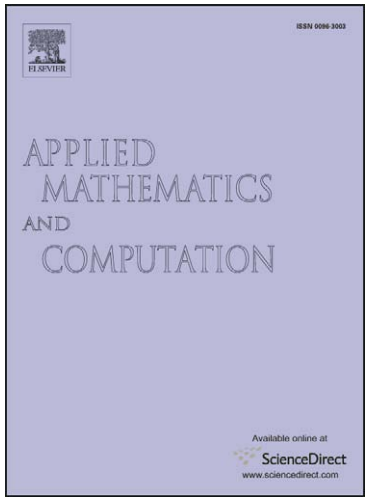

Multigrid Algorithm from Cyclic Reduction for Markovian Queueing Networks

Shu-Ling Yang, Jian-Feng Cai, Hai-Wei Sun

PII: S0096-3003(11)00150-0
DOI: 10.1016/j.amc.2011.01.108
Reference: AMC 15730

To appear in: *Appl. Math. Comput.*

Received Date: 13 July 2009
Revised Date: 19 January 2011
Accepted Date: 30 January 2011

Please cite this article as: S-L. Yang, J-F. Cai, H-W. Sun, Multigrid Algorithm from Cyclic Reduction for Markovian Queueing Networks, *Appl. Math. Comput.* (2011), doi: 10.1016/j.amc.2011.01.108

Multigrid Algorithm from Cyclic Reduction for Markovian Queueing Networks ☆

Shu-Ling Yang[a], Jian-Feng Cai[b], Hai-Wei Sun*[,c]

[a] *School of Applied Mathematics, Guangdong University of Technology, Guangzhou, China.*
[b] *Temasek Laboratories, National University of Singapore, 2 Science Drive 2, Singapore 117543.*
[c] *Department of Mathematics, University of Macau, Macao, China.*

Abstract

A multigrid method based on cyclic reduction strategy is proposed to solve huge, nonsymmetric singular linear systems arising from Markovian queueing networks. A simple way to construct the matrix-dependent prolongation and restriction operators is presented in this paper. Numerical results for multiple queues are given to illustrate the efficiency and robustness of our methods.

Key words: multigrid method, cyclic reduction, Markovian queueing network, nonsymmetric linear system, boundary value method, singular systems.

1. Introduction

Markovian queueing networks have played a very significant role in a number of physical systems [1, 12, 13, 14, 18, 22, 27]. A queueing network is studied under two different situations. One is the steady state for long-run analysis. The other is the transient state for analysis in a finite horizon. The steady state probability distribution of Markovian queueing network can be obtained by solving a nonsymmetric singular linear system [3, 4, 6, 8, 11], while the transient solutions can be obtained by solving a system of ordinary differential equations (ODEs). In this paper, we are interested in finding both the steady state probability distribution vectors and the transient solutions of Markovian queueing networks.

The steady state probability distribution vectors are normalized null-vectors of coefficient matrices which are nonsymmetric and singular with a one-dimensional null space. We note that the size N of the matrices can be huge even with a relatively

☆The research was partially supported by the research grant 033/2009/A from FDCT of Macao, UL020/08-Y2/MAT/JXQ01/FST and RG063/08-09S/SHW/FST from University of Macau.

*Corresponding author
Email address: HSun@umac.mo (Hai-Wei Sun)

small number of queues. For example, for a 5-queue network with 20 waiting spaces and servers per queue, $N = 20^5 = 3,200,000$! In general the null-vector cannot be obtained efficiently by direct methods such as the LU decomposition due to the huge size of the coefficient matrix. Fortunately, the coefficient matrix can be expressed by the tensor sum of a series of tridiagonal matrices plus a low rank sparse residual matrix. Efficient numerical algorithms should make use of the special structures of the matrices and their fast matrix-vector multiplications.

For the transient solutions, we need to solve a system of ODEs. Many classical numerical methods can be employed to solve the ODE systems. A survey on numerical methods for finding transient solutions of homogeneous irreducible Markov chains can be found in [27]. The initial value methods (IVMs) such as the Runge-Kutta methods are good explicit methods for their efficiency and easy implementation. But it may require a small time step to guarantee the convergence. In order to alleviate the problem, one can employ the boundary value method (BVM) to discretize the ODE system [2]. The advantages of using BVM to discretize the Kolmogorov backward equations can be found in [6]. However, the resulting linear system is even much bigger than the one in solving the steady-state probability distribution. With the same strategy in [6], the huge linear system is firstly decomposed into L (the number of time steps) sub-systems with each sub-system having similar structure to the linear system in the steady state case, then each sub-system can be solved one by one.

There are many classical iterative methods discussed for solving the steady state problems in [18]. The preconditioned Krylov subspace methods are introduced in [3, 4, 5]. But the linear convergence of their methods has not been proven. Algebraic multigrid (AMG) methods are also proposed for such problems [6, 8]. However, both the prolongation and restriction operators in [6] are only slightly modified from the classical ones to keep the singularity as well as the structure of the coarse grid matrix, i.e., they are matrix-independent. The AMG method proposed in [8] for the steady state problem does not make use of the special structure of the matrix. The authors just modify the general AMG method [10, 20] to cope with the singular linear systems. The construction of the prolongation and restriction operators in [8] is a little bit complicated and expensive. Furthermore, it remains uncertain if the coarse grid matrices can keep the same structure as the fine grid matrix.

The cyclic reduction (CR) method is a direct solver for the tridiagonal systems which is easier to implement on parallel computers [25]. An AMG method which is based on CR method was proposed by Shapira to solve the partial differential equations (PDEs) with several spatial variables [23, 24, 25]. Unlike the general AMG method in [10, 20], Shapira's AMG algorithm can preserve the same structure for the coarse-grid matrices, e.g., for the 5-point stencil matrix arising from two dimensional PDEs, the coarse grid matrices are also 5-point stencil.

In this paper, we exploit Shapira's strategy to develop a CR-based AMG (CRAMG)

method for solving the problems from Markovian queueing networks. We derive the prolongation and restriction operators from the CR method in a simple and fast way. The coarse grid matrices can preserve the singularity as well as the structure. We remark here that our scheme is different from Shapira's method [23, 24, 25] since the special structure is involved.

The paper is organized as follows. In Section 2, we introduce the Markovian queueing networks. A CRAMG method for the queueing networks is proposed in Section 3. In Section 4, numerical examples are given to demonstrate the efficiency of our method. Finally, concluding remarks are given in Section 5.

2. Linear systems from the Models

For continuous-time Markovian queueing networks, the transient probability distribution can be found by solving Kolmogorov's backward equations, and the steady state probability distribution can be found by solving Kolmogorov's balance equations [1, 3, 4, 5, 11, 22, 27, 28].

Assume that the network has q queues receiving customers from q independent Poisson sources. In the i-th queue there are s_i parallel servers and $n_i - s_i - 1$ waiting spaces. Customers enter the i-th queue with mean arrival rate $\lambda_i > 0$. The service time distribution is independent and exponential with mean rate $\mu_i > 0$. Let $p_{i_1,i_2,\ldots,i_q}$ denote the steady state probability distribution of state $(i_1, i_2, \ldots, i_q)$, i.e., the probability that i_j customers are in the j-th queue. So the total number of states in the system is $N = \prod_{j=1}^{q} n_j$. In the following, I_k denotes the identity matrix of order k and δ_{ij} denotes the Kronecker delta.

2.1. Markovian Queueing Networks

First we consider a simple model, i.e., a two-queue free network. In this network, there are no interactions between the two Markovian $M/M/s_i/(n_i - s_i - 1)$ queues.

If we write a transient state probability distribution at time t as

$$\mathbf{p}(t) = (p_{0,0}(t), \ldots, p_{0,n_2-1}(t), p_{1,0}(t), \ldots, p_{1,n_2-1}(t), \ldots, p_{n_1-1,0}(t), \ldots, p_{n_1-1,n_2-1}(t))^{\intercal},$$

then the Kolmogorov backward equations can be written as

$$\frac{d\mathbf{p}(t)}{dt} = -A_0\mathbf{p}(t), \tag{1}$$

where

$$A_0 = G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 \tag{2}$$

and

$$G_i = \begin{bmatrix} \lambda_i & -\mu_i & & & & & & \\ -\lambda_i & \lambda_i+\mu_i & -2\mu_i & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & -\lambda_i & \lambda_i+s_i\mu_i & -s_i\mu_i & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & -\lambda_i & \lambda_i+s_i\mu_i & -s_i\mu_i & \\ & & & & & -\lambda_i & s_i\mu_i \end{bmatrix}. \tag{3}$$

When the right hand side of (1) reaches $\mathbf{0} = (0, \ldots, 0)^\intercal$, the probability distribution will not change again, i.e., the network is in steady state. Therefore, the steady state probability distribution $\mathbf{p}_s$ satisfies the so called Kolmogorov balance equation

$$A_0 \mathbf{p}_s = \mathbf{0}.$$

Since $\mathbf{p}_s$ is a probability distribution, we require $\mathbf{p}_s \geq 0$ (componentwise) and $\|\mathbf{p}_s\|_1 = 1$. We note that G_i (and hence A_0) has one dimensional null-spaces. Therefore the corresponding null-vector can be chosen to have positive entries and the positive null-vector of A_0 can be normalized to give $\mathbf{p}_s$ [3, 4].

For a general q-queue free model, the coefficient matrices in (1) and (2) become

$$A_0 = \sum_{j=1}^{q} \bigotimes_{i=1}^{q} G_i^{\delta_{ij}}, \tag{4}$$

where G_i are as in (3).

Let $\mathbf{1}_k = (1, \ldots, 1)^\intercal \in \mathbb{R}^k$. Then the singular matrix A_0 satisfies

$$\mathbf{1}_N^\intercal A_0 = \mathbf{0}^\intercal. \tag{5}$$

That means each column's sum of A_0 is zero.

Unlike free networks, overflow networks allow jumping from one queue to another: when one queue is full, customer arriving at that queue is allowed to overflow to another queue if it is not full yet, while overflow in the opposite direction is not allowed. For a general q-queue overflow model, let

$$i_1 \mapsto i_2 \mapsto \cdots \mapsto i_j$$

be the queueing discipline that customers from the i_1-th queue can overflow and be served at the i_j-th queue if the i_1-th, i_2-th, $\cdots$,i_{j-1}-th queues are all full and the i_j-th queue is not full yet. We also assume that there is at most one direction of overflow of customers, i.e.,

$$\cdots \mapsto i \mapsto j \mapsto \cdots \text{and} \cdots \mapsto i \mapsto k \mapsto \cdots \Rightarrow j = k,$$

and any given queueing discipline does not form a loop, i.e.,

$$k \neq l \Rightarrow i_k \neq i_l, 1 \leq k, l \leq j,$$

see [4] for details. Then the transient state probability distribution $\mathbf{p}(t)$ at time t satisfies

$$\frac{d\mathbf{p}(t)}{dt} = -A\mathbf{p}(t), \tag{6}$$

and the steady state probability distribution $\mathbf{p}_s$ satisfies

$$A\mathbf{p}_s = \mathbf{0}, \tag{7}$$

where $A = A_0 + R_0$, A_0 is as in (4),

$$R_0 = \bigotimes_{k=1}^{q} E_k^{\delta_{i_1 k}} \cdots E_k^{\delta_{i_{j-1} k}} (R_k^{i_{j-1}})^{\delta_{i_j k}},$$

E_k=$\mathbf{e}_{n_k}\mathbf{e}_{n_k}^\intercal$, $\mathbf{e}_{n_k}$ is the n_k-th unit vector in $\mathbb{R}^{n_k}$, and

$$R_k^{i_{j-1}} = \lambda_{i_{j-1}} \begin{bmatrix} 1 & & & & \\ -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & -1 & 1 & \\ & & & -1 & 0 \end{bmatrix}$$

is a square matrix of order n_k. Particularly, $R_0 = \mathbf{e}_{n_1}\mathbf{e}_{n_1}^\intercal \otimes R_2^1$ for a two-queue model. We note that the singularity property of A still holds as in (5):

$$\mathbf{1}_N^\intercal A = \mathbf{0}^\intercal \tag{8}$$

2.2. Generating the Linear Systems

We are interested in getting both the steady and transient state probability distribution of q-queue overflow networks. We note that the coefficient matrix A in (7) is singular with a one dimensional null-space and a positive null-vector. Therefore there exists a unique solution of (7), see [4, 8] for details. Although (7) is a singular linear system, we can transform it into a nonsingular one by the trick introduced in [4].

We consider the perturbed version of (7):

$$\tilde{A}\tilde{\mathbf{p}}_s \triangleq (A + \lambda_1 \hat{\mathbf{e}}_{n_1}\hat{\mathbf{e}}_{n_1}^\intercal)\tilde{\mathbf{p}}_s = \hat{\mathbf{e}}_{n_1}, \tag{9}$$

where $\hat{\mathbf{e}}_{n_1}$ is the n_1-th unit vector in $\mathbb{R}^N$. We note that $\tilde{A}$ is weakly diagonally dominant (column-wise) and irreducible. Furthermore the sum of the n_1-th column is strictly

positive. Therefore $\tilde{A}$ is nonsingular, hence there exists a unique solution $\tilde{\mathbf{p}}_s$ of (9). Left multiplying both sides of (9) by $\mathbf{1}_N^\intercal$, according to (8), we have

$$(\tilde{\mathbf{p}}_s)_{n_1} = 1/\lambda_1 \neq 0,$$

where $(\tilde{\mathbf{p}}_s)_{n_1}$ is the n_1-th entry of $\tilde{\mathbf{p}}_s$. Hence

$$A\tilde{\mathbf{p}}_s = \hat{\mathbf{e}}_{n_1} - \lambda_1 \hat{\mathbf{e}}_{n_1}\hat{\mathbf{e}}_{n_1}^\intercal \tilde{\mathbf{p}}_s = \mathbf{0}.$$

Therefore, we can get the steady state probability distribution $\mathbf{p}_s$ of (7) by

$$\mathbf{p}_s = \frac{\tilde{\mathbf{p}}_s}{\|\tilde{\mathbf{p}}_s\|_1}.$$

For the transient solutions, we need to solve the system of ODEs (6). In this paper, we follow the idea in [6] by employing a BVM to discretize (6). The BVMs are implicit stable methods and hence there is no restriction on the size of the time step in order to make them converge. The disadvantage is that they require solutions of large linear systems. An example of BVM is the third order generalized backward differentiation formula (GBDF3). We will use it to illustrate how to use BVM to discretize (6). Suppose we want to find $\mathbf{p}(T)$ for some final time T. Then we divide the horizon into L steps, with step size $h = T/L$. Denote $\mathbf{p}_k = \mathbf{p}(kh)$, $0 \leq k \leq L$, the probability distribution that we want to find. For the ODEs (6),

$$\frac{d\mathbf{p}(t)}{dt} = -A\mathbf{p}(t),$$

the GBDF3 can be written as

$$\begin{array}{lcll} \frac{1}{6}(2\mathbf{p}_{k+1} + 3\mathbf{p}_k - 6\mathbf{p}_{k-1} + \mathbf{p}_{k-2}) & = & hA\mathbf{p}_k, & k = 2, 3, \ldots, L-1 \\ \frac{1}{6}(-\mathbf{p}_3 + 6\mathbf{p}_2 - 3\mathbf{p}_1 - 2\mathbf{p}_0) & = & hA\mathbf{p}_1, & \\ \frac{1}{6}(11\mathbf{p}_L - 18\mathbf{p}_{L-1} + 9\mathbf{p}_{L-2} - 2\mathbf{p}_{L-3}) & = & hA\mathbf{p}_L, & \end{array}$$

see [2] for details. In matrix form, the above equation becomes

$$[H \otimes I_N + hI_L \otimes A]\bar{\mathbf{p}} = -\mathbf{a} \otimes \mathbf{p}_0, \tag{10}$$

where $\mathbf{a}^\intercal = (-\frac{1}{3}, \frac{1}{6}, 0, \ldots, 0) \in \mathbb{R}^L$, $\bar{\mathbf{p}}^\intercal = (\mathbf{p}_1^\intercal, \ldots, \mathbf{p}_L^\intercal)$, and H is an L-by-L matrix

$$H = \begin{bmatrix} -\frac{1}{2} & 1 & -\frac{1}{6} & & & & \\ -1 & \frac{1}{2} & \frac{1}{3} & & & & \\ \frac{1}{6} & -1 & \frac{1}{2} & \frac{1}{3} & & & \\ & \frac{1}{6} & -1 & \frac{1}{2} & \frac{1}{3} & & \\ & & \ddots & \ddots & \ddots & \ddots & \\ & & & \frac{1}{6} & -1 & \frac{1}{2} & \frac{1}{3} \\ & & & -\frac{1}{3} & \frac{3}{2} & -3 & \frac{11}{6} \end{bmatrix}.$$

Let $H = P^{-1}DP$ be the spectral decomposition of H with D being a diagonal matrix with diagonal entries ρ_i, $1 \le i \le L$. Note that the eigenvalues of H are complex numbers with positive real parts [2].

Let $\mathbf{q} = (P \otimes I_N)\bar{\mathbf{p}}$. Then (10) becomes

$$(D \otimes I_N + hI_L \otimes A)\mathbf{q} = -(P\mathbf{a}) \otimes \mathbf{p}_0. \tag{11}$$

We decompose this system of equations into L sub-systems of smaller size:

$$(\rho_i I_N + hA)\mathbf{q}_i = c_i \mathbf{p}_0. \tag{12}$$

where c_i is the i-th entry of $-P\mathbf{a}$ and $\mathbf{q}^\intercal = (\mathbf{q}_1^\intercal, \mathbf{q}_2^\intercal, \ldots, \mathbf{q}_L^\intercal)$. That means we can carry out a parallel computing for (11) by (12). In the next section, we study how to solve (9) and (12) by an AMG method.

3. Multigrid Algorithm

3.1. V-Cycle Multigrid Method

For a general linear system

$$A^h \mathbf{u}^h = \mathbf{f}^h, \tag{13}$$

the V-cycle multigrid can be described as the algorithm below [21], where we have to provide the smoothing operator `smooth`, the prolongation operator I_H^h, the restriction operator I_h^H and the coarse grid operators A^H.

Algorithm 1. $\mathbf{u}^h$ =V-cycle$(A^h, \mathbf{u}_0^h, \mathbf{f}^h)$

1.	Pre-smooth:	$\mathbf{u}^h := \texttt{smooth}^{v_1}(A^h, \mathbf{u}_0^h, \mathbf{f}^h)$
2.	Get residual:	$\mathbf{r}^h = \mathbf{f}^h - A^h\mathbf{u}^h$
3.	Coarsen:	$\mathbf{r}^H = I_h^H \mathbf{r}^h$
4.	If $(H == h_0)$	
5.	Solve:	$A^H \xi^H = \mathbf{r}^H$
6.	Else	
7.	Recursion:	ξ^H=V-cycle$(A^H, \mathbf{0}, \mathbf{r}^H)$
8.	Endif	
9.	Correct:	$\mathbf{u}^h := \mathbf{u}^h + I_H^h \xi^H$
10.	Post-smooth:	$\mathbf{u}^h := \texttt{smooth}^{v_2}(A^h, \mathbf{u}^h, \mathbf{f}^h)$
11.	Return $\mathbf{u}^h$	

The coarsening process is performed in order to ensure that the range of interpolation approximates the errors which are not sufficiently reduced via smoothing. In geometric multigrid (GMG) method, the prolongation and restriction operators are

fixed, while the coarse matrix is obtained by discretizing the original problem on the coarse grid. Therefore, generally the GMG requires that the problem to be solved has a geometric background. However, there are many problems which do not have any geometric background or GMG method has a poor performance on them. In order to broaden the application area of GMG, AMG methods are developed to utilize the principle of the geometry oriented GMG. In AMG methods, the restriction operators and prolongation operators are defined in a pure algebraic way, i.e., only from the knowledge of the matrix. The coarse matrix is typically defined by using the Galerkin approach, i.e.

$$A^H = I_h^H A^h I_H^h,$$

see [15, 20, 21] for details.

Our problems (6) and (7) do not have any geometry background. Therefore, the GMG is not applicable and we should turn to the AMG method. Although the AMG methods have already been proposed to solve the steady and transient problems in [8, 6] respectively, both of them have not made use of the special structure of the coefficient matrices. In [8], the authors just modify the general AMG method to cope with the singular linear systems to get the steady state solution. The construction of the prolongation and restriction operators is complicated and expensive: it requires separating grid points into two sets according to the matrix graphs of the coefficient matrices on each level. The formula for the prolongation operator is also very complicated. In [6], the authors do not construct matrix-dependent prolongation and restriction operators for the transient problems. They use the classical prolongation and restriction operators for different queueing networks.

3.2. Cyclic Reduction

We present our method for constructing the prolongation and restriction operators by describing the CR method (see [25]). We firstly assume that A^h in (13) is an n-by-n tridiagonal matrix with nonzero diagonal entries, where n is an odd number,

$$A^h = \begin{bmatrix} b_1 & c_2 & & & \\ a_1 & b_2 & c_3 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-2} & b_{n-1} & c_n \\ & & & a_{n-1} & b_n \end{bmatrix}. \tag{14}$$

We permute the matrix (14) based on an odd-even structure,

$$
\begin{aligned}
QA^hQ^\intercal &= \begin{bmatrix}
b_2 & & & & a_1 & c_3 & & & \\
& b_4 & & & & a_3 & c_5 & & \\
& & \ddots & & & & \ddots & \ddots & \\
& & & b_{n-1} & & & & a_{n-2} & c_n \\
c_2 & & & & b_1 & & & & \\
a_2 & c_4 & & & & b_3 & & & \\
& a_4 & \ddots & & & & \ddots & & \\
& & \ddots & c_{n-1} & & & & b_{n-2} & \\
& & & a_{n-1} & & & & & b_n
\end{bmatrix} \\
&\triangleq \begin{bmatrix} D_1 & C_1 \\ B_1 & D_0 \end{bmatrix}.
\end{aligned} \tag{15}
$$

Decomposing $QA^hQ^\intercal$ into block LDU form, we obtain

$$
A^h = Q^\intercal \begin{bmatrix} I & \\ B & I \end{bmatrix} \begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix} \begin{bmatrix} I & C \\ & I \end{bmatrix} Q,
$$

where $B = B_1 D_1^{-1}$, $C = D_1^{-1} C_1$ and

$$
D_2 = D_0 - B D_1 C. \tag{16}
$$

Now we consider how to solve (13) by using Gaussian Elimination (GE). We denote $\mathbf{u}^h = Q^\intercal \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix}$, and $\mathbf{f}^h = Q^\intercal \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix}$. Then, we are going to solve

$$
\begin{bmatrix} I & \\ B & I \end{bmatrix} \begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix} \begin{bmatrix} I & C \\ & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix}.
$$

By forward substitution, we obtain two vectors $\mathbf{y}_1$, $\mathbf{y}_2$ which satisfy

$$
\begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix} \begin{bmatrix} I & C \\ & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 - B\mathbf{f}_1 \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix}.
$$

Now we need to solve two smaller systems of linear equations of size about $\frac{n}{2}$, and denote their solutions by $\mathbf{z}_1$ and $\mathbf{z}_2$ respectively,

$$
\begin{bmatrix} I & C \\ & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} = \begin{bmatrix} D_1^{-1}\mathbf{y}_1 \\ D_2^{-1}\mathbf{y}_2 \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix}.
$$

After backward substitution and permutation, we get the solution

$$
\mathbf{u}^h = Q^\intercal \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix} = Q^\intercal \begin{bmatrix} \mathbf{z}_1 - C\mathbf{z}_2 \\ \mathbf{z}_2 \end{bmatrix}.
$$

Since D_1 is diagonal, it is easy to get the vector $\mathbf{z}_1$. Noting that D_2 is also a tridiagonal matrix of size about $\frac{n}{2}$, we can get the solution $\mathbf{z}_2$ recursively. The above method, which applies the GE method to odd-even ordered linear system and solves D_1 by direct method and D_2 by recursion, is also called cyclic reduction.

In the multigrid V-cycle, after the pre-smoothing, we get an approximated solution $\mathbf{u}^h$ and the corresponding residual $\mathbf{r}^h = Q^\intercal \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \end{bmatrix}$. We then correct $\mathbf{u}^h$ by the solution obtained from the coarse grid. Now we consider GE method to get the correction, i.e., we use an approximated solution $\mathbf{x}$ of $A^h\mathbf{x} \approx \mathbf{r}^h$ as the correction, and compare it with the coarse grid correction in the V-cycle.

Since a good smoother can reduce the residual much for the unknowns of the fine grids, we can assume $\mathbf{r}_1 \approx \mathbf{0}$, if we choose even grids as fine grids. Therefore, approximately we set $\mathbf{r}_1 = \mathbf{0}$ hence $\mathbf{z}_1 = \mathbf{0}$, and apply the GE method. Thus, the GE method now becomes the following. First we get $\mathbf{y}_2 = \mathbf{r}_2 - B\mathbf{r}_1$, and then we use recursion to solve $D_2\mathbf{z}_2 = \mathbf{y}_2$, and finally we get the correction by $\mathbf{x} = Q^\intercal \begin{bmatrix} -C\mathbf{z}_2 \\ \mathbf{z}_2 \end{bmatrix}$. Comparing it with the coarse grid correction in the multigrid V-cycle, we find that the above procedure is essentially a coarse grid correction in the AMG method by choosing the restriction operator

$$I_h^H = [\,-B \quad I\,]\,Q = \begin{bmatrix} 1 & -\frac{c_2}{b_2} & & & & \\ & -\frac{a_2}{b_2} & 1 & -\frac{c_4}{b_4} & & \\ & & & \cdots & \cdots & \cdots \\ & & & & & -\frac{a_{n-1}}{b_{n-1}} \quad 1 \end{bmatrix} \tag{17}$$

and the prolongation operator

$$I_H^h = Q^\intercal \begin{bmatrix} -C \\ I \end{bmatrix} = \begin{bmatrix} 1 & & & \\ -\frac{a_1}{b_2} & -\frac{c_3}{b_2} & & \\ & 1 & & \\ & -\frac{a_3}{b_4} & \vdots & \\ & & \vdots & \\ & & \vdots & -\frac{c_n}{b_{n-1}} \\ & & & 1 \end{bmatrix}, \tag{18}$$

and using Galerkin approach to get the coarse-grid matrix by (16)

$$A^H = I_h^H A^h I_H^h = D_2. \tag{19}$$

Hence, from the view of CR method, we have derived the restriction and prolongation operators for an AMG method. We remark that if we choose red-black Gauss-Seidel iteration as a pre-smoother, then $\mathbf{r_1} = \mathbf{0}$. Therefore, this AMG method is a direct solver

for one-dimensional problem. In the following, we will derive a CRAMG algorithm for solving the problems from multi-dimensional queueing networks.

3.3. CRAMG Method

Now we consider the multi-dimensional problems. Without loss of generality and for simplicity, we study the two-dimensional problem to illustrate the construction of the restriction and prolongation operators. In this case, the finest grid operator is

$$A^h = A_1 \otimes I_{n_2} + I_{n_1} \otimes A_2,$$

where both A_1 and A_2 are tridiagonal matrices. We simply choose $I_h^H(A_1) \otimes I_h^H(A_2)$ and $I_H^h(A_1) \otimes I_H^h(A_2)$ as the restriction and prolongation operators respectively, where $I_H^h(A_i)$ and $I_h^H(A_i)$ denote the corresponding tridiagonal matrices derived from A_i. The coarser grid matrix now becomes $A_1^H \otimes T_2 + T_1 \otimes A_2^H$, where T_1 and T_2 are both tridiagonal matrices. The restriction operator for the coarser level operator will be $I_h^H(A_1^H) \otimes I_h^H(A_2^H)$ and the prolongation operator will be $I_H^h(A_1^H) \otimes I_H^h(A_2^H)$, respectively. We process it until the coarsest level is reached.

For the q-dimensional problems,

$$A^h = \sum_{j=1}^{q} \bigotimes_{i=1}^{q} A_i^{\delta_{ij}}, \tag{20}$$

where A_i are tridiagonal matrices, $i = 1, \ldots, q$. The restriction operator for the coarser level operator will be

$$I_h^H = \bigotimes_{i=1}^{q} I_h^H(A_i) = I_h^H(A_1) \otimes \cdots \otimes I_h^H(A_q), \tag{21}$$

and the prolongation operator will be

$$I_H^h = \bigotimes_{i=1}^{q} I_H^h(A_i) = I_H^h(A_1) \otimes \cdots \otimes I_H^h(A_q), \tag{22}$$

respectively. Obviously, the cost of constructing the prolongation and restriction operators and the coarse grid matrix is only $\mathcal{O}(N)$, where N is the matrix size. Therefore, it can be easily implemented.

If A^h is singular with one dimensional null-spaces, it will be better if we keep the singularity in the coarse grid operators for fast convergence [8, 9]. In the following, we will discuss this issue.

Lemma 1. *Assume that A^h is an n-by-n singular tridiagonal matrix and its column's sums are zeros as in* (5)*,*

$$\mathbf{1}_n^\intercal A^h = \mathbf{0}^\intercal. \tag{23}$$

Then the restriction operator I_h^H of A^h in (17) *satisfies*

$$\mathbf{1}_{n^H}^\intercal I_h^H = \mathbf{1}_n^\intercal, \tag{24}$$

where $n^H \approx n/2$.

Proof: According to (23) and (15), we have

$$\mathbf{1}_n^\intercal Q A^h Q^\intercal = \mathbf{1}_n^\intercal \begin{bmatrix} D_1 & C_1 \\ B_1 & D_0 \end{bmatrix} = \mathbf{0}^\intercal.$$

It follows that $\mathbf{1}_{n^H}^\intercal D_1 = -\mathbf{1}_{n^H}^\intercal B_1$, or $\mathbf{1}_{n^H}^\intercal = -\mathbf{1}_{n^H}^\intercal B_1 D_1^{-1} = -\mathbf{1}_{n^H}^\intercal B$. Then the formula (24) is resulted from (17). □

Lemma 1 can be extended to the multi-dimensional problems. It is easy to verify the follow lemma.

Lemma 2. *Let $A^h = \sum_{j=1}^q \bigotimes_{i=1}^q A_i^{\delta_{ij}}$ be singular and satisfy $\mathbf{1}_N^\intercal A^h = \mathbf{0}^\intercal$. Then the restriction operator I_h^H of A^h satisfies $\mathbf{1}_{N^H}^\intercal I_h^H = \mathbf{1}_N^\intercal$, where $N^H \approx N/2^q$.*

With Lemma 2 , we can get the following theorem to guarantee the singularity of the coarse matrices.

Theorem 1. *If $A^h = \sum_{j=1}^q \bigotimes_{i=1}^q A_i^{\delta_{ij}}$ is singular and satisfies $\mathbf{1}_N^\intercal A^h = \mathbf{0}^\intercal$, then*

$$\mathbf{1}_{N^H}^\intercal A^H = \mathbf{0}^\intercal.$$

Proof: According to Lemma 2, we have

$$\mathbf{1}_{N^H}^\intercal A^H = \mathbf{1}_{N^H}^\intercal I_h^H A^h I_H^h = \mathbf{1}_N^\intercal A^h I_H^h = \mathbf{0}^\intercal.$$

□

Theorem 1 shows that our CRAMG can preserve the singularity in the coarse grid matrices.

For the multi-dimensional networks, we will use the steady state solution of 2-queue overflow network as an example to illustrate the construction of the restriction and prolongation operators. In this case the coefficient matrix in (9) is

$$G_1 \otimes I_{n_2} + I_{n_1} \otimes G_2 + \mathbf{e}_{n_1}\mathbf{e}_{n_1}^\intercal \otimes R_1 + \lambda_1 \hat{\mathbf{e}}_{n_1}\hat{\mathbf{e}}_{n_1}^\intercal.$$

We ignore the last two terms and choose $I_h^H(G_1) \otimes I_h^H(G_2)$ and $I_H^h(G_1) \otimes I_H^h(G_2)$ as the restriction and prolongation operators respectively.

From the above procedure, we see that the matrices at each level are at most 3^q bands since they can all be written in tensor products of tridiagonal matrices. Then the cost of each V-cycle iteration is $\mathcal{O}(N)$ for the steady solution, and $\mathcal{O}(LN)$ for the transient solution.

Parameters: $s_1 = s_2, \lambda_1 = \lambda_2 = 1, \mu_1 = \mu_2 = 2$							
$s_1 = s_2 = 1$				$s_1 = s_2 = 2$			
PCGS		CRAMG		PCGS		CRAMG	
(n_1, n_2)	Iter	(n_1, n_2)	Iter	(n_1, n_2)	Iter	(n_1, n_2)	Iter
(32,32)	8	(33,33)	7	(32,32)	10	(33,33)	6
(64,32)	10	(65,33)	7	(64,32)	12	(65,33)	6
(128,32)	12	(129,33)	7	(128,32)	14	(129,33)	6
(256,32)	14	(257,33)	7	(256,32)	16	(257,33)	6
(512,32)	14	(513,33)	7	(512,32)	16	(513,33)	6

Table 1: Comparison of the Numbers of Iterations for Convergence with PCGS

4. Numerical Experiments

Throughout this section, we only consider the overflow networks, which are more complicated than the free networks and it is harder to get their solutions. We will illustrate the efficiency of our CRAMG method for both the steady state and the transient state probability distributions of the overflow networks. In the CRAMG method, we perform one pre-smooth operator and one post-smooth operator with Gauss-Seidel iteration as the smoother. At each level, the size of the matrix is $2^M + 1$, where M is the level. We set the coarsest grid level to be 1, i.e., the matrix on the coarsest grid is 3-by-3.

In the first part of this section, we focus on the steady state probability distribution of overflow networks. We first compare our CRAMG method with the circulant preconditioned conjugate gradient squared (PCGS) method proposed in [5]. Table 1 gives the results of our method compared with the results for PCGS are copied from [5], and the stopping criteria is the same as that in [5]. We see that the number of iterations for convergence of the CRAMG method is less than that of PCGS. We emphasize that the computational cost per iteration of the PCGS is $\mathcal{O}(n_1 n_2 \log n_1)$, while the CRAMG method is only $\mathcal{O}(n_1 n_2)$. Hence the CRAMG method is more efficient than the PCGS.

In Table 2, we compare our CRAMG method with the old AMG (OAMG) method proposed in [8]. As before, the results for the OAMG method are copied from [8], and the stopping criteria is the same as that in [8]. We see from the table that the number of iterations for convergence of the CRAMG method is slightly smaller than that of the OAMG method. Moreover, the iteration number of the OAMG method increases as the size of the queue increases, while the iteration number of our CRAMG method keeps as a constant. Furthermore, in the OAMG method, the construction of the projection operators is very complicated. The procedure can be done in $\mathcal{O}(N)$

Parameters: $s_1 = s_2, \lambda_1 = \lambda_2 = 1, \mu_1 = \mu_2 = 1$							
$s_1 = s_2 = 3$				$s_1 = s_2 = 5$			
OAMG		CRAMG		OAMG		CRAMG	
(n_1, n_2)	Iter	(n_1, n_2)	Iter	(n_1, n_2)	Iter	(n_1, n_2)	Iter
(8,8)	4	(9,9)	5	(8,8)	4	(9,9)	5
(16,16)	5	(17,17)	5	(16,16)	5	(17,17)	5
(32,32)	5	(33,33)	5	(32,32)	5	(33,33)	5
(64,64)	6	(65,65)	5	(64,64)	6	(65,65)	5
(128,128)	6	(129,129)	5	(128,128)	6	(129,129)	5

Table 2: Comparison of the Numbers of Iterations for Convergence with the AMG Method in [8]

(n_1, n_2)	Ex.1	Ex.2	Ex.3	Ex.4	Ex.5
(9,9)	7	7	7	5	5
(17,17)	7	7	7	5	5
(33,33)	7	7	7	5	5
(65,65)	7	7	7	5	5
(129,129)	7	7	7	5	5

Table 3: Numbers of Iterations for Convergence for 2-queue Overflow Networks

flops, but it requires a very careful programming, see [20]. Obviously, the construction of the operators in the CRAMG is also $\mathcal{O}(N)$ flops, and it can be easily implemented. Therefore, the CRAMG method is superior to the OAMG method at this point.

From Table 3 to Table 5, we apply our CRAMG method to overflow networks with 2, 3 and more queues. The test examples are:

Ex.1: $n_i = 2^M + 1, \mu_i = 1, s_i = 5, \lambda_i = s_i\mu_i - 0.5n_i^{-1}, i = 1, 2, \cdots, q$
Ex.2: $n_i = 2^M + 1, \mu_i = 2^{M-1}, s_i = 1, \lambda_i = 2^{M-1}, i = 1, 2, \cdots, q$
Ex.3: $n_i = 2^M + 1, \mu_i = 2^{M-1}, s_i = 5, \lambda_i = s_i\mu - 0.5n_i^{-1}, i = 1, 2, \cdots, q$
Ex.4: $n_i = 2^M + 1, \mu_i = 2^{M-4}, s_i = 5, \lambda_i = 2^{M-1}, i = 1, 2, \cdots, q$
Ex.5: $n_i = 2^M + 1, \mu_i = 1, s_i = 5, \lambda_i = 1, i = 1, 2, \cdots, q$

The stopping criteria is

$$\|\mathbf{r}^k\|/\|\mathbf{b}\| < 10^{-6}, \tag{25}$$

where $\mathbf{r}^k$ is the residual at the k-th iteration and $\mathbf{b}$ is the right hand side of the linear equation to be solved. From these tables, we see that the CRAMG method is very efficient even for networks consisting of 5 queues.

In the following numerical experiments, we focus on the transient solutions of the 2-queue networks. In our experiments, we are only interested in solving the huge linear

(n_1, n_2, n_3)	Ex.1	Ex.2	Ex.3	Ex.4	Ex.5
(9,9,9)	9	8	9	5	5
(17,17,17)	9	8	9	5	5
(33,33,33)	8	8	8	5	5
(65,65,65)	8	8	8	5	5

Table 4: Numbers of Iterations for Convergence for 3-queue Networks with Overflow Discipline $1 \mapsto 2 \mapsto 3$

n_i	q	Ex.1	Ex.2	Ex.3	Ex.4	Ex.5
9	2	7	7	7	5	5
	3	9	8	9	5	5
	4	10	9	10	5	6
	5	11	10	11	5	7
17	2	7	7	7	5	5
	3	9	8	9	5	6
	4	10	9	10	5	6

Table 5: Numbers of Iterations for Convergence for q-queue Networks with Overflow Discipline $1 \mapsto 2 \mapsto \cdots \mapsto q$

systems after BVM discretization. In all examples, we aim to find the distribution vector at $T = 10$. The step size h is fixed as 1. We assume that the initial state is $(0, 0)$. In the V-cycle, we use a stopping tolerance 10^{-6} as in (25).

We report the maximum numbers of V-cycle required for convergence for each subsystem in Table 6, where "C" denotes the classical AMG (CAMG) method proposed in [6], "CR" denotes our CRAMG method and "**" means that it does not converge after 300 iterations. The testing examples are the same as in the steady state case.

In Table 6, we also compare the preconditioned GMRES method with the CRAMG method. We will use T. Chan's preconditioner [5], which is denoted by "P" in the table. If $c(G_i)$ are T. Chan's preconditioner for G_i, then the preconditioner for (12) is

$$\rho_i(I_{n_1} \otimes I_{n_2}) + h(c(G_1) \otimes I_{n_2} + I_{n_1} \otimes c(G_2)).$$

From Table 6, we see that AMG methods need fewer iterations for convergence. Furthermore, the iteration number required for convergence of the preconditioned GMRES increases as the queue size increases in example 2, 3, 4 and 5. We should also note that the cost per iteration of the preconditioned GMRES is $\mathcal{O}(n_1 n_2 \log n_1 n_2)$, and it is more costly than the AMG methods. Among the two AMG methods, the CRAMG is more efficient and robust than the CAMG. The iteration numbers of V-

	Ex. 1			Ex. 2			Ex. 3			Ex. 4			Ex. 5		
M	C	CR	P	C	CR	P	C	CR	P	C	CR	P	C	CR	P
3	6	6	23	5	5	18	6	7	24	**	3	60	11	4	39
4	6	6	24	5	5	23	7	7	31	**	3	140	55	4	79
5	6	6	24	5	5	28	7	7	37	**	3	**	**	4	152
6	6	6	23	5	5	34	7	7	45	**	3	**	**	4	265

Table 6: Maximum numbers of iterations for Convergence for 2-queue networks

cycle for convergence of the CRAMG method are almost the same as those of the CAMG in example 1–3. In example 4 and 5, the CAMG method does not converge, but our CRAMG method works well. We also see that the iteration number of V-cycles required for convergence is independent of the queue size even when $\|A\|$ increases.

Comparing the iterative numbers of different state solutions in the same network, we find that the numbers of iterations for solving the transient probability distributions are smaller than those in the steady-state case. The reason is that (12) is in a better condition than (9) since there is a $\rho_i I_N$ term in (12).

5. Concluding Remarks

In this paper, we propose a CRAMG method to solve the huge nonsymmetric singular linear systems arising from the steady and transient state probability distributions of Markovian queueing networks. Numerical results show that our CRAMG method is better than other AMG methods in [6, 8] and preconditioned Krylov subspace method in [5]. Convergence analysis will be studied in the future.

Acknowledgements

We thank Prof. Raymond H. Chan for his valuable comments and suggestions. This paper will not be possible without his help. The authors are also thankful for the referee's helpful comments and suggestions.

References

[1] A. Benoit, L. Brenner, P. Fernandes, B. Plateau, Aggregation of stochastic automata networks with replicas, *Linear Algebra and its Applications* 386 (2004) 111–136.

[2] L. Brugnano, D. Trigiante, *Solving Differential Problems by Multistep Initial and Boundary Value Methods*, Gordon and Breach Science Publishers, 1998.

[3] R. H. Chan, Iterative methods for overflow queueing models I, *Numerische Mathematik* 51 (1987) 143–180.

[4] R. H. Chan, Iterative methods for overflow queueing models II, *Numerische Mathematik* 54 (1988) 57–78.

[5] R. H. Chan, W. K. Ching, Circulant preconditioners for stochastic automata networks, *Numerische Mathematik* 87 (2000) 35–57.

[6] R. H. Chan, K. C. Ma, W. K. Ching, Boundary value methods for solving transient solutions of Markovian queueing networks, *Applied Mathematics and Computation* 172 (2006) 690–700.

[7] Q. S. Chang, Z. H. Huang, Efficient algebraic multigrid algorithms and their convergence, *SIAM Journal on Scientific Computing* 24 (2002) 597–618.

[8] Q. S. Chang, S. Q. Ma, G. Y. Lei, Algebraic multigrid method for queueing networks, *International Journal of Computer Mathematics* 70 (1999) 539–552.

[9] Q. S. Chang, W. W. Sun, On convergence of multigrid method for nonnegative definite systems, *Journal of Computational Mathematics* 23 (2005) 177–184.

[10] Q. S. Chang, Y. S. Wong, H. Q. Fu, On the algebraic multigrid method, *Journal of Computational Physics* 125 (1996) 279–292.

[11] W. K. Ching, *Iterative Methods for Queueing and Manufacturing Systems*, Springer Monographs in Mathematics, Springer, London, 2001.

[12] W. K. Ching, R. H. Chan, X. Y. Zhou, Circulant preconditioner for Markov modulated Possion processes and their applications to manufacturing systems, *SIAM Journal on Matrix Analysis and Applications* 18 (1997) 464–481.

[13] W. K. Ching, A. W. Loh, Iterative methods for flexible manufacturing systems, *Applied Mathematics and Computation* 141 (2003) 553–564.

[14] W. K. Ching, W. O. Yuen, A. W. Loh, An inventory model with returns and lateral transshipments, *Journal of the Operational Research Society* 54 (2003) 636–641.

[15] W. Hackbusch, *Multi-Grid Methods and Applications*, Springer-Verlag, Berlin, 1985.

[16] G. Horton, G. T. Leutenegger, A multi-level solution algorithm for steady-state Markov chains, *ACM SIGMETRICS Performance Evaluation Review* 22 (1994) 191–200.

[17] Z. H. Huang, Q. S. Chang, Gauss-Seidel-type multigrid methods, *Journal of Computational Mathematics* 21 (2003) 421–434.

[18] L. Kaufman, Matrix methods for queueing problems, *Journal on Scientific and Statistical Computing* 4 (1983) 525–552.

[19] A. Reusken, On the approximate cyclic reduction preconditioner, *SIAM Journal on Scientific Computing* 21 (1999) 565–590.

[20] J. Ruge, K. Stüben, Algebraic multigrid, in *Multigrid Methods* (S. McCormick, Eds.), Frontiers in Applied Mathematics, No. 3 73–130, SIAM, Philadelphia, 1986.

[21] Y. Saad, *Iterative Methods for Sparse Linear Systems* (2nd edn), SIAM, Philadelphia, 2003.

[22] I. Sbeity, L. Brenner, B. Plateau, W. J. Stewart, Phase-type distributions in stochastic automata networks, *European Journal of Operational Research* 186 (2008) 1008–1028.

[23] Y. Shapira, Analysis of matrix-dependent multigrid algorithms. *Numerical Linear Algebra with Applications* 5 (1998) 165–201.

[24] Y. Shapira, Model case analysis of an algebraic multilevel method, *Numerical Linear Algebra with Applications* 6 (1999) 655–685.

[25] Y. Shapira, *Matrix-based multigrid*, Theory and applications, Numerical Methods and Algorithms 2, Kluwer Academic Publishers, Boston, 2003.

[26] Y. Y. Shi, Q. S. Chang, Remark on convergence of algebraic multigrid in the form of matrix decomposition, *Applied Mathematics and Computation* 170 (2005) 1170–1184.

[27] W. J. Stewart, *Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, Princeton, 1994.

[28] H. Taha, *Operations Research: An Introduction*, Prentice Hall, 1997.