^{s} \boldsymbol{\pi}^{(i)} 1^T = 1$.

To better observe the efficiency of ETAQA-MG1, Figures 6(a) and 6(b) present the relative computation time of the matrix-analytic method vs. ETAQA-MG1 for the cases presented in Figures 5(a) and 5(b), respectively. ETAQA-MG1 outperforms the matrix-analytic method in both small
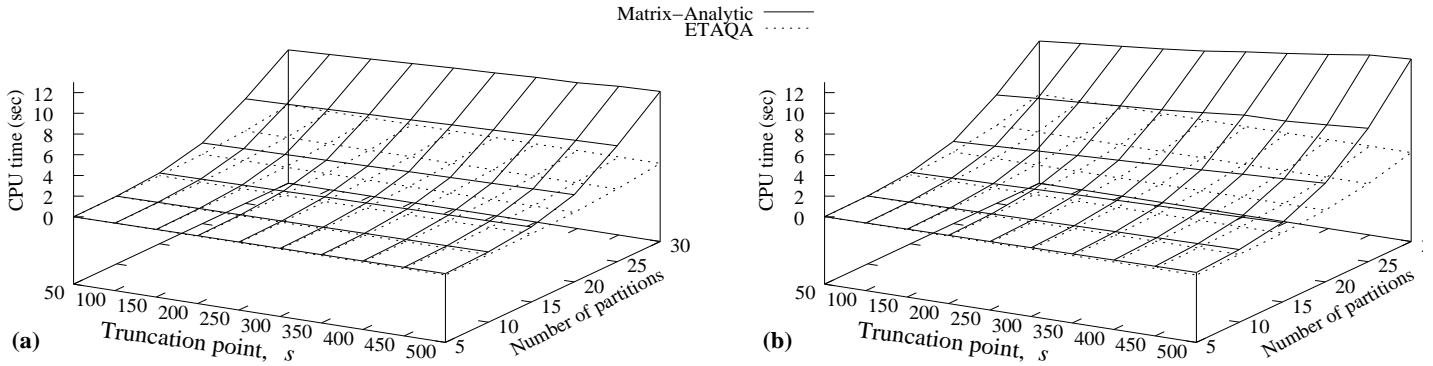
17

Figure 5: Computation time (in sec.) for a process with bulk sizes of (a) 50 and (b) 100.
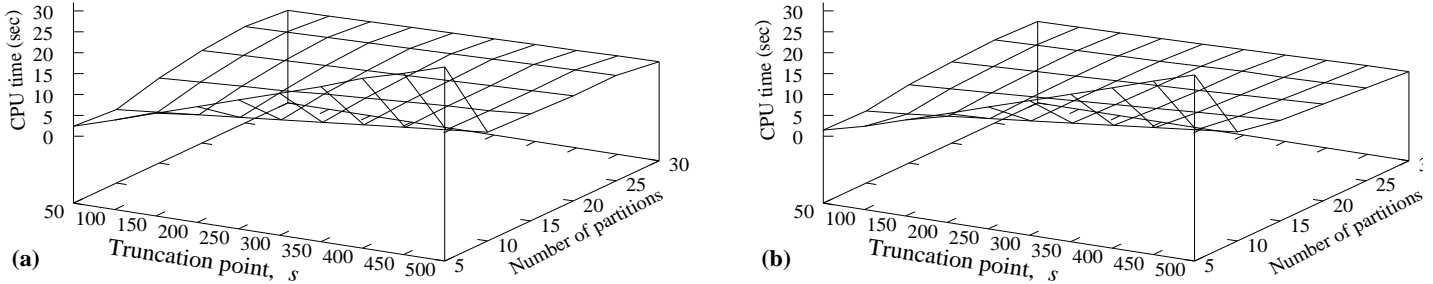


Figure 6: Relative computation time (in sec.) for a process with bulk sizes of (a) 50 and (b) 100.

and large size problems. The efficiency of ETAQA-MG1 increases when the length of the computed probability vector for matrix-analytic increases and when the cardinality $m$ of $\mathcal{S}^{(0)}$ is in the same range as the cardinality $n$ of $\mathcal{S}^{(j)}$, for $j > 0$. In the case under study, $m = O(n^2)$, which explains higher computational gains for ETAQA-MG1 for partitions into small number of classes.

We observe that, for the experiments of Figures 5 and 6, we used the same stopping criteria in the solution of the numerical systems required by ETAQA-MG1 or the matrix-analytic method; indeed, the probability vectors computed by the two methods match very closely. We elaborate further on the numerical stability of our technique in the next section.

## 9    Numerical stability of ETAQA-MG1

ETAQA-MG1 is a numerical approach to solve M/G/1-type processes, and it entails matrix multiplications, additions, and subtractions, as well as the solution of systems of linear equations. This raises the issue of numerical stability for ETAQA-MG1, both in the construction phase of its components and in the solution phase.

We investigate the numerical stability of ETAQA-MG1 experimentally, adopting the definition of a "numerically stable algorithm" given in [22]: *an algorithm is stable if, for an almost exact input,*

*it produces an almost exact output.* As a testbed, we choose a queueing system with bulk arrivals and coxian service, and solve it with both ETAQA-MG1 and matrix-analytic/Ramaswami's formula (which is known to be numerically stable).

For the case under study, the input consists of the parameters for the arrival process and for the coxian service process, while the output consists of $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)}$ (for the matrix-analytic method, we compute $\boldsymbol{\pi}^{(*)}$ as $\sum_{j=1}^{s} \boldsymbol{\pi}^{(j)}$). First, we solve the queueing system with both methods for a chosen set of input parameters, which we refer to as the unperturbed input. Then, we perturb the values of the input parameters with uniformly distributed quantities in $(-\epsilon, \epsilon)$, where $\epsilon$ is a small constant, either $10^{-14}$ or $10^{-12}$. We solve again each model with each method and compare the output with that obtained from solving the problem with unperturbed input using the same method. To increase the statistical confidence, we conduct 50 (differently) perturbed experiments.

The results for perturbations of magnitude $10^{-12}$ and $10^{-14}$, in Figures 7(a) and (b), respectively, indicate that ETAQA-MG1 is indeed a stable algorithm. In Figures 8(a) and 8(b) we present the maximum difference between the solutions obtained from both methods for one of the perturbed experiments for different problem sizes, i.e., different sizes on the matrices of the process and different number of matrices involved.
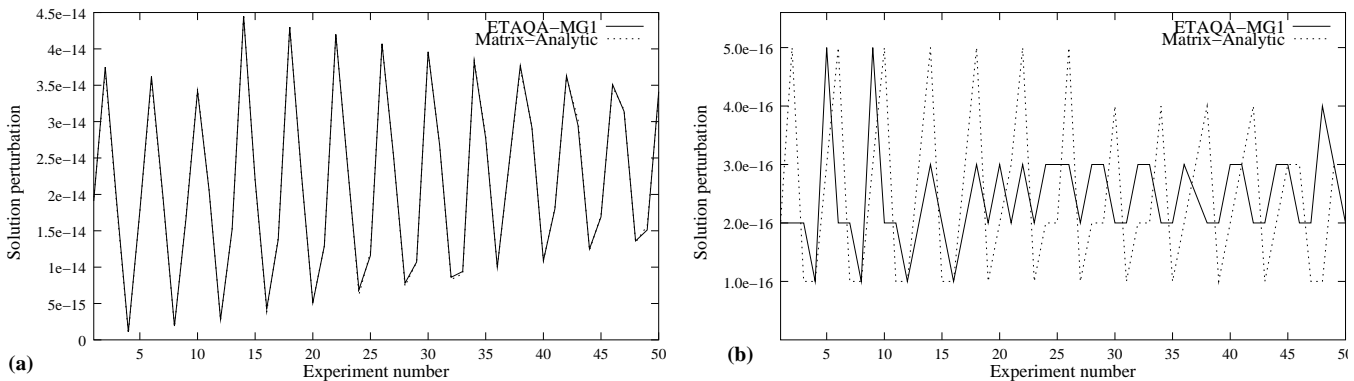


Figure 7: Perturbation of the solution as result of the perturbation of the input for both ETAQA-MG1 and matrix-analytic method for 50 different experiments, with matrix size $n = 32$ and bulk size of $p = 64$, for perturbation magnitudes of (a) $10^{-12}$ and (b) $10^{-14}$.

The results presented in Figures 7 and 8 correspond to cases where the entries within each of the input matrices are in the same range. To study the effect of perturbations on the more problematic "stiff" chains, we consider the same queueing system where entries differ up to six orders of magnitude. The results are in Figures 9 and 10.

For stiff inputs, the resulting perturbation in the solution is higher than for non-stiff inputs. Again, however, ETAQA-MG1 performs similarly to the stable matrix-analytic approach. Thus, while we do not provide a theoretical proof of the numerical stability of ETAQA-MG1, the results of the numerical experiments of this section indicate that ETAQA-MG1 is a stable approach.
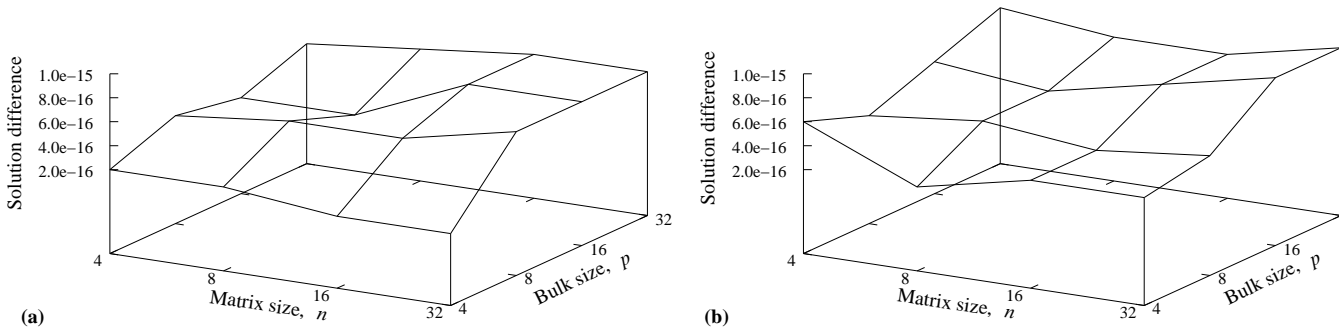
Figure 8: Maximum difference in the solutions obtained from ETAQA-MG1 and the matrix-analytic method for one perturbation experiment as a function of different problem sizes, with perturbation magnitudes of (a) $10^{-12}$ and (b) $10^{-14}$.
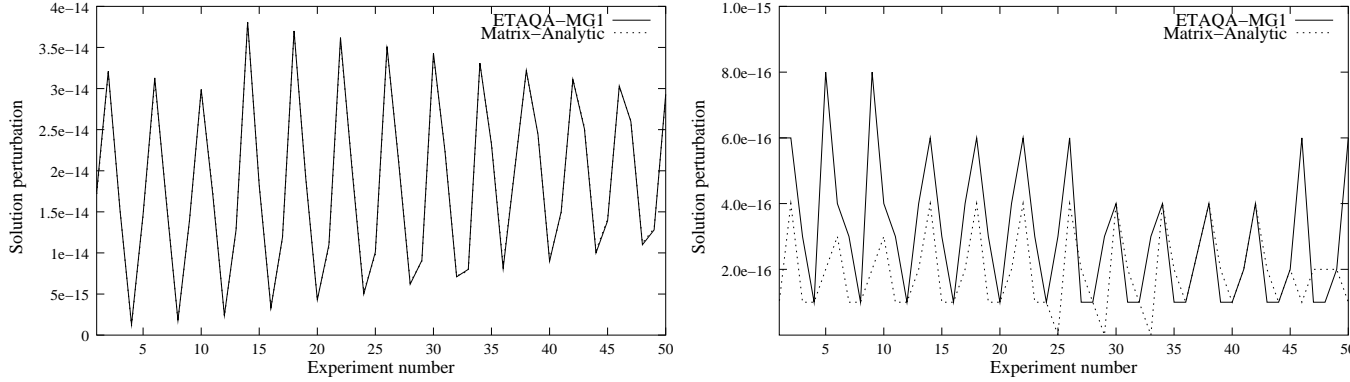


Figure 9: Perturbation of the solution as result of the perturbation of the stiff input for both ETAQA-MG1 and the matrix-analytic method for 50 different experiments, with matrix size $n = 32$ and bulk size of $p = 64$, with perturbation magnitudes of (a) $10^{-12}$ and (b) $10^{-14}$.

# 10    Conclusions and future work

In this paper we presented an extension of the ETAQA approach to M/G/1-type processes. Our exposition focuses on the description of the extended ETAQA methodology and its application to
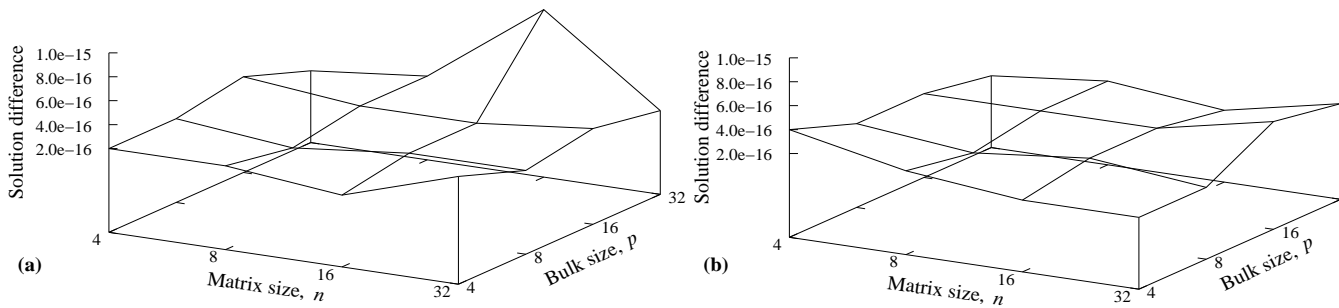


Figure 10: Maximum difference in the solutions obtained from ETAQA-MG1 and thematrix-analytic method for one perturbation experiment with stiff input as a function of different problem sizes, with perturbation magnitudes of (a) $10^{-12}$ and (b) $10^{-14}$.

20

efficiently compute the *exact* probabilities of the boundary states of the process and the aggregate probability distribution of the states in each of the equivalence classes corresponding to a specific partitioning of the remaining infinite portion of the state space. Although the method does not compute the probability distribution of all states, it still provides enough information for the "mathematically exact" computation of a wide variety of Markov reward functions.

We must emphasize that our treatment applies only to M/G/1-type processes where all transitions from one level to the previous one are directed to a single state (transitions within a level or toward higher levels are completely unrestricted). When this condition is satisfied, the solution is derived by solving a system of $m + 2n$ linear equations and provides significant savings over standard algorithms for the solution of general M/G/1-type processes. While the condition on **B** is indisputably restrictive and might not hold on a given CTMC as stated, we also provide an algorithm that finds a repartition of the state space so that the condition becomes satisfied, if such a repartition exists.

In the future, we expect to release a software tool that efficiently implements the extended ETAQA, whose implementation is currently under way. We also intend to explore ways to extend the methodology to a wider class of M/G/1-type processes than that presented in this paper.

## Acknowledgments

We would thank the anonymous referees, whose comments helped us greatly. We also thank Andreas Stathopoulos for giving us advice on addressing the numerical stability of our method.

## References

[1] S.-H. Chiang, R. K. Mansharamani, and M. K. Vernon. Use of application characteristics and limited preemption for run-to-completion parallel processor scheduling policies. In *Proceedings of the 1994 Sigmetrics Conference on Measurement and Modeling of Computer Systems*, pages 33–44, May 1994.

[2] G. Ciardo, A. Riska, and E. Smirni. An aggregation-based solution method for M/G/1-type processes. In B. Plateau, W. J. Stewart, and M. Silva, editors, *Numerical Solution of Markov Chains*, pages 21–40. Prensas Universitarias de Zaragoza, Zaragoza, Spain, Sept. 1999.

[3] G. Ciardo and E. Smirni. ETAQA: An efficient technique for the analysis of QBD-processes by aggregation. *Perf. Eval.*, 36-37:71–93, 1999.

[4] D. R. Cox. A use of complex probabilities in the theory of stochastic processes. *Proc. of the Cambridge Philosophical Society*, 51:313–319, 1955.

[5] W. K. Grassmann and D. A. Stanford. Matrix analytic methods. In W. K. Grassmann, editor, *Computational Probability*, pages 153–204. Kluwer Academic Publishers, Boston, MA, 2000.

[6] D. S. Hochbaum and J. Naor. Simple and fast algorithms for linear and integer programs with two variables per inequality. *SIAM J. Comp.*, 23(6):1179–1192, 1994.

[7] D. B. Johnson. *Algorithms for Shortest Paths*. PhD thesis, Cornell University, 1973.

[8] G. Latouche. Algorithms for infinite Markov chains with repeating columns. In C. Meyer and R. J. Plemmons, editors, *Linear Algebra, Markov Chains, and Queueing Models*, volume 48 of *IMA Volumes in Mathematics and its Applications*, pages 231–265. Springer-Verlag, 1993.

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

[10] E. L. Lawler. *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart and Winston, New York, 1976.

[11] D. M. Lucantoni. *An algorithmic analysis of a communication model with retransmission of flawed messages*. Pitman, Boston, 1983.

[12] B. Meini. Solving M/G/1 type Markov chains: Recent advances and applications. *Stochastic Models*, 14(1 & 2):479–496, 1998.

[13] R. Nelson. *Probability, Stochastic Processes, and Queueing Theory*. Springer-Verlag, 1995.

[14] M. F. Neuts. *Matrix-geometric Solutions in Stochastic Models*. Johns Hopkins University Press, Baltimore, MD, 1981.

[15] M. F. Neuts. *Structured Stochastic Matrices of M/G/1 Type and Their Applications*. Marcel Dekker, Inc., New York, NY, 1989.

[16] V. R. Pratt. Two easy theories whose combination is hard. Technical report, MIT, 1977.

[17] V. Ramaswami. A stable recursion for the steady state vector in Markov chains of M/G/1 type. *Comm. Stat. – Stochastic Models*, 4:183–263, 1988.

[18] V. Ramaswami and G. Latouche. A general class of Markov processes with explicit matrix-geometric solutions. *OR Spektrum*, 8:209–218, Aug. 1986.

[19] E. Rosti, E. Smirni, G. Serazzi, L. W. Dowdy, and K. C. Sevcik. On processor saving scheduling policies for multiprocessor systems. *IEEE Trans. Comp.*, 47(2):178–189, Feb. 1998.

[20] R. Shostak. Deciding linear inequalities by computing loop residues. *J. ACM*, 28(4):769–779, 1981.

[21] M. S. Squillante, F. Wang, and M. Papaefthymiou. Stochastic analysis of gang scheduling in parallel and distributed systems. *Perf. Eval.*, 27/28:273–296, 1996.

[22] L. N. Trefethen and D. B. III. *Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[23] J. van Leeuwen. Graph algorithms. In *Handbook of Theoretical Computer Science*. The MIT Press, Cambridge, MA, 1990.

# A   State repartitioning

## A.1   Problem definition

We seek an algorithm that, given an infinitesimal generator $\mathbf{Q}$ block-partitioned as in Eq. (5), corresponding to a state space initially partitioned as $\mathcal{S}^{(0)} \cup \mathcal{S}^{(1)} \cup \mathcal{S}^{(2)} \ldots$, with $|\mathcal{S}^{(0)}| = m$ and $|\mathcal{S}^{(j)}| = n$ for $j \geq 1$, discovers a new partition $\widetilde{\mathcal{S}}^{(0)} \cup \widetilde{\mathcal{S}}^{(1)} \cup \widetilde{\mathcal{S}}^{(2)} \ldots$, with $|\widetilde{\mathcal{S}}^{(0)}| = \widetilde{m}$ and $|\widetilde{\mathcal{S}}^{(j)}| = n$, i.e., possibly different-size boundary class but same-size repetitive classes, such that the new corresponding block-partition of Eq. (5) is such that $\mathbf{B}$ contains a single nonzero column, if such a repartition exists, or returns a "no" otherwise.

If we just focus on the zero-nonzero pattern of the infinitesimal generator $\mathbf{Q}$, the result is a directed graph. The repeating pattern of $\mathbf{Q}$ implies that there is an edge from $s_i^{(j)}$ to $s_{i'}^{(j)}$ iff there is an edge from $s_i^{(j')}$ to $s_{i'}^{(j')}$, for $i, i' = 1, \ldots, n$ and $j, j' = 1, 2, \ldots$ (since $\mathbf{L}^{(1)}$ can differ from $\mathbf{L}$, exceptions can occur when $j = j' = 1$, but this inessential). We call $s_i^{(1)}, s_i^{(2)}, \ldots$, for $i = 1, \ldots, n$, *same-position* nodes. We call the index $a$ corresponding to the nonzero column in $\mathbf{B}$ the *anchor*. Furthermore, since $\mathbf{Q}$ has an M/G/1-type structure, we know that there are no arcs from $s_i^{(j)}$ to $s_{i'}^{(j')}$, for $j' < j - 1$.

## A.2   An algebraic approach

Since the subgraphs induced by $\mathcal{S}^{(1)}, \mathcal{S}^{(2)}, \ldots$ in the original partition are all isomorphic and those defined by the new sets $\widetilde{\mathcal{S}}^{(1)}, \widetilde{\mathcal{S}}^{(2)}, \ldots$ after the repartition are required to be isomorphic as well, and of the same size, any change we make to the definition of $\mathcal{S}^{(j)}$ should be made also to the other sets $\mathcal{S}^{(j')}$, for $j, j' \geq 1$. For example, if we shift one node from $\mathcal{S}^{(2)}$ to $\mathcal{S}^{(1)}$, the same-position node in $\mathcal{S}^{(j+1)}$ must also be shifted to $\mathcal{S}^{(j)}$ for $j \geq 2$. Applying the same moving action (shift left, shift right, or leave alone) to same-position nodes guarantees the isomorphism of the resulting subgraphs and that of the forward and backward edges between different subgraphs. As a result, we only need determine where to move the nodes of one of the repetitive subgraphs, say $\mathcal{S}^{(j)}$, to form the new partition. The other subgraphs follow the same pattern.

Consider this subgraph, which contains nodes $s_1^{(j)}, s_2^{(j)}, \ldots, s_n^{(j)}$. After the repartition some nodes will be shifted left to subgraphs with indices smaller than $j$, some will go right to subgraphs with indices larger than $j$, and some will stay in their current home. Let $x_i^{(j)}$ be the index of the subgraph to which node $s_i^{(j)}$ will be moved by the repartition. For the time being, suppose that we know the position $a$ of the anchor. We then apply the following rules to establish a linear system of integer inequalities.

- For each edge with a starting point in $\mathcal{S}^{(j)}$, say $s_i^{(j)}$, to $s_l^{(j+r)}$, $l$ other than the anchor $a$, we

must ensure that this edge will not be a backward edge in the new partition, thus

$$x_i^{(j)} \leq x_l^{(j+r)} = x_l^{(j)} + r.$$

- For each edge with a starting point in $\mathcal{S}^{(j)}$, say $s_i^{(j)}$, to an anchor node $s_a^{(j+r)}$, we must ensure that it will not be a backward edge going back more than one level, thus

$$x_i^{(j)} \leq x_l^{(j+r)} + 1 = x_l^{(j)} + r + 1.$$

(in the above, $r = -1$, $r = 0$, or $r > 0$ if the edge is originally a backward, local, or forward edge, respectively).

We then obtain a system of inequalities in $n$ unknowns, $x_1, \ldots, x_n$, with each inequality of the form

$$x_k \leq x_l + c,$$

where $c$ is an integer no smaller than $-1$, and we have dropped the superscripts "$(j)$", since no confusion can arise. Clearly, the number of inequalities to be considered equals the number of off-diagonal nonzero entries in $\mathbf{B} + \mathbf{L} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)}$, we let this quantity be $e$, the number of "edges". Thus, $\mathbf{Q}$ has a new partition with a prespecified anchor $a$ satisfying the condition on $\mathbf{B}$ iff the linear system of inequalities has an integer solution.

## A.3    A necessary and sufficient condition

For the linear system, denoted by $L_a$, developed in the previous section under the assumption that $a$ is the anchor, we can construct a graph, denoted by $G_a$, where there is a node for each variable and there is an edge from node $x_l$ to node $x_k$ with weight $c$ iff $x_k \leq x_l + c$ is one of the inequalities in the system.

The sufficient and necessary condition for the linear system $L_a$ to have an integer solution is that there are no cycles with negative total weight in graph $G_a$. This biimplication was observed by Pratt [16] and later further discussed by Shostak [20], hence the detailed proof is omitted here.

According to van Leeuwen [23], there are efficient algorithms to determine whether there is a negative-weight cycle in a directed graph. One of these algorithms [7, 10] achieves $O(ne)$ worst-case time complexity, if $G_a$ has $n$ nodes and at $e$ edges. Thus we conclude that determining whether the linear system $L_a$ has an integer solution takes $O(ne)$ time. Next, we need to show how to construct a solution if there is one.

## A.4    Solving the linear system

Hochbaum and Naor [6] presented an algorithm that finds a feasible integer solution to a system of monotone inequalities of the form $ax_k \leq bx_l + c$, where $a$ and $b$ are positive. Obviously, our inequalities, $x_k \leq x_l + c$, are monotone. Thus, Hochbaum and Naor's algorithm may be applied.

However, the algorithm requires that each variable $x_k$ has a lower bound $x_k^{min}$ and an upper bound $x_k^{max}$, and its resulting complexity is $O(n^2 e \log e + e \sum_{k=1}^{n}(x_k^{max} - x_k^{min} + 1))$. To use this algorithm, we need to show that our variables are bounded.

Consider any system of inequalities of the form $x_k \leq x_l + c$ with at least one integer solution. Without loss of generality, assume that, among the $n$ variables, there is one variable taking the value of $j$ in the solution, and all other variables have the values relative to $j$. For example, if $x_1 = j$, $x_2 = j + 1$, $x_3 = j$, and $x_4 = j - 1$ is a solution to a system with four variables, then for nodes in $\mathcal{S}^{(j)}$, $s_1^{(j)}$ and $s_3^{(j)}$ stay in $\mathcal{S}^{(j)}$, $s_2^{(j)}$ is moved to $\mathcal{S}^{(j+1)}$ by the repartition, and $s_4^{(j)}$ is moved to $\mathcal{S}^{(j-1)}$. We next argue that, if there is an integer solution (with reference parameter $j$) then there is an integer solution with $j - n < x_i < j + n$ for $i = 1, \ldots, n$ (in other words, we never need to shift a state by $n$ or more positions). To do so, we construct a new integer solution with the bounded variables based on the solution given.

Let $\mathcal{V}_i$ be the set of all variables given the value of $i$ in the initial solution. Clearly, $\mathcal{V}_j \neq \emptyset$ according to our assumption. Let $a$ be the smallest index such that $\mathcal{V}_a$ is non-empty and $b$ be the largest index such that $\mathcal{V}_b$ is non-empty. If $a \leq j - n$, there must be at least one empty set ("hole") with index between $a$ and $j$. Let $r$ be the smallest such index. For variables in $\mathcal{V}_a \cup \cdots \cup \mathcal{V}_{r-1}$, we increase their values by one. We argue that the resulting assignment is still a feasible solution. For an inequality $x_k \leq x_l + c$ with both variables in or not in $\mathcal{V}_a \cup \cdots \cup \mathcal{V}_{r-1}$, the equality still holds under the new assignment, since the difference of the values of $x_k$ and $x_l$ is unchanged. For an inequality $x_k \leq x_l + c$ with one variable in $\mathcal{V}_a \cup \cdots \cup \mathcal{V}_{r-1}$ and the other in $\mathcal{V}_s$, for $s > r$, there are two cases. The easy case is $x_k \in \mathcal{V}_s$ and $x_l \in \mathcal{V}_a \cup \cdots \cup \mathcal{V}_{r-1}$. Under the new assignment, the value of $x_k$ is unchanged while the value of $x_l$ is increased by one. The inequality still holds since the increased variable is on the right-hand side of $\leq$. The second case is $x_k \in \mathcal{V}_a \cup \cdots \cup \mathcal{V}_{r-1}$ and $x_l \in \mathcal{V}_s$. Since there is at least one "hole" between $\mathcal{V}_{r-1}$ and $\mathcal{V}_s$, we must have $r - 1 \leq s - 2$, thus $x_k \leq x_l - 2$ before the increase of $x_k$. Increasing the value of $x_k$ by one, we get $x_k \leq x_l - 1 \leq x_l + c$. So the inequality remains true under the new assignment.

We continue increasing values of some variables until the smallest value in the solution is larger than $j - n$. To the variables holding the values greater than or equal to $j + n$ in the solution, we can decrease their values by one at a time using the similar method until all variables have values smaller than $j + n$ in the solution.

Using the procedure described above, we can bound the variables within the range between $j - n$ and $j + n$. This does not change the integer feasibility of our linear system. For any variable, the difference of the upper bound and lower bound is smaller than $2n$. Using Hochbaum and Naor's algorithm, we achieve the time complexity of $O(n^2 e \log e + e \sum_{k=1}^{n}(2n + 1)) = O(n^2 e \log e + en(2n + 1)) = O(n^2 e \log e)$.

## A.5 Our algorithm and its complexity

We next outline our algorithm for the repartitioning problem. Given an initial partition $\mathcal{S}^{(0)}, \mathcal{S}^{(1)}, \ldots,$ for each possible anchor $s_i^{(j)}$, for $i = 1, 2, \ldots, n$, we construct a linear system using the method described in Section A.2 and its corresponding graph $G_i$. Then we determine whether $G_i$ contains any negative-weight cycle. If not, Hochbaum and Naor's algorithm is called to find an integer solution, which is then converted to a new partition of **Q**. Otherwise, we check the next node to see if it can be the anchor. If no node can be the anchor, the algorithm returns a negative answer.

Input: Graph **Q** with partition $\mathcal{S}^{(0)}, \mathcal{S}^{(1)}, \ldots$

Output: A new partition $\widetilde{\mathcal{S}}^{(0)}, \widetilde{\mathcal{S}}^{(1)}, \ldots$, if there is one, or "No" otherwise

For $i$ from 1 to $n$

    Construct the linear system, $L_i$, using $s_i^{(j)}$ as the anchor

    Construct the corresponding graph, $G_i$

    If $G_i$ does not have any negative-weight cycle

        Use Hochbaum and Naor's algorithm to construct an integer solution

        Output the corresponding partition

        Stop

    End if

End for

Output "No"

In any case Hochbaum and Naor's algorithm is called by our algorithm at most once, while the negative-weight cycle detection algorithm may be called $n$ times in the worst case. Therefore, the time complexity of our repartitioning algorithm is $O(n^2 e + n^2 e \log e) = O(n^2 e \log e)$. Since the data structures involved are graphs with $n$ nodes and at most $e$ edges and a linear system with $n$ unknowns and at most $e$ inequalities, the space complexity of our repartitioning algorithm is $O(n + e)$.