Exact Aggregate Solutions for M/G/1-type Markov Processes *

Alma Riska Evgenia Smirni
Department of Computer Science
College of William and Mary
Williamsburg, VA 23187-8795
e-mail: {riska,esmirni}@cs.wm.edu

ABSTRACT
We introduce a new methodology for the exact analysis of M/G/1-type Markov processes. The methodology uses basic, well-known results for Markov chains by exploiting the structure of the repetitive portion of the chain and recasting the overall problem into the computation of the solution of a finite linear system. The methodology allows for the calculation of the aggregate probability of a finite set of classes of states from the state space, appropriately defined. Further, it allows for the computation of a set of measures of interest such as the system queue length or any of its higher moments. The proposed methodology is exact. Detailed experiments illustrate that the methodology is also numerically stable, and in many cases can yield significantly less expensive solutions when compared with other methods, as shown by detailed time and space complexity analysis.
Keywords: M/G/1-type processes; aggregation; matrix analytic method; Markov chains.

1. INTRODUCTION
Over the last two decades, considerable effort has been put into the development of *matrix analytic techniques* [10, 11] for the exact analysis of a general and frequently encountered class of queuing models. In these models, the embedded Markov chains are two-dimensional generalizations of elementary GI/M/1 and M/G/1 queues [9], and their intersection, i.e., quasi-birth-death (QBD) processes. GI/M/1 and M/G/1 queues model systems with interarrival and service times characterized, respectively, by *general* distributions rather than simple exponentials and are often used as the modeling tool of choice in modern computer and communication systems [9, 15, 18]. As a consequence, considerable effort has been placed into the development of efficient methodologies for their analysis [11, 8].

*This work has been supported by National Science Foundation under grands EIA-9974992, CCR-0098278, and ACI-0090221.

The class of models that can be analyzed using M/G/1-type Markov chains includes the important class of BMAP/G/1 queues, where the arrival process is a batch Markovian arrival process (BMAP) [11]. Special cases of BMAPs include phase-type renewal processes (e.g.., Erlang or Hyperexponential processes) and non-renewal processes (e.g., the Markov modulated Poisson process, MMPP). The importance of BMAPs lies in their ability to be more effective and powerful traffic models than the simple Poisson process or the batch Poisson process, as they can effectively capture dependence and correlation, salient characteristics of Internet traffic [12, 5, 17].

In this paper, we focus on solving M/G/1-type Markov chains. Neuts [10] defines various classes of infinite-state Markov chains with a repetitive structure, whose 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 $j \geq 1$, that correspond to the repetitive portion of the chain. For the class of M/G/1-type Markov chains, the infinitesimal generator $\mathbf{Q}_{M/G/1}$ has upper block Hessenberg form and matrix analytic methods have been proposed for its solution [11]. The key in the matrix-analytic solution is the computation of an auxiliary matrix called $\mathbf{G}$. The traditional solution methodologies for M/G/1-type processes compute the stationary probability vector with a recursive function based on $\mathbf{G}$ and an iterative algorithm is used to determine $\mathbf{G}$ [8, 6].

Distinctively from the classic techniques of solving M/G/1-type processes, we 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. The proposed methodology uses basic, well-known results for Markov chains. Assuming that the state space $\mathcal{S}$ is partitioned into sets $\mathcal{S}^{(j)}$, $j \geq 0$, instead of evaluating the probability distribution of *all* states in each $\mathcal{S}^{(j)}$, we calculate the *aggregate* probability distribution of n classes of states $\mathcal{T}^{(i)}$, $1 \leq i \leq n$, appropriately defined (see Figure 1). The computation of the aggregate probability distribution of the n classes is exact. Furthermore, this aggregate probability distribution does provide the means for calculating a variety of measures of interest including the expected queue length and any of its higher moments.

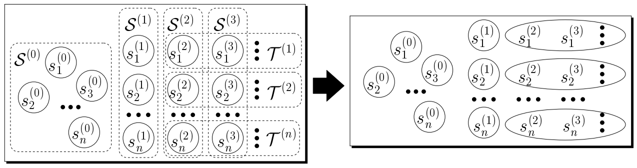

Figure 1: Aggregation of an infinite $\mathcal{S}$ into a finite number of states.

Our contributions are both *theoretical* and *practical*. On the theoretical side, we propose a new methodology for the solution of M/G/1-type processes that requires the solution of a finite linear system of equations. This system of equations is derived from first principles. The new methodology that we propose uses partially the existing matrix analytic methodologies for the computation of matrix $\mathbf{G}$ but in distinct difference from the classic matrix analytic methodologies we use a new approach based on a few simple steps for the computation of the stationary probability vector.

On the practical side, the methodology results in significantly more efficient solutions than the traditional methods. Detailed big-O complexity analysis of the proposed method with the best known methods illustrates the superiority of the new methodology. We further illustrate its benefits by providing detailed experimental comparisons with the best known methods on a set of realistic examples.

An additional important issue that arises is related to the numerical stability of the method. Since the system of linear equations that we derive and solve could be ill-conditioned, there is the danger of proposing a method that is numerically unstable. To address this issue, we conducted detailed experimental analysis. Our experiments consistently provide strong indications that the method is numerically stable.

This paper presents a new method for the exact analysis of M/G/1-type processes. The stated goals and outline of this work are:

- to outline the matrix analytic methods for the solution of M/G/1-type processes (Section 2),
- to present a new methodology along with detailed time and storage complexity analysis for the solution of M/G/1-type processes (Section 3),
- to experimentally compare its efficiency with the best known methods in a set of realistic examples (see Section 4),
- to conduct detailed experiments to illustrate the numerical stability of our methodology (Section 5), and
- to summarize our findings and report on the proposed methodology's efficiency (Section 6).

We note that our approach is in the same spirit as the one presented in [4, 3] for the exact solution of a very limited class of QBD and M/G/1-type Markov chains, but in distinct contrast to these works, our method does not require any restriction on the form of the chain's repeating pattern, thus can be applied to *any* type of M/G/1 chain.

2. BACKGROUND

In this paper, we assume continuous time Markov chains, or CTMCs, hence we refer to the infinitesimal generator $\mathbf{Q}$, but our discussion applies just as well to discrete time Markov chains, or DTMCs. Neuts [10] 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 $j \geq 1$, while $\boldsymbol{\pi}^{(0)}$ and $\boldsymbol{\pi}^{(j)}$, are the stationary probability vectors for states in $\mathcal{S}^{(0)}$ and $\mathcal{S}^{(j)}$, for $j \geq 1$. For the class of M/G/1-type Markov chains, the infinitesimal generator $\mathbf{Q}_{M/G/1}$ is block-partitioned as:

$$\mathbf{Q}_{M/G/1} = \begin{bmatrix} \widehat{\mathbf{L}} & \widehat{\mathbf{F}}^{(1)} & \widehat{\mathbf{F}}^{(2)} & \widehat{\mathbf{F}}^{(3)} & \widehat{\mathbf{F}}^{(4)} & \cdots \\ \widehat{\mathbf{B}} & \mathbf{L} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \mathbf{F}^{(3)} & \cdots \\ \mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{L} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}. \quad (1)$$

We use the letters "L", "F", and "B" according to whether they describe "local", 'forward", and "backward" transition rates, respectively, in relation to a set of states $\mathcal{S}^{(j)}$ for $j \geq 1$, and a "$\widehat{\ }$" for matrices related to $\mathcal{S}^{(0)}$.

For the solution of M/G/1-type processes, several algorithms exist [2, 8, 11]. These algorithms first compute the matrix $\mathbf{G}$ as the solution of the following equation:

$$\mathbf{B} + \mathbf{L}\mathbf{G} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)}\mathbf{G}^{j+1} = \mathbf{0}. \quad (2)$$

The matrix $\mathbf{G}$, which is stochastic if the process is recurrent and irreducible, has an important probabilistic interpretation: 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 $\mathcal{S}^{(j)}$ [11, page 81][1]. The $\mathbf{G}$ matrix is obtained by solving iteratively Eq.(2). However, recent advances show that the computation of $\mathbf{G}$ is more efficient when displacement structures are used based on the representation of M/G/1-type processes by means of QBD processes [8, 2, 1, 6]. The most efficient algorithm for the computation of $\mathbf{G}$ is the cyclic reduction algorithm [2].

The calculation of the stationary probability vector is based on the recursive Ramaswami's formula [14], which is numerically stable because it entails only additions and multiplications[2]. Ramaswami's formula defines the following recursive relation among stationary probability vectors $\boldsymbol{\pi}^{(i)}$ for $i \geq 0$:

$$\boldsymbol{\pi}^{(j)} = -\left(\boldsymbol{\pi}^{(0)}\widehat{\mathbf{S}}^{(j)} + \sum_{k=1}^{j-1} \boldsymbol{\pi}^{(k)}\mathbf{S}^{(j-k)}\right) {\mathbf{S}^{(0)}}^{-1} \quad \forall j \geq 1, \quad (3)$$

where, letting $\mathbf{F}^{(0)} \equiv \mathbf{L}$, matrices $\widehat{\mathbf{S}}^{(j)}$ and $\mathbf{S}^{(j)}$ are defined as follows:

$$\widehat{\mathbf{S}}^{(j)} = \sum_{l=j}^{\infty} \widehat{\mathbf{F}}^{(l)}\mathbf{G}^{l-j}, \; j \geq 1, \quad \mathbf{S}^{(j)} = \sum_{l=j}^{\infty} \mathbf{F}^{(l)}\mathbf{G}^{l-j}, \; j \geq 0 \quad (4)$$

[1]The probabilistic interpretation of $\mathbf{G}$ is the same for both DTMCs and CTMCs.

[2]Subtractions on these type of formulas present the possibility of numerical instability [11, 14].

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

$$\boldsymbol{\pi}^{(0)}\left[\left(\widehat{\mathbf{L}}^{(0)}-\widehat{\mathbf{S}}^{(1)}\mathbf{S}^{(0)^{-1}}\widehat{\mathbf{B}}\right)^{\diamond}\ \mid \mathbf{1}^T-\mathbf{H}\mathbf{1}^T\right]=[\mathbf{0}\mid 1], \quad (5)$$

where $\mathbf{H}=\sum_{j=1}^{\infty}\widehat{\mathbf{S}}^{(j)}\left(\sum_{j=0}^{\infty}\mathbf{S}^{(j)}\right)^{-1}$ and 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 $j\geq 1$, stopping when the accumulated probability mass is close to one. After this point, measures of interest can be computed. Since the relation between $\boldsymbol{\pi}^{(j)}$ for $j\geq 1$ is not straightforward, computation of measures of interest requires generation of the whole stationary probability vector.

[7] gives an improved version of Ramaswami's formula. Once $\boldsymbol{\pi}^{(0)}$ is known using Eq.(5), the stationary probability vector is computed using matrix-generating functions associated with triangular Toeplitz matrices[3]. These matrix-generating functions are computed efficiently by using fast Fourier transforms (FFT).

The outline of this algorithm follows:

$$\begin{array}{ll}\tilde{\boldsymbol{\pi}}^{(1)}=-\mathbf{b}\cdot\mathbf{Y}^{-1} & \\ \tilde{\boldsymbol{\pi}}^{(i)}=-\tilde{\boldsymbol{\pi}}^{(i-1)}\cdot\mathbf{Z}\mathbf{Y}^{-1} & i\geq 2,\end{array} \quad (6)$$

where $\tilde{\boldsymbol{\pi}}^{(1)}=[\boldsymbol{\pi}^{(1)},...,\boldsymbol{\pi}^{(p)}]$ and $\tilde{\boldsymbol{\pi}}^{(i)}=[\boldsymbol{\pi}^{(pi+1)},...,\boldsymbol{\pi}^{(p(i+1))}]$ for $i\geq 2$. Matrices $\mathbf{Y}$, $\mathbf{Z}$, and $\mathbf{b}$ are defined as follows:

$$\mathbf{Y}=\begin{bmatrix}\mathbf{S}^{(1)} & \mathbf{S}^{(2)} & \mathbf{S}^{(3)} & \cdots & \mathbf{S}^{(p)}\\ \mathbf{0} & \mathbf{S}^{(1)} & \mathbf{S}^{(2)} & \cdots & \mathbf{S}^{(p-1)}\\ \mathbf{0} & \mathbf{0} & \mathbf{S}^{(1)} & \cdots & \mathbf{S}^{(p-2)}\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{S}^{(1)}\end{bmatrix},$$

$$\mathbf{Z}=\begin{bmatrix}\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0}\\ \mathbf{S}^{(p)} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0}\\ \vdots & \vdots & \ddots & \vdots & \vdots\\ \mathbf{S}^{(3)} & \mathbf{S}^{(4)} & \cdots & \mathbf{0} & \mathbf{0}\\ \mathbf{S}^{(2)} & \mathbf{S}^{(3)} & \cdots & \mathbf{S}^{(p)} & \mathbf{0}\end{bmatrix},$$

$$\mathbf{b}=\boldsymbol{\pi}^{(0)}\left[\widehat{\mathbf{S}}^{(1)},\ \widehat{\mathbf{S}}^{(2)},\ \widehat{\mathbf{S}}^{(3)},\cdots,\ \widehat{\mathbf{S}}^{(k)}\right],$$

where p is a constant that defines how many of matrices $\mathbf{S}^{(i)}$ and $\widehat{\mathbf{S}}^{(i)}$ are computed. In the above representation, the matrix $\mathbf{Y}$ is an upper block triangular Toeplitz matrix and the matrix $\mathbf{Z}$ is an lower block triangular Toeplitz matrix.

3. M/G/1-AGGREGATE

We introduce an aggregated technique that computes only $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$ and the aggregated stationary probabilities of n classes of states. The block partitioning of the infinitesimal generator as shown in Eq.(1) defines a block partitioning of the stationary probability vector $\boldsymbol{\pi}$ as $\boldsymbol{\pi}=[\boldsymbol{\pi}^{(0)},\ \boldsymbol{\pi}^{(1)},\ \boldsymbol{\pi}^{(2)},...]$ with $\boldsymbol{\pi}^{(0)}\in\mathbb{R}^m$ and $\boldsymbol{\pi}^{(i)}\in\mathbb{R}^n$, for $i\geq 1$. First, we rewrite the matrix equality $\boldsymbol{\pi}\cdot\mathbf{Q}_{M/G/1}=\mathbf{0}$ as:

$$\begin{cases}\boldsymbol{\pi}^{(0)}\cdot\widehat{\mathbf{L}}+\boldsymbol{\pi}^{(1)}\cdot\widehat{\mathbf{B}}=\mathbf{0}\\ \boldsymbol{\pi}^{(0)}\cdot\widehat{\mathbf{F}}^{(1)}+\boldsymbol{\pi}^{(1)}\cdot\mathbf{L}+\boldsymbol{\pi}^{(2)}\cdot\mathbf{B}=\mathbf{0}\\ \boldsymbol{\pi}^{(0)}\cdot\widehat{\mathbf{F}}^{(2)}+\boldsymbol{\pi}^{(1)}\cdot\mathbf{F}^{(1)}+\boldsymbol{\pi}^{(2)}\cdot\mathbf{L}+\boldsymbol{\pi}^{(3)}\cdot\mathbf{B}=\mathbf{0}\\ \boldsymbol{\pi}^{(0)}\cdot\widehat{\mathbf{F}}^{(3)}+\boldsymbol{\pi}^{(1)}\cdot\mathbf{F}^{(2)}+\boldsymbol{\pi}^{(2)}\cdot\mathbf{F}^{(1)}+\boldsymbol{\pi}^{(3)}\cdot\mathbf{L}+\boldsymbol{\pi}^{(4)}\cdot\mathbf{B}=\mathbf{0}\\ \vdots\end{cases} \quad (7)$$

THEOREM 3.1. *Given an ergodic CTMC with infinitesimal generator $\mathbf{Q}_{M/G/1}$ having the structure shown in Eq.(1), with stationary probability vector $\boldsymbol{\pi}=[\boldsymbol{\pi}^{(0)},\ \boldsymbol{\pi}^{(1)},\ \boldsymbol{\pi}^{(2)},\ ...]$ the system of linear equations*

$$\mathbf{x}\cdot\mathbf{X}=[1,\mathbf{0}] \quad (8)$$

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

$$\mathbf{X}=\left[\begin{array}{c|c|c|c}\mathbf{1}^T & \widehat{\mathbf{L}} & \widehat{\mathbf{F}}^{(1)}-\sum_{i=3}^{\infty}\widehat{\mathbf{S}}^{(i)}\cdot\mathbf{G} & (\sum_{i=2}^{\infty}\widehat{\mathbf{F}}^{(i)}+(\sum_{i=3}^{\infty}\widehat{\mathbf{S}}^{(i)}\cdot\mathbf{G})^{\diamond}\\ \mathbf{1}^T & \widehat{\mathbf{B}} & \mathbf{L}-\sum_{i=2}^{\infty}\mathbf{S}^{(i)}\cdot\mathbf{G} & (\sum_{i=1}^{\infty}\mathbf{F}^{(i)}+\sum_{i=2}^{\infty}\mathbf{S}^{(i)}\cdot\mathbf{G})^{\diamond}\\ \mathbf{1}^T & \mathbf{0} & \mathbf{B}-\sum_{i=1}^{\infty}\mathbf{S}^{(i)}\cdot\mathbf{G} & (\sum_{i=1}^{\infty}\mathbf{F}^{(i)}+\mathbf{L}+\sum_{i=1}^{\infty}\mathbf{S}^{(i)}\cdot\mathbf{G})^{\diamond}\end{array}\right] \quad (9)$$

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. We first show that $[\boldsymbol{\pi}^{(0)},\ \boldsymbol{\pi}^{(1)},\ \boldsymbol{\pi}^{(*)}]$ is a solution of Eq.(8) by verifying that it satisfies four matrix equations corresponding to the four sets of columns we used to define $\mathbf{X}$.

(i) The first equation is the normalization constraint:

$$\boldsymbol{\pi}^{(0)}\cdot\mathbf{1}^T+\boldsymbol{\pi}^{(1)}\cdot\mathbf{1}^T+\boldsymbol{\pi}^{(*)}\cdot\mathbf{1}^T=1. \quad (10)$$

(ii) The second set of m equations is the first line in Eq.(7):

$$\boldsymbol{\pi}^{(0)}\cdot\widehat{\mathbf{L}}+\boldsymbol{\pi}^{(1)}\cdot\widehat{\mathbf{B}}=\mathbf{0}. \quad (11)$$

(iii) The third set of n equations is derived beginning from the second line in Eq.(7):

$$\boldsymbol{\pi}^{(0)}\cdot\widehat{\mathbf{F}}^{(1)}+\boldsymbol{\pi}^{(1)}\cdot\mathbf{L}+\boldsymbol{\pi}^{(2)}\cdot\mathbf{B}=\mathbf{0}$$

Because our solution does not compute explicitly $\boldsymbol{\pi}^{(2)}$, we rewrite $\boldsymbol{\pi}^{(2)}$, such that it is expressed in terms of $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$ and $\boldsymbol{\pi}^{(*)}$ only. By substituting $\boldsymbol{\pi}^{(2)}$ in the above equation we obtain:

$$\boldsymbol{\pi}^{(0)}\cdot\widehat{\mathbf{F}}^{(1)}+\boldsymbol{\pi}^{(1)}\cdot\mathbf{L}+\boldsymbol{\pi}^{(*)}\cdot\mathbf{B}-\sum_{j=3}^{\infty}\boldsymbol{\pi}^{(j)}\cdot\mathbf{B}=\mathbf{0}. \quad (12)$$

To compute the sum $\sum_{j=3}^{\infty}\boldsymbol{\pi}^{(j)}$, we use Ramaswami's recursive formula, Eq.(3), and obtain:

$$\begin{array}{l}\boldsymbol{\pi}^{(3)}=-(\boldsymbol{\pi}^{(0)}\widehat{\mathbf{S}}^{(3)}+\boldsymbol{\pi}^{(1)}\mathbf{S}^{(2)}+\boldsymbol{\pi}^{(2)}\mathbf{S}^{(1)})(\mathbf{S}^{(0)})^{-1}\\ \boldsymbol{\pi}^{(4)}=-(\boldsymbol{\pi}^{(0)}\widehat{\mathbf{S}}^{(4)}+\boldsymbol{\pi}^{(1)}\mathbf{S}^{(3)}+\boldsymbol{\pi}^{(2)}\mathbf{S}^{(2)}+\boldsymbol{\pi}^{(3)}\mathbf{S}^{(1)})(\mathbf{S}^{(0)})^{-1}\\ \vdots\end{array} \quad (13)$$

[3]A Toeplitz matrix has equal elements in each of its diagonals facilitating special methods for their analysis.

where the matrices $\widehat{\mathbf{S}}^{(i)}$, for $i \geq 3$, and $\mathbf{S}^{(j)}$, for $j \geq 0$ are determined using the definitions in Eq.(4).

From the definition of matrix $\mathbf{G}$ in Eq.(2), it follows that

$$\mathbf{B} = -(\mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} \mathbf{G}^i) \cdot \mathbf{G} = -\mathbf{S}^{(0)} \cdot \mathbf{G}.$$

After summing all equations in Eq.(13) and multiplying by $\mathbf{B}$, we obtain the sum $\sum_{j=3}^{\infty} \boldsymbol{\pi}^{(j)} \cdot \mathbf{B}$:

$$\begin{aligned} \sum_{j=3}^{\infty} \boldsymbol{\pi}^{(j)} \mathbf{B} \;&= \left(\boldsymbol{\pi}^{(0)} \sum_{i=3}^{\infty} \widehat{\mathbf{S}}^{(i)} + \boldsymbol{\pi}^{(1)} \sum_{i=2}^{\infty} \mathbf{S}^{(i)} + \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \sum_{i=1}^{\infty} \mathbf{S}^{(i)} \right) \\ &\cdot\; (\mathbf{S}^{(0)})^{-1} \mathbf{S}^{(0)} \mathbf{G} \end{aligned}$$

which further results in:

$$\sum_{j=3}^{\infty} \boldsymbol{\pi}^{(j)} \mathbf{B} = \boldsymbol{\pi}^{(0)} \sum_{i=3}^{\infty} \widehat{\mathbf{S}}^{(i)} \mathbf{G} + \boldsymbol{\pi}^{(1)} \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \mathbf{G} + \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \sum_{i=1}^{\infty} \mathbf{S}^{(i)} \mathbf{G}. \tag{14}$$

Substituting Eq.(14) in Eq.(12) we obtain the third set of equations as a function of $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$ and $\boldsymbol{\pi}^{(*)}$ only:

$$\begin{aligned} \boldsymbol{\pi}^{(0)} \left(\widehat{\mathbf{F}}^{(1)} - \sum_{i=3}^{\infty} \widehat{\mathbf{S}}^{(i)} \mathbf{G} \right) + \boldsymbol{\pi}^{(1)} \left(\mathbf{L} - \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \mathbf{G} \right) & \\ + \boldsymbol{\pi}^{(*)} \left(\mathbf{B} - \sum_{i=1}^{\infty} \mathbf{S}^{(i)} \mathbf{G} \right) &= \mathbf{0}. \end{aligned} \tag{15}$$

(iv) Another set of n equations is obtained by summing all the remaining lines in Eq.(7):

$$\begin{aligned} \boldsymbol{\pi}^{(0)} \sum_{i=2}^{\infty} \widehat{\mathbf{F}}^{(i)} + \boldsymbol{\pi}^{(1)} \sum_{i=1}^{\infty} \mathbf{F}^{(i)} + \sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \left(\mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} \right) & \\ + \sum_{j=3}^{\infty} \boldsymbol{\pi}^{(j)} \mathbf{B} \qquad &= \mathbf{0} \end{aligned}$$

Since $\sum_{j=3}^{\infty} \boldsymbol{\pi}^{(j)} \cdot \mathbf{B}$ can be expressed as a function of $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and $\boldsymbol{\pi}^{(*)}$ only, the above equation can be rewritten as:

$$\begin{aligned} & \boldsymbol{\pi}^{(0)} \left(\sum_{i=2}^{\infty} \widehat{\mathbf{F}}^{(i)} + \sum_{i=3}^{\infty} \widehat{\mathbf{S}}^{(i)} \mathbf{G} \right) + \boldsymbol{\pi}^{(1)} \left(\sum_{i=1}^{\infty} \mathbf{F}^{(i)} + \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \mathbf{G} \right) \\ & + \boldsymbol{\pi}^{(*)} \left(\sum_{i=1}^{\infty} \mathbf{F}^{(i)} + \mathbf{L} + \sum_{i=1}^{\infty} \mathbf{S}^{(i)} \mathbf{G} \right) = \mathbf{0}. \end{aligned} \tag{16}$$

In steps (i) through (iv) we showed that the vector of probabilities $[\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(*)}]$ satisfies Eqs. (10), (11), (15), and (16), hence it is a solution of Eq.(8). Now we have to show that this solution is unique. For this, it is enough to prove that the rank of $\mathbf{X}$ is $m + 2n$ by showing that its $m + 2n$ rows are linearly independent.

Since the process with the infinitesimal generator $\mathbf{Q}_{M/G/1}$ is ergodic, we know that the vector $\mathbf{1}^T$ and the set of vectors corresponding to all the columns of $\mathbf{Q}_{M/G/1}$ except one, any of them, are linearly independent. We also note that by multiplying a block column of the infinitesimal generator $\mathbf{Q}_{M/G/1}$ with a matrix, we get a block column which is a linear combination of the columns of the selected block column. In our proof we use multiplication of the block columns with powers of the matrix $\mathbf{G}$. We begin from the columns of the infinitesimal generator. In Figure 2 we show the blocks of column vectors that we use in our proof. The blocks labeled $\mathbf{V}^{(i)}$ for $i \geq 0$ are the original block columns of $\mathbf{Q}_{M/G/1}$. The block $\mathbf{U}$ is obtained by summing all $\mathbf{V}^{(i)}$ for $i \geq 2$:

$$\mathbf{U} = \sum_{i=2}^{\infty} \mathbf{V}^{(i)}.$$

Blocks $\mathbf{W}^{(i)}$ for $i \geq 1$ are obtained by multiplying the block columns $\mathbf{V}^{(j)}$ for $j \geq i+2$ with the $(j-i+1)^{th}$ power of matrix $\mathbf{G}$ and summing them all together

$$\mathbf{W}^{(i)} = \sum_{j=i}^{\infty} \mathbf{V}^{(j+2)} \cdot \mathbf{G}^{j-i+1}, \quad i \geq 1$$

which are used to define

$$\mathbf{Y} = \mathbf{V}^{(1)} - \sum_{i=1}^{\infty} \mathbf{W}^{(i)} \quad \text{and} \quad \mathbf{Z} = \mathbf{U} + \sum_{i=1}^{\infty} \mathbf{W}^{(i)}$$

$$\begin{array}{c|c|c|c|c}
\mathbf{V}^{(0)} & \mathbf{V}^{(1)} & \mathbf{V}^{(2)} & \mathbf{V}^{(3)} & \cdots \\
\hline
\widehat{\mathbf{L}} & \widehat{\mathbf{F}}^{(1)} & \widehat{\mathbf{F}}^{(2)} & \widehat{\mathbf{F}}^{(3)} & \cdots \\
\widehat{\mathbf{B}} & \mathbf{L} & \mathbf{F}^{(1)} & \mathbf{F}^{(2)} & \cdots \\
\mathbf{0} & \mathbf{B} & \mathbf{L} & \mathbf{F}^{(1)} & \cdots \\
\mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{L} & \cdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B} & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{array}
\qquad
\begin{array}{|c|}
\multicolumn{1}{c}{\mathbf{U}} \\
\hline
\sum_{i=2}^{\infty} \widehat{\mathbf{F}}^{(i)} \\
\sum_{i=1}^{\infty} \mathbf{F}^{(i)} \\
\mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} \\
\mathbf{B} + \mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} \\
\mathbf{B} + \mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} \\
\vdots
\end{array}$$

$$\begin{array}{|c|c|c|c}
\multicolumn{1}{c}{\mathbf{W}^{(1)}} & \multicolumn{1}{c}{\mathbf{W}^{(2)}} & \multicolumn{1}{c}{\mathbf{W}^{(3)}} & \cdots \\
\hline
\widehat{\mathbf{S}}^{(3)} \cdot \mathbf{G} & \widehat{\mathbf{S}}^{(4)} \cdot \mathbf{G} & \widehat{\mathbf{S}}^{(5)} \cdot \mathbf{G} & \\
\mathbf{S}^{(2)} \cdot \mathbf{G} & \mathbf{S}^{(3)} \cdot \mathbf{G} & \mathbf{S}^{(4)} \cdot \mathbf{G} & \\
\mathbf{S}^{(1)} \cdot \mathbf{G} & \mathbf{S}^{(2)} \cdot \mathbf{G} & \mathbf{S}^{(3)} \cdot \mathbf{G} & \\
-\mathbf{B} & \mathbf{S}^{(1)} \cdot \mathbf{G} & \mathbf{S}^{(2)} \cdot \mathbf{G} & \\
\mathbf{0} & -\mathbf{B} & \mathbf{S}^{(1)} \cdot \mathbf{G} & \\
\vdots & \vdots & \vdots &
\end{array}$$

$$\begin{array}{|c|}
\multicolumn{1}{c}{\mathbf{Y}} \\
\hline
\widehat{\mathbf{F}}^{(1)} - \sum_{i=3}^{\infty} \widehat{\mathbf{S}}^{(i)} \cdot \mathbf{G} \\
\mathbf{L} - \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\mathbf{B} - \sum_{i=1}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\mathbf{B} - \sum_{i=1}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\mathbf{B} - \sum_{i=1}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\vdots
\end{array}
\qquad
\begin{array}{|c|}
\multicolumn{1}{c}{\mathbf{Z}} \\
\hline
\sum_{i=2}^{\infty} \widehat{\mathbf{F}}^{(i)} + \sum_{i=3}^{\infty} \widehat{\mathbf{S}}^{(i)} \cdot \mathbf{G} \\
\sum_{i=1}^{\infty} \mathbf{F}^{(i)} + \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} + \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} + \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(i)} + \sum_{i=2}^{\infty} \mathbf{S}^{(i)} \cdot \mathbf{G}^i \\
\vdots
\end{array}$$

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

In the matrix $\mathbf{X}$ defined in Eq.(9) we make use of the three upper blocks of $\mathbf{V}^{(0)}$, $\mathbf{Y}$, and $\mathbf{Z}$. We argue that the rank of the matrix $[\mathbf{V}^{(0)} | \mathbf{Y} | \mathbf{Z}]$ is $m + 2n - 1$ because we obtained $\mathbf{Y}$ and $\mathbf{Z}$ respectively as linear combination of blocks $\mathbf{V}^{(1)}$ and $\mathbf{V}^{(2)}$ with the blocks $\mathbf{W}^{(i)}$ for $i \geq 1$, and none of the columns used to generate $\mathbf{W}^{(i)}$ for $i \geq 1$ is from either $\mathbf{V}^{(1)}$ or $\mathbf{V}^{(2)}$. Recall that $\mathbf{Q}_{M/G/1}$ is an infinitesimal generator, therefore the defect is one and the rank of $[\mathbf{V}^{(0)} | \mathbf{Y} | \mathbf{Z}]$ is exactly $m + 2n - 1$. Substituting one (any) of these columns with a column of 1s, we obtain the rank of $m + 2n$. $\square$

We stress that the first step toward the solution of an M/G/1-type process is the computation of matrix $\mathbf{G}$. In our approach we assume that $\mathbf{G}$ is available, i.e., it has been computed using an efficient iterative method, e.g., the cyclic reduction algorithm [2], or that it can be explicitly obtained [13, 15].

3.1 Computing Measures of Interest for M/G/1-type Processes

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, to compute the expected queue length in steady state, where $\mathcal{S}^{(j)}$ represents the system states with j customers in the queue, we let $\boldsymbol{\rho}_i^{(j)} = j$. To compute the second moment of the queue length, we let $\boldsymbol{\rho}_i^{(j)} = j^2$.

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

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

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 $\mathbf{a}_i^{[0]}, \mathbf{a}_i^{[1]}, \ldots, \mathbf{a}_i^{[k]}$:

$$\forall j \geq 2,\ \forall i \in \{1, 2, \ldots, n\}, \qquad \boldsymbol{\rho}_i^{(j)} = \mathbf{a}_i^{[0]} + \mathbf{a}_i^{[1]} j + \cdots + \mathbf{a}_i^{[k]} j^k. \tag{17}$$

The definition of $\boldsymbol{\rho}_i^{(j)}$ illustrates that the set of measures of interest that we can compute includes any moment of the probability vector $\boldsymbol{\pi}$ as long as the reward rate of the i^{th} state in each set $\mathcal{S}^{(j)}$ has the same polynomial coefficients for all $j \geq 2$.

We compute $\sum_{j=2}^{\infty} \boldsymbol{\pi}^{(j)} \boldsymbol{\rho}^{(j)T}$ as follows

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

and the problem is reduced to the computation of $\mathbf{r}^{[l]} = \sum_{j=2}^{\infty} j^l \boldsymbol{\pi}^{(j)}$, for $l = 0, \ldots, k$. We show how $\mathbf{r}^{[k]}$, $k > 0$, can be computed recursively, starting from $\mathbf{r}^{[0]}$, which is simply $\boldsymbol{\pi}^{(*)}$. Multiplying the equations in Eq.(7) from the second line on by the appropriate factor j^k results in

$$\left\{ \begin{array}{l} 2^k \boldsymbol{\pi}^{(0)} \widehat{\mathbf{F}}^{(1)} + 2^k \boldsymbol{\pi}^{(1)} \mathbf{L}^{(1)} + 2^k \boldsymbol{\pi}^{(2)} \mathbf{B} = \mathbf{0} \\ 3^k \boldsymbol{\pi}^{(0)} \widehat{\mathbf{F}}^{(2)} + 3^k \boldsymbol{\pi}^{(1)} \mathbf{F}^{(1)} + 3^k \boldsymbol{\pi}^{(2)} \mathbf{L} + 3^k \boldsymbol{\pi}^{(3)} \mathbf{B} = \mathbf{0} \\ \qquad\qquad\qquad \vdots \end{array} \right.$$

Summing these equations by parts we obtain

$$\underbrace{\boldsymbol{\pi}^{(0)} \sum_{j=1}^{\infty} (j+1)^k \widehat{\mathbf{F}}^{(j)}}_{\stackrel{\text{def}}{=}\ \widehat{\mathbf{f}}} + \underbrace{\boldsymbol{\pi}^{(1)} \left(2^k \mathbf{L} + \sum_{j=1}^{\infty} (j+2)^k \mathbf{F}^{(j)} \right)}_{\stackrel{\text{def}}{=}\ \mathbf{f}} +$$

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

which, since $\sum_{h=2}^{\infty} \boldsymbol{\pi}^{(h)} h^k = \mathbf{r}^{[k]}$, can then be rewritten as

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

Exchanging the order of summations we obtain

$$\sum_{l=0}^{k} \binom{k}{l} \mathbf{r}^{[k-l]} \left(\mathbf{L} + \sum_{j=1}^{\infty} (j+1)^l \mathbf{F}^{(j)} \right) + \mathbf{r}^{[k]} \mathbf{B} = -\widehat{\mathbf{f}} - \mathbf{f}.$$

Finally, isolating the case $l = 0$ in the outermost summation we obtain

$$\begin{aligned} &\mathbf{r}^{[k]} \left(\mathbf{B} + \mathbf{L} + \sum_{j=1}^{\infty} \mathbf{F}^{(j)} \right) = \\ &\quad -\widehat{\mathbf{f}} - \mathbf{f} - \sum_{l=1}^{k} \binom{k}{l} \mathbf{r}^{[k-l]} \left(\mathbf{L} + \sum_{j=1}^{\infty} (j+1)^l \mathbf{F}^{(j)} \right), \end{aligned}$$

which is a linear system of the form $\mathbf{r}^{[k]}(\mathbf{B}+\mathbf{L}+\sum_{i=1}^{\infty} \mathbf{F}^{(j)}) = \mathbf{b}^{[k]}$, where the right-hand side $\mathbf{b}^{[k]}$ is an expression that can be computed from $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$, and the vectors $\mathbf{r}^{[0]}$ through $\mathbf{r}^{[k-1]}$. However, the rank of $(\mathbf{B} + \mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(j)})$ is $n-1$. This is true because $(\mathbf{B} + \mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(j)})$ is an infinitesimal generator with rank $n-1$, so the above system is under-determined. We drop any of the columns of $\mathbf{B} + \mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(j)}$, resulting in

$$\mathbf{r}^{[k]} (\mathbf{B} + \mathbf{L} + \sum_{i=1}^{\infty} \mathbf{F}^{(j)})^{\diamond} = (\mathbf{b}^{[k]})^{\diamond}, \tag{18}$$

and obtain one additional equation for $\mathbf{r}^{[k]}$ by using the flow balance equations between $\cup_{l=0}^{j} \mathcal{S}^{(l)}$ and $\cup_{l=j+1}^{\infty} \mathcal{S}^{(l)}$ for each $j \geq 1$ and multiplying them by the appropriate factor j^k,

$$\left\{ \begin{array}{l} 2^k \boldsymbol{\pi}^{(0)} \sum_{l=2}^{\infty} \widehat{\mathbf{F}}^{(l)} \mathbf{1}^T + 2^k \boldsymbol{\pi}^{(1)} \sum_{l=1}^{\infty} \mathbf{F}^{(l)} \mathbf{1}^T = 2^k \boldsymbol{\pi}^{(2)} \mathbf{B} \mathbf{1}^T \\ 3^k \boldsymbol{\pi}^{(0)} \sum_{l=3}^{\infty} \widehat{\mathbf{F}}^{(l)} \mathbf{1}^T + 3^k \boldsymbol{\pi}^{(1)} \sum_{l=2}^{\infty} \mathbf{F}^{(l)} \mathbf{1}^T \\ \qquad\qquad + 3^k \boldsymbol{\pi}^{(2)} \sum_{l=1}^{\infty} \mathbf{F}^{(l)} \mathbf{1}^T = 3^k \boldsymbol{\pi}^{(3)} \mathbf{B} \mathbf{1}^T \\ \qquad \vdots \end{array} \right. . \tag{19}$$

We introduce the following notation

$$\widehat{\mathbf{F}}_{[k,j]} = \sum_{l=j}^{\infty} l^k \cdot \widehat{\mathbf{F}}^{(l)}, \qquad \mathbf{F}_{[k,j]} = \sum_{l=j}^{\infty} l^k \cdot \mathbf{F}^{(l)}, \qquad j \geq 1, \tag{20}$$

We then sum all lines in Eq.(19) and obtain:

$$\underbrace{\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty} j^k \widehat{\mathbf{F}}_{[0,j]}\mathbf{1}^T}_{\stackrel{\mathrm{def}}{=}\ \hat{f}_c} + \underbrace{\boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}(j+1)^k \mathbf{F}_{[0,j]}\mathbf{1}^T}_{\stackrel{\mathrm{def}}{=}\ f_c} + \sum_{h=2}^{\infty}\boldsymbol{\pi}^{(h)}\sum_{j=1}^{\infty}(j+h)^k\mathbf{F}_{[0,j]}\mathbf{1}^T \quad = \mathbf{r}^{[k]}\mathbf{B}\mathbf{1}^T,$$

which, with steps analogous to those just performed to obtain Eq.(18), can be written as

$$\mathbf{r}^{[k]}(\mathbf{F}_{[1,1]} - \mathbf{B})\mathbf{1}^T = c^{[k]} \tag{21}$$

where $c^{[k]}$ is defined as:

$$c^{[k]} = -(\hat{f}_c + f_c + \sum_{l=1}^{k}\left(\begin{array}{c} k \\ l \end{array}\right)\mathbf{r}^{[k-l]}\sum_{j=1}^{\infty} j^l \mathbf{F}_{[0,j]}\cdot\mathbf{1}^T) \tag{22}$$

Note that the $n \times n$ matrix

$$[(\mathbf{B}+\mathbf{L}+\mathbf{F}_{[0,1]})^{\diamond}|(\mathbf{F}_{[1,1]}-\mathbf{B})\mathbf{1}^T] \tag{23}$$

has full rank. This is true because $(\mathbf{B}+\mathbf{L}+\mathbf{F}_{[0,1]})$ is an infinitesimal generator with rank $n-1$, thus has a unique stationary probability vector $\boldsymbol{\gamma}$ satisfying $\boldsymbol{\gamma}(\mathbf{B}+\mathbf{L}+\mathbf{F}_{[0,1]}) = \mathbf{0}$. However, this same vector must satisfy $\boldsymbol{\gamma}\mathbf{B}\mathbf{1}^T > \boldsymbol{\gamma}\mathbf{F}_{[1,1]}\mathbf{1}^T$ to ensure that the process has a positive drift toward $\mathcal{S}^{(0)}$, thus is ergodic, hence $\boldsymbol{\gamma}(\mathbf{F}_{[1,1]}-\mathbf{B})\mathbf{1}^T < 0$, which shows that $(\mathbf{F}_{[1,1]}-\mathbf{B})\mathbf{1}^T$ cannot be possibly obtained as linear combination of columns in $(\mathbf{B}+\mathbf{L}+\mathbf{F}_{[0,1]})$, therefore the $n \times n$ matrix defined in Eq.(23) has full rank.

Hence, we can compute $\mathbf{r}^{[k]}$ using Eqs. (18) and (21), i.e., solving a linear system in n unknowns (of course, we must do so first for $l = 1, \ldots, k-1$). 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,

$$\begin{aligned}
\mathbf{b}^{[1]} = \quad & -\boldsymbol{\pi}^{(0)}\sum_{j=1}^{\infty}(j+1)\widehat{\mathbf{F}}^{(j)} - \boldsymbol{\pi}^{(1)}(2\mathbf{L}+\sum_{j=1}^{\infty}(j+2)\mathbf{F}^{(j)}) \\
& -\boldsymbol{\pi}^{(*)}(\mathbf{L}+\sum_{j=1}^{\infty}(j+1)\mathbf{F}^{(j)}) \\
c^{[1]} = \quad & -\boldsymbol{\pi}^{(0)}\sum_{j=2}^{\infty} j\widehat{\mathbf{F}}_{[0,j]}\mathbf{1}^T - \boldsymbol{\pi}^{(1)}\sum_{j=1}^{\infty}(j+1)\mathbf{F}_{[0,j]}\mathbf{1}^T \\
& -\boldsymbol{\pi}^{(*)}\sum_{j=1}^{\infty} j\mathbf{F}_{[0,j]}\mathbf{1}^T
\end{aligned}$$

We conclude by observing that, when the sequences $\{\widehat{\mathbf{F}}^{(j)} : j \geq 1\}$ and $\{\mathbf{F}^{(j)} : j \geq 1\}$ do have a nicer relation, like a geometric one, the treatment in this section can be modified appropriately to simplify the different sums introduced here, and give closed form formulas.

In the general case that was considered here some measures might be infinite. For example, if the sequences 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). From the practical point of view, we always store a finite set of matrices from the sequences $\{\widehat{\mathbf{F}}^{(j)} : j \geq 1\}$ and $\{\mathbf{F}^{(j)} : j \geq 1\}$, so the sums of type $\widehat{\mathbf{F}}_{[k,j]}$ and $\mathbf{F}_{[k,j]}$ for $j \geq 1, k \geq 0$ are always finite.

3.2 Time and Storage Complexity

In this section, we present a detailed comparison of our aggregate solution for M/G/1-type processes with the Matrix-analytic method outlined in Section 2. The complexity analysis is within the accuracy of O-notation. In our analysis, $O^L(x)$ denotes the time complexity of solving a linear system described by x nonzero entries and $\eta\{\mathbf{A}\}$ denotes the number of nonzero entries in matrix $\mathbf{A}$. In the general case, $\eta\{\widehat{\mathbf{F}}\}$ and $\eta\{\mathbf{F}\}$ should be taken to mean $\eta\{\cup_{j=1}^{p}\widehat{\mathbf{F}}^{(j)}\}$ and $\eta\{\cup_{j=1}^{p}\mathbf{F}^{(j)}\}$, respectively.

Since practically, we cannot store an infinite number of matrices, we store up to p matrices of type $\widehat{\mathbf{F}}^{(j)}$ and $\mathbf{F}^{(j)}$, $j \geq 1$, given that these matrices accurately capture the behavior of the whole system. Furthermore, for the matrix analytic method to reach the necessary accuracy, it is necessary to compute up to s block vectors $\boldsymbol{\pi}^{(i)}$ of the stationary probability vector $\boldsymbol{\pi}$. We outline the required steps for each method and analyze the computation and storage complexity of each step up to the computation of the expected queue length of the process and summarize the discussion in Table 1[4].

Analysis for M/G/1-Aggregate:

- Computation of the aggregate stationary probability vector $\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(*)}$
  - $O(p \cdot (m \cdot \eta\{\widehat{\mathbf{F}}, \mathbf{G}\} + n \cdot \eta\{\mathbf{F}, \mathbf{G}\}))$ to compute sums of the form $\widehat{\mathbf{S}}^{(i)}$ for $i \geq 1$, and $\mathbf{S}^{(i)}$ for $i \geq 0$, whose sparsity depends directly on the sparsity of $\mathbf{G}$, $\widehat{\mathbf{F}}^{(i)}$ and $\mathbf{F}^{(i)}$ for $i \geq 1$.
  - $O(p \cdot (\eta\{\widehat{\mathbf{F}}\} + \eta\{\mathbf{F}\}))$ to compute sums of the form $\sum_{j=1}^{\infty}\mathbf{F}^{(j)}$, and $\sum_{j=2}^{\infty}\widehat{\mathbf{F}}^{(j)}$.
  - $O^L(\eta\{\widehat{\mathbf{B}}, \widehat{\mathbf{L}}, \mathbf{L}, \widehat{\mathbf{F}}, \mathbf{F}, \mathbf{G}\})$ for the solution of the system of $m + 2n$ linear equations.
- Storage requirements for computation of $\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(*)}$
  - $O(m \cdot n + n^2)$ to store the sums $\sum_{i=1}^{\infty}\widehat{\mathbf{S}}^{(i)}$ and $\sum_{i=1}^{\infty}\mathbf{S}^{(i)}$.
  - $m + 2n$ to store the probability vectors $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{\pi}^{(1)}$ and $\boldsymbol{\pi}^{(*)}$.
- Computation of the expected queue length
  - $O(p \cdot (\eta\{\widehat{\mathbf{F}}\} + \eta\{\mathbf{F}\}))$ to compute $\sum_{j=1}^{\infty}\mathbf{F}^{(j)}$ and $\sum_{j=2}^{\infty}\widehat{\mathbf{F}}^{(j)}$.
  - $O^L(\eta\{\mathbf{F}, \mathbf{L}, \mathbf{B}\})$ for the solution of the sparse system of n linear equations.

Analysis for M/G/1 Matrix-analytic solution

- Computation of the stationary probability vector $\boldsymbol{\pi}$

[4] Since *both* methodologies require the computation of $\mathbf{G}$, we do not include its cost in our complexity analysis. The different sums are considered to be computed only once and stored in the memory untill the computation is completed.

	Computation Complexity	Storage Complexity	π storage
Computation of probability vector			
M/G/1-Aggregate	$O^L(m^2) + O(p(m\eta\{\widehat{\mathbf{F}}, \mathbf{G}\} + n\eta\{\mathbf{F}, \mathbf{G}, \widehat{\mathbf{B}}\}) + pn^3 + sn^2 + p\log p)$	$O(mn + n^2)$	$m + 2n$
Matrix-analytic	$O^L(\eta\{\widehat{\mathbf{B}}, \widehat{\mathbf{L}}, \mathbf{L}, \widehat{\mathbf{F}}, \mathbf{F}, \mathbf{G}\}) + O(p(m\eta\{\widehat{\mathbf{F}}, \mathbf{G}\} + n\eta\{\mathbf{F}, \mathbf{G}\}))$	$O(m^2 + p(mn + n^2))$	$m + sn$
Computation of first moment measure			
M/G/1-Aggregate	$O^L(\eta\{\mathbf{B}, \mathbf{L}, \mathbf{F}\}) + O(p\eta(\widehat{\mathbf{F}}) + p\eta(\mathbf{F}))$	none	none
Matrix-analytic	$O(s \cdot n)$	none	none

Table 1: Computational and storage complexity of the M/G/1-Aggregate and the Matrix-analytic methods for computation of the probability vector ($\pi^{(0)}$, $\pi^{(1)}$, $\pi^{(*)}$ for M/G/1-Aggregate and π for Matrix-analytic) and first moment. We note that the fast FFT-based version of Ramaswami's recursive formula is used for the comparison.

  - $O(p \cdot (m \cdot \eta\{\widehat{\mathbf{F}}, \mathbf{G}\} + n \cdot \eta\{\mathbf{F}, \mathbf{G}\}))$ to compute the sums of the form $\widehat{\mathbf{S}}^{(i)}$ for $i \geq 1$, and $\mathbf{S}^{(i)}$ for $i \geq 0$.
  - $O(n^3 + m \cdot \eta\{\widehat{\mathbf{F}}, \mathbf{G}\} + n \cdot \eta\{\widehat{\mathbf{B}}\})$ for the computation of the inverses of $\mathbf{S}^{(0)}$, and $\sum_{j=0}^{\infty} \mathbf{S}^{(j)}$ and additional multiplications of full matrices.
  - $O^L(m^2)$ for the solution of the system of m linear equations.
  - $O(pn^3 + sn^2 + p\log p)$ [7], since the fast FFT-based version of Ramaswami's recursive formula is used to compute the s vectors of the stationary probability vector.

- Storage requirements for computation of $\boldsymbol{\pi}$
  - $O(p \cdot (m \cdot n + n^2))$ to store all sums of form $\widehat{\mathbf{S}}^{(i)}$ for $i \geq 1$, and $\mathbf{S}^{(i)}$ for $i \geq 0$
  - $O(m^2)$ for storing the matrix of the system of linear equation.
  - m to store $\boldsymbol{\pi}^{(0)}$.
  - $s \cdot n$ to store vectors $\boldsymbol{\pi}^{(i)}$ for $i \geq 1$.
- Computation of the expected queue length
  - $O(s \cdot n)$ to compute the queue length

Concluding our analysis, we point out that the aggregate solution is a more efficient approach, both computation- and storage-wise. In comparison to the Matrix-analytic solution, it entails only a few steps and is thus much easier to implement. Since we do not need to generate the whole stationary probability vector, in our complexity analysis the term s does not appear for M/G/1-Aggregate which in comparison with the value of p or n is several times higher.

Furthermore, since the aggregate solution does not introduce any matrix inversion or matrix multiplication, the sparsity of the original process is preserved resulting in significant savings with respect to both computation and storage. We emphasize the fact that the sparsity of $\mathbf{G}$ is key for preserving the sparsity of the original process, in both methods. There are special cases where $\mathbf{G}$ is very sparse (e.g., $\mathbf{G}$ is a single column matrix if $\mathbf{B}$ is a single column matrix [13]). In these cases, the sums of the form $\widehat{\mathbf{S}}^{(i)}$ for $i \geq 1$, and $\mathbf{S}^{(i)}$ for $i \geq 0$ almost preserve the sparsity of the original process and reduce the computation and storage cost.

4. COMPUTATIONAL EFFICIENCY

In the previous section, we argue using O-notation about the the computational and storage efficiency of M/G/1-Aggregate. Here, we present further numerical evidence supporting the fact that M/G/1-Aggregate is an efficient approach. For our comparisons, we use the classic Ramaswami's formula and the fast FFT implementation of Ramaswami's formula, the most efficient known algorithm for solving M/G/1-type processes [7]. We used Meini's implementation[5] for the cyclic reduction for the computation of $\mathbf{G}$ that is required in all three solution algorithms. We also used Meini's code for the fast FFT implementation of Ramaswami's formula that was made available to us via a personal communication. We implemented the M/G/1-Aggregate method and the classic Ramaswami's formula in C. All experiments were conducted on a 450 MHz Sun Enterprise 420R server with 4 GB memory.

The chain we selected for our experiments represents a general BMAP/M/1 queue. Recall that in practice, it is not possible to store an infinite number of $\widehat{\mathbf{F}}^{(i)}$ and $\mathbf{F}^{(i)}$ matrices, $1 < i < \infty$. One should stop storing when all entries of $\widehat{\mathbf{F}}^{(i)}$ and $\mathbf{F}^{(i)}$ for $i > p$ are below a sufficient threshold (i.e., *very close* to zero in a practical implementation) [6]. As illustrated in the previous section, computation time does depend on the size (i.e., parameters m and n) and the number (of stored) matrices (i.e., parameter p) that define the infinitesimal generator $\mathbf{Q}$. Finally, one last parameter that affects computation time is the number s of vector probabilities that should be computed so as the normalization condition $\sum_{i=1}^{s} \boldsymbol{\pi}^{(i)} = 1.0$ is reached (again, within a sufficient numerical threshold).

In our experiments, we vary the parameters n, p, and s (for the case of BMAP/M/1 queue $m = n$) and provide timing results for the computation of the stationary vector $\boldsymbol{\pi}$ using the classic Ramaswami implementation, the fast FFT implementation, and the computation of $(\boldsymbol{\pi}^{(0)}, \boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(*)})$ using M/G/1-Aggregate. We also provide timings for the computation of the queue length for both methods. Results are presented in Figure 3.

The first experiment, considers a BMAP/M/1 queue with $n = 16$ and $p = 32$, a relatively small case. The timings[6]

[5] Code available at `http://www.dm.unipi.it/~meini/ric.html`.
[6] We point out that our timing results do not take into consideration the computation of $\mathbf{G}$, which is used in all three methods

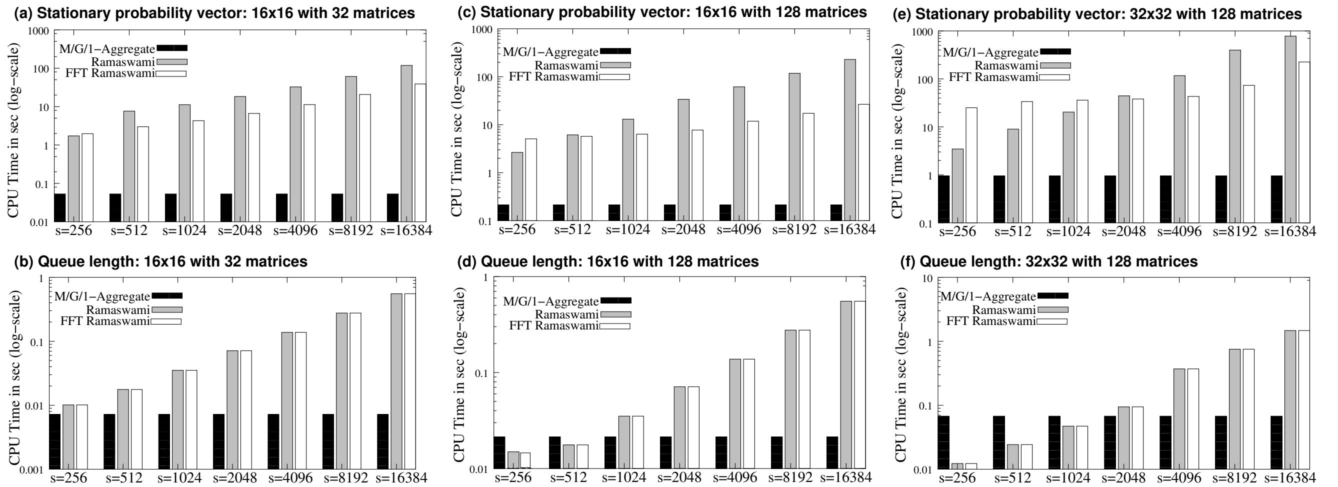

Figure 3: Execution time (in seconds) for M/G/1-Aggregate, classic implementation of Ramaswami's formula, and fast FFT implementation of Ramaswami's formula. Upper row of the figure illustrates timings for computation of the stationary probability vector and the lower row for computation of queue length.

of the three algorithms are shown as a function of s. Figure 3(a) depicts the computation cost of the probability vector and Figure 3(b) illustrates the c computation cost the queue length. Observe that the y-axis is in log-scale. Note that the value of s does affect the execution time of both Matrix-analytic approaches, but does not have any affect on M/G/1-Aggregate. As expected, for the computation of the stationary vector, the FFT implementation is superior to the classic Ramaswami formula, behavior that persists when we increase p and n (see Figures 3(c) and 3(e)). M/G/1-Aggregate consistently outperforms the other two methods, plus its performance is insensitive to s (see figures Figures 3(a), 3(c) and 3(e)).

Figures 3(b), 3(d) and 3(f) illustrate the computation cost of the queue length for the three algorithms for various values of n, p, and s. Note that the two implementations of Ramaswami's formula have the same cost, since the same classic formula is used for the computation of queue length: first weight appropriately and then sum the probability vector which is already computed. The figures further confirm that the cost of solving a small system of linear equations that M/G/1-Aggregate requires for the computations of queue length is in many cases preferable to the classic methods. If this linear system increases and s is also small, then the classic methods may offer better performance.

5. NUMERICAL STABILITY

The algorithm proposed in Section 3 results in a finite system of linear equations that can be solved with numerical methods. Because the linear system is a result of matrix additions, subtractions, and multiplications, its *numerical stability* should be examined. However, because of the presence of a linear system, and because our matrices are not M-matrices, an analytic examination of the numerical stability is not easily feasible. In this section we argue via experimentation that M/G/1-Aggregate is numerically stable and compare its stability behavior with Ramaswami's recursive formula. Ramaswami's recursive formula for the computation of the steady state probability vector of an M/G/1-type process consists of matrix additions of non-negative matrices, computations that are known to be numerically stable [7].

In the following we focus on the *stability of the method* used to solve the original problem, rather than the *stability of the problem* itself. The latter is measured by a condition number (or conditioning), which depends on a specific instance of the problem, but not on the method used. The stability of a method $A : \Re^N \longrightarrow \Re^M$, given an input $x \in \Re^N$, is determined as follows:

$$\|A(x) - A(x + \delta)\| < \kappa(x) \cdot \|\delta\|$$

where $\delta \in \Re^N$ is a small perturbation of the input, and $\kappa(x)$ is the conditioning of the problem with input x. Any norms of the corresponding vector spaces can be used, but in the following we limit our discussion to the infinity (maximum) norm. [19] states that *a stable algorithm gives nearly the right answer to nearly the right question*. In other words, if we change the input of a stable algorithm by a *small* δ we should obtain an output that is perturbed, within the constant factor $\kappa(x)$, by a corresponding amount.

We follow the above definition to examine experimentally the stability of M/G/1-Aggregate versus that of Ramaswami's formula. The output of the aggregate scheme is a probability vector of $m + 2n$ elements and is denoted as $A(x)$, where x belongs to the domain of the method, i.e., it is a choice of all the elements of the input matrices. The output of Ramaswami's is again a probability vector of $m + 2n$ elements and is denoted as $R(x)$. Note that Ramaswami's original output is post-processed to produce the same aggregate probabilities that $A(x)$ produces. We run two sets of experiments, one for a well conditioned instance of the problem, and one for an ill-conditioned instance. This is performed via the following steps:

[7]We opt not to compare M/G/1-Aggregate with the Fast FFT Ramaswami's formula because the FFTs may be the source of numerical instabilities [7].

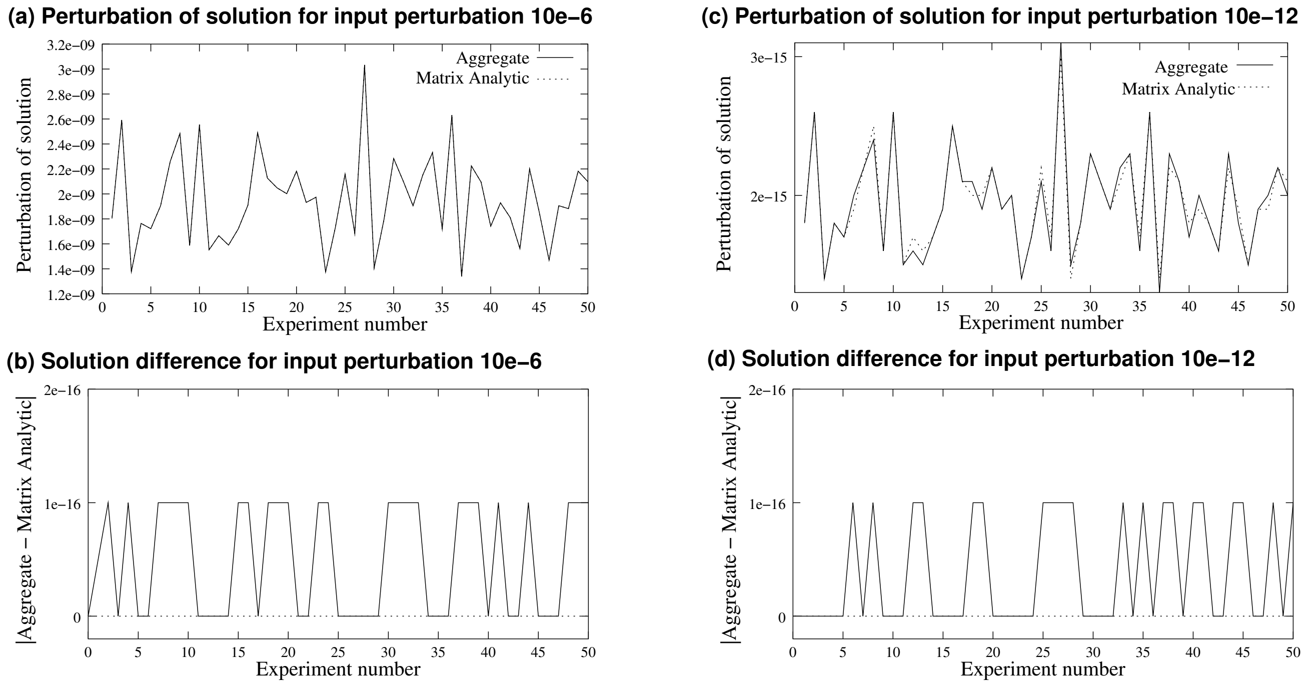

Figure 4: Numerical behavior of algorithms under well-conditioned input. Graphics (a) and (c) plot $\|A(x) - A(x+\delta_i)\|$ for M/G/1-Aggregate and $\|R(x) - R(x+\delta_i)\|$ for Ramaswami's recursive formula for different perturbation magnitudes and for $1 \leq i \leq 50$ distinct experiments. Graphics (b) and (d) illustrate $\|A(x+\delta_i) - R(x+\delta_i)\|$ for the same perturbation magnitudes and the same $1 \leq i \leq 50$ distinct experiments.

1. Select a CTMC and solve it using both M/G/1-Aggregate and Ramaswami's formula and check the closeness of the results.
2. Perturb the input of the selected CTMC by *small* random numbers. We select three different perturbation magnitudes: 10^{-12}, 10^{-10} and 10^{-6}, and solve the CTMC with the perturbed input.
3. Repeat the perturbation experiment 50 times with different sets of random perturbation values within the same magnitude range to achieve statistical accuracy.
4. Calculate the perturbation of output for each randomly perturbed input for M/G/1-Aggregate solutions considering as base case the output obtained by using M/G/1-Aggregate to solve the CTMC without any perturbation of input, i.e., $\|A(x) - A(x+\delta_i)\|$, for each experiment i. Calculate the same for the set of perturbed solutions using Ramaswami's formula, where the base case is the solution obtained using Ramaswami's formula on the un-perturbed input, i.e., $\|R(x) - R(x+\delta_i)\|$, for each experiment i.
5. Calculate the absolute difference between the solution obtained by using M/G/1-Aggregate and Ramaswami's formula, i.e., $\|A(x+\delta_i) - R(x+\delta_i)\|$, for each experiment i.

For our experiments, we selected a CTMC that models a bursty hyper-exponential server with burst sizes ranging from 1 to $p = 64$. The dimension of matrices $\mathbf{B}$, $\mathbf{L}$ and $\mathbf{F}^{(i)}$ for $1 \leq i \leq p$ is 16×16 and matrices $\widehat{\mathbf{L}}$, $\widehat{\mathbf{B}}$ a $\widehat{\mathbf{F}}^{(i)}$ for $1 \leq i \leq p$ are of dimensions 1×1, 16×1 and 1×16 respectively. Since the corresponding $\mathbf{G}$ matrix for the process as well as matrices $\widehat{\mathbf{S}}^{(i)}$ and $\mathbf{S}^{(i)}$ for $1 \leq i \leq p$ are full, we consider the case to a representative one.[8] All experiments are conducted on a Pentium III with 64-bit double precision arithmetic, and 10^{-16} machine precision.

Our first set of experiments considers *well-conditioned* input matrices, where the values of their elements differ at most by two orders of magnitude. Figure 4 illustrates the behavior of M/G/1-Aggregate and Ramaswami's formula under well-conditioned input for 50 distinct experiments. Each experiment corresponds to a different δ_i but within the same magnitude range. Figure 4(a) shows that the perturbation of solution for each of 50 experiments for M/G/1-Aggregate and Ramaswami's formula is within the same magnitude range of 10^{-9}. Observe that Figure 4(a) does present two lines, one for M/G/1-Aggregate and one for Ramaswami's formula but the lines are almost indistinguishable at this level. The proximity of the two solutions is better illustrated in Figure 4(b) where the difference between the solutions obtained by the two different methods is plotted and is in the range of $[0.0, 10^{-16}]$. The two methods are equal for all numerical purposes. Figure 4(c) illustrates the perturbation of solution for both methods with δ_i's in the range of 10^{-12} [9]. Across all experiments, the degree of perturbation in the solution (i.e., the conditioning of the

[8] We conducted a large number of stability experiments but due to space restrictions we only present a few experiments here. We note however that *all* of our experiments did produce consistent results with those presented in this section.

[9] All experiment of this section has been run for different

problem) is within three orders of magnitude less than δ_i. Consistently with Figure 4(b), Figure 4(d) illustrates that the two methods agree to machine precision. Regardless of the magnitude of the input perturbation, the differences between the solutions are *consistently* within the same range, i.e., 10^{-16}.

Next, we turn to a worse conditioned problem, where the elements within the various input matrices vary significantly. We use the same CTMC as the one in the previous set of experiments but the entries in all input matrices vary in magnitude up to 10^{11} with the largest element in the range of 10^2 and the smallest in the range of 10^{-9}. Therefore, by increasing the stiffness of the problem the possibility of numerical error increases. Again, we perturb the input with random values within ranges of 10^{-12} and 10^{-6}. Results are presented in Figure 5. The perturbation of input matrices with values at the level of 10^{-6} introduces a perturbation of the solution in the range of 10^{-7}, higher than the perturbation of solution in the well-conditioned case (compare Figures 5(a) and 4(a)). We point out that there are two lines on top of each-other in Figure 5(a) corresponding to M/G/1-Aggregate and Ramaswami's output respectively. The differences between the solutions obtained by both methods for each experiments are presented in Figure 5(b) and are in the range of $[0.0, 1.8 \times 10^{-14})$. Comparing to the results of the well-conditioned case we note an increase on the difference among the two solutions, but still very small and clearly less than the perturbation value. Figure 5(c) illustrates the perturbation of solutions for perturbation of inputs in the range of 10^{-12}. Comparing to the results of Figure 4, we observe that the conditioning of the problem increases. The degree of perturbation remains constant for all experiments and is one order of magnitude less than δ_i, consistently across experiments. The difference of solutions between the two methods in the case of input perturbation ranges of 10^{-12} is presented in Figure 5(d). The differences are within the same range as for the experiment depicted in Figure 5(b).

The results presented in Figures 4 and 5 show that both methods, M/G/1-Aggregate and Ramaswami's formula behave very similarly under different numerical scenarios. Since for nearly the same input we obtain, in both cases, nearly the same output we argue that the stability of Ramaswami's recursive formula is re-confirmed. Our experiments also illustrate that M/G/1-Aggregate and Ramaswami's recursive formula are in good agreement.

6. CONCLUDING REMARKS

In this paper we presented a new, aggregate approach for the exact solution of M/G/1-type processes. Our exposition focuses on computing efficiently the *exact* probabilities of the boundary states of the process and the aggregate probability of being in each of the classes of states 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 such as the expected queue length or any of its higher moments.

values of δ_i between 10^{-6} and 10^{-12} and the same results are obtained. We choose however to present here the two extreme cases.

We presented detailed analysis of the computation and storage complexity of our method. We conclude that M/G/1-Aggregate requires a few simple steps that provide significant savings with respect to both computation and storage when compared with the traditional matrix analytic. These gains are a direct outcome of the fact that our methodology is based on an aggregate approach and produces such output. The proposed aggregate methodology for solution of M/G/1 processes results in a much simpler, thus easy to implement, algorithm comparing to matrix analytic. The method presented in this paper, along with the classic ones, have been implemented in the MAMSolver tool [16] that is available to the community[10].

An issue that often arises in the area of numerical solutions of Markov chains is the method's numerical stability. The numerical stability of algorithms for the solution of processes that focus on in this paper has hardly been investigated, if at all [6]. In this paper we provide strong experimental indications that our methodology is stable. Examining theoretically the numerical stability of our methodology is subject of future work.

Acknowledgments

Special thanks go to Gianfranco Ciardo for his contributions in earlier stages of this work and to Andreas Stathopoulos who guided us through the numerical stability discussion and analysis presented in this paper. We thank Beatrice Meini for providing us with her implementation of the FFT algorithm for Ramaswami's recursive formula, Guy Latouche for his invaluable feedback, and Mark Squillante for insightful discussions. We also thank the anonymous referees whose comments improved this presentation.

7. REFERENCES

[1] D.A. Bini and B. Meini. Using displacement structure for solving non-skip-free M/G/1 type Markov chains. In A. Alfa and S. Chakravarthy, editors, *Advances in Matrix Analytic Methods for Stochastic Models*, pages 17–37, Notable Publications Inc. NJ, 1998.

[2] D.A. Bini, B. Meini and V. Ramaswami. Analyzing M/G/1 paradigms through QBDs: the role of the block structure in computing the matrix G. In G. Latouche and P. Taylor, editors, *Advances in Algorithmic Methods for Stochastic Models*, pages 73–86, Notable Publications Inc. NJ, 2000.

[3] 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, 1999.

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

[5] D. Heyman and D. Lucantoni. Modeling multiple IP traffic streams with rate limits. In Proceedings of the

[10] `http://www.cs.wm.edu/MAMSolver/`.

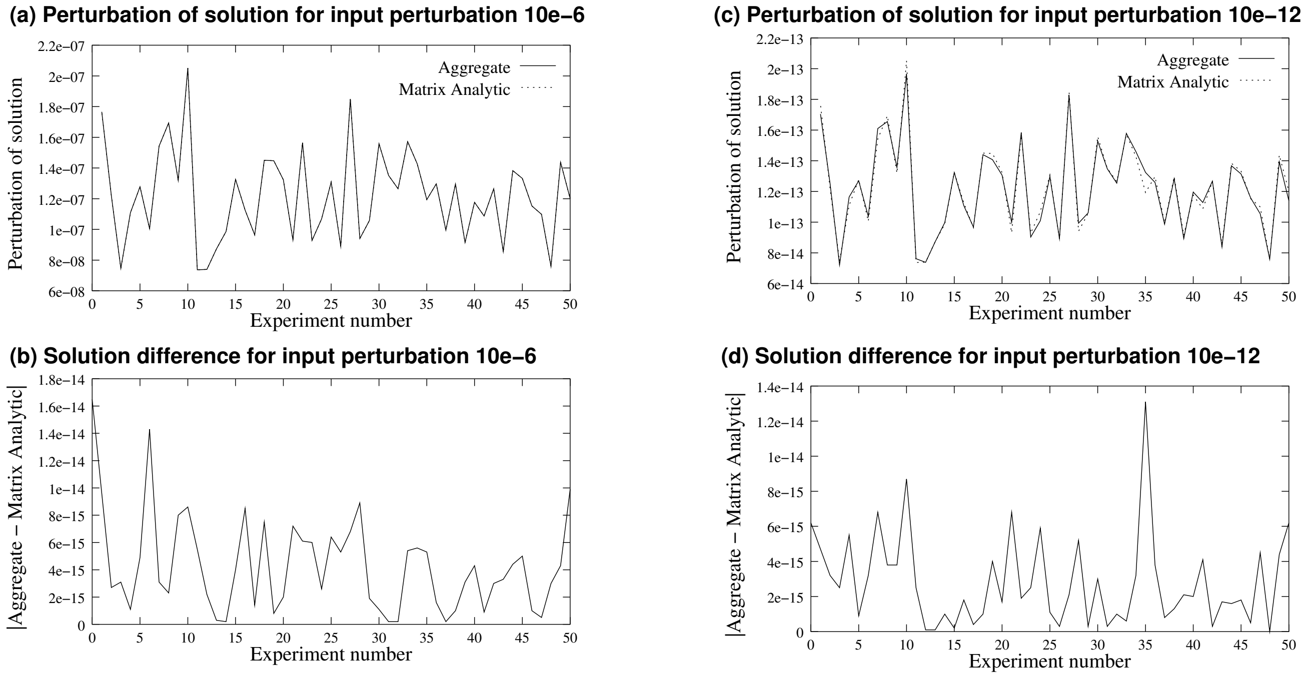

Figure 5: Numerical behavior of the two algorithms with ill-conditioned input. Graphics (a) and (c) plot $\|A(x) - A(x+\delta_i)\|$ for M/G/1-Aggregate and $\|R(x) - R(x+\delta_i)\|$ for Ramaswami's recursive formula for different perturbation magnitudes and for $1 \leq i \leq 50$ distinct experiments. Graphics (b) and (d) illustrate $\|A(x+\delta_i) - R(x+\delta_i)\|$ for the same perturbation magnitudes and the same $1 \leq i \leq 50$ distinct experiments.

17^{th} International Teletraffic Congress, Brazil, Dec. 2001.

[6] 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.

[7] B. Meini. An improved FFT-based version of Ramaswami's formula. *Comm. Statist. Stochastic Models*, vol. 13 pages 223–238, 1997.

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

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

[10] M. F. Neuts. *Matrix-geometric solutions in stochastic models*. Johns Hopkins University Press, Baltimore, MD, 1981.

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

[12] B. F. Nielsen. Modelling long-range dependent and heavy-tailed phenomena by matrix analytic methods. In *Advances in Matrix-Analytic Methods for Stochastic Models, G. Latouche and P. Taylor (eds.)*. Notable Publications, pages 265–278, 2000.

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

[14] V. Ramaswami. A stable recursion for the steady state vector in Markov chains of M/G/1 type. *Commun. Statist.- Stochastic Models*, vol. 4 pages 183-263, 1988.

[15] V. Ramaswami and J.L. Wang. A hybrid analysis/simulation for ATM performance with application to quality-of-service of CBR traffic. *Telecommunication Systems*, vol. 5 pages 25-48, 1996.

[16] A. Riska and E. Smirni. MAMSolver: a Matrix-analytic methods tools, to appear at the 12^{th} *International Conference on Modeling Techniques and Tools for Computer and Communication Systems Performance Evaluation*, London, UK, April 2002.

[17] A. Riska, M. Squillante, S.-Z. Yu, Z. Liu, and L. Zhang, Matrix-analytic analysis of a MAP/PH/1 queue fitted to web server data. to appear at the 4^{th} *Conference on Matrix-Analytic Methods*, Adelaide, Australia, July 2002.

[18] M. S. Squillante. Matrix-analytic methods: Applications, results and software tools. In G. Latouche and P. Taylor, editors, *Advances in Matrix-Analytic Methods for Stochastic Models*, Notable Publications Inc. NJ, 2000.

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