

Open Mathematics

Research Article

Chun Wen, Ting-Zhu Huang, Xian-Ming Gu*, Zhao-Li Shen, Hong-Fan Zhang, and Chen Liu

Multipreconditioned GMRES for simulating stochastic automata networks

<https://doi.org/10.1515/math-2018-0083>

Received October 29, 2017; accepted July 9, 2018.

Abstract: Stochastic Automata Networks (SANs) have a large amount of applications in modelling queueing systems and communication systems. To find the steady state probability distribution of the SANs, it often needs to solve linear systems which involve their generator matrices. However, some classical iterative methods such as the Jacobi and the Gauss-Seidel are inefficient due to the huge size of the generator matrices. In this paper, the multipreconditioned GMRES (MPGMRES) is considered by using two or more preconditioners simultaneously. Meanwhile, a selective version of the MPGMRES is presented to overcome the rapid increase of the storage requirements and make it practical. Numerical results on two models of SANs are reported to illustrate the effectiveness of these proposed methods.

Keywords: Preconditioner, GMRES, Arnoldi process, Stochastic automata network

MSC: 65F10, 90B15

1 Introduction

The use of Stochastic Automata Networks (SANs) is of interest in a wide range of applications, including modeling queueing systems [1–3], communication systems [4, 5], manufacturing systems and inventory control [6]. The generator matrices of these SANs are block tridiagonal matrices with each block diagonal being a sum of tensor products of matrices. To analyze the system performance of a SAN, it is required to find the steady state probability distribution of the SAN. The solution can be obtained by solving a large linear system involving its generator matrix.

Chun Wen: School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China, E-mail: wchun17@163.com

Ting-Zhu Huang: School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China, E-mail: tingzhu Huang@126.com

***Corresponding Author: Xian-Ming Gu:** School of Economic Mathematics/Institute of Mathematics, Southwestern University of Finance and Economics, Chengdu, Sichuan 611130, China and Johann Bernoulli Institute of Mathematics and Computer Science, University of Groningen, Nijenborgh 9, P.O. Box 407, 9700 AK Groningen, The Netherlands, E-mail: guxianming@live.cn, x.m.gu@rug.nl

Zhao-Li Shen: School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China and Johann Bernoulli Institute of Mathematics and Computer Science, University of Groningen, Nijenborgh 9, P.O. Box 407, 9700 AK Groningen, The Netherlands, E-mail: szlxiaoyao@163.com

Hong-Fan Zhang: School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, China, E-mail: zhfbeyond@126.com

Chen Liu: Dietrich School of Arts and Sciences, University of Pittsburgh, 139 University Pl, Pittsburgh, PA 15260, USA, E-mail: CHL254@pitt.edu

Iterative procedures are often employed to compute the steady state probability distribution of SANs, such as the classical Jacobi, Gauss-Seidel, SOR, SSOR and TSS iterative methods [3, 6–8]. These methods are simple and easy to implement, but they usually suffer from slow convergence and they do not appear to be practical when the size of the generator matrix is huge [6, 9, 10]. Hence, Krylov subspace methods which use the matrix-vector multiplications as their main arithmetic operations are developed. For instance, the popular Arnoldi/FOM, GMRES, CGS, BiCGSTAB and QMR methods and so on [7, 11–18]. Based on a small-dimension Krylov subspace, such iterative methods are able to efficiently approximate the steady state probability distribution of SANs [7, 10, 19]. However, they may also suffer from slow convergence, even though lots of numerical experiments indicate that they generally outperform the classical iterative methods, refer, e.g., to [3, 7] for details.

It is well known that preconditioning techniques are necessary to improve the convergence of iterative methods by modifying the eigenvalue distribution of the iteration matrix, while the solution remains unchanged. Preconditioners based on incomplete LU factorizations are possible choices and have been discussed in [3, 7, 20]. A SAN preconditioner (based on the Neumann) series was proposed by Stewart et al. [10]. Numerical experiments show that the SAN preconditioner is expensive and needs more executing time. For taking use of the tensor structure of the generator matrix of a SAN, an approximate inverse preconditioner for the SAN was proposed by Langville and Stewart [19]. It is called the Nearest Kronecker Product (NKP) preconditioner. Numerical tests on some SANs have shown the effectiveness of the NKP preconditioner, however it is not always appropriate for general Markov chains [21]. Taking circulant approximations of the tensor blocks of the generator matrix, circulant preconditioners were constructed for a SAN [2, 4]. These are simple to construct and as since circulant matrices can be easily diagonalized by using the fast Fourier transforms [22]. However, the successful applications of these circulant preconditioners depend on whether the tensor blocks of the generator matrix of a SAN are near-Toeplitz matrices. In fact, different preconditioners have different efficiencies for some model problems. Which one is better and how to develop more efficient preconditioners, especially for very large systems, remains to be done in the future.

In this paper, motivated by the idea of [23], our main contribution is to find the steady state probability distribution of a SAN by employing a variant of GMRES where two or more preconditioners are applied simultaneously, and denote it as multipreconditioned GMRES (MPGMRES). Compared to the standard GMRES, one advantage of the MPGMRES is that it is able to find an approximate solution over a rich search space that has been enlarged by collecting all possible search direction generated by different preconditioners at each iteration. However, its disadvantage is that more memory requirements are required since the search space has an exponential growth at each iteration. Fortunately, a selective version of the MPGMRES (sMPGMRES) is considered by discarding some search directions such that the search space only has a linear growth with the size of problems increasing at each iteration. In fact, the idea to use multiple preconditioners is not new, e.g., Bridson et al. [24] has proposed the multipreconditioned conjugate gradients (MPCG) for symmetric positive definite linear systems. Even though the short-term recurrence of the standard CG does not hold in general, the MPCG is very useful in terms of determining what the most effective preconditioner is by executing a few iterations of the MPCG. On the other hand, for linear systems with a nonsymmetric coefficient matrix, a combined preconditioning strategy has been analyzed in [25] and shows that an optimal linear combination of two preconditioners could obtain the good numerical performance.

The rest of this paper is organized as follows. In Section 2, a brief introduction of two models of SANs is given. The derivation of the MPGMRE and its selective version are discussed in Section 3. In Section 4, numerical experiments on two SANs are employed to illustrate the effectiveness of these methods. Finally, some conclusions are given in Section 5.

2 Two models of SANs

To introduce certain terminologies and notations of SANs, two models will be described in this section; a two-queue overflow network and another one is a telecommunication system.

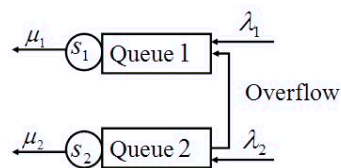
2.1 A two-queue overflow network

The early research on the two-queue overflow network can be found in [2, 26]. This network consists of two independent queues with the number of servers and waiting spaces being s_i and $l_i - s_i - 1$ ($i = 1, 2$) respectively, where l_i is the total number of customers in the queue i . For queue i , $i = 1, 2$, let λ_i be the arrival rate of customers, and let μ_i be the service rate of the servers, then the state space of the queueing system can be represented by the elements in the following set:

$$S = \{(i, j) | 0 \leq i \leq l_1, 0 \leq j \leq l_2\},$$

where i represents the state that there are i customers in the queue 1, and j represents the state that there are j customers in the queue 2. Thus it is a two-dimensional queueing system.

Fig. 1. The two-queue overflow network.



Generally speaking, the queueing discipline is First-come-first-served. When a customer arrives and finds all the servers are busy, the customer has two choices: one is to wait in the queue, provided that there is still a waiting space available, another is to leave the system. But for now we allow the overflow of customers from queue 2 to queue 1 whenever it is full and queue 1 is not, see Fig. 1. The generator matrix can be shown to be the following $l_1 l_2 \times l_1 l_2$ matrix in the tensor product form

$$A = Q_1 \otimes I_{l_2} + I_{l_1} \otimes Q_2 + R \otimes e_{l_2}^T e_{l_2}, \quad (1)$$

where

$$Q_i = \begin{pmatrix} \lambda_i & -\mu_i & & & & & 0 \\ -\lambda_i & \lambda_i + \mu_i & -2\mu_i & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & -\lambda_i & \lambda_i + (s_i - 1)\mu_i & -s_i\mu_i & & \\ & & & -\lambda_i & \lambda_i + s_i\mu_i & -s_i\mu_i & \\ & & & & \ddots & \ddots & \ddots \\ & & & & & -\lambda_i & \lambda_i + s_i\mu_i & -s_i\mu_i \\ 0 & & & & & & -\lambda_i & s_i\mu_i \end{pmatrix}, \quad (2)$$

$$R = \begin{pmatrix} \lambda_2 & & & 0 \\ -\lambda_2 & \lambda_2 & & \\ & -\lambda_2 & \ddots & \\ & & \ddots & \lambda_2 \\ 0 & & & -\lambda_2 & 0 \end{pmatrix}, \quad (3)$$

and I_{l_i} is the identity matrix of size l_i , $i = 1, 2$, and e_{l_2} is the unit vector $(0, \dots, 0, 1)^T$, see [2, 26] for instance. Let $n = l_1 l_2$, then the steady state probability distribution vector $p = (p_1, p_2, \dots, p_n)^T$ is the solution of the linear system $Ap = 0$ with constraints

$$\sum_{i=1}^n p_i = 1, \quad \text{and} \quad p_i \geq 0, \quad \forall 1 \leq i \leq n. \quad (4)$$

According to the properties of the tensor product [9, 10, 27], the matrix $R \otimes e_{l_2}^T e_{l_2}$ is a lower triangular matrix which describes the overflow of customers from the queue 2 to the queue 1, and the matrix A in (1)

is a singular and block tridiagonal matrix. Particularly, the matrix A is irreducible, has zero column sums, positive diagonal elements and nonpositive off-diagonal elements. According to [28, 29], matrix A has a one-dimensional null space with a positive null vector. Hence, the existence of the steady state probability distribution vector p can be guaranteed.

2.2 A telecommunication system

Here an (MMPP/M/s/s+m) network arising in telecommunication systems is considered. It is known that the Markov Modulated Poisson Process (MMPP) is a generalization of the Poisson process and is often regarded as the input model of telecommunication system [4, 5]. To construct the (MMPP/M/s/s+m) network, first define the system parameters as follows:

- (a). $1/\lambda$, the mean arrival time of the exogenously originating calls of the main queue,
- (b). $1/\mu$, the mean service time of each server of the main queue,
- (c). s , the number of servers in the main queue,
- (d). $l - s - 1$, the number of waiting spaces in the main queue,
- (e). q , the number of overflow,
- (f). (Q_j, Λ_j) , $1 \leq j \leq q$, the parameters of the MMPP's modeling overflow parcels, where

$$Q_j = \begin{pmatrix} \sigma_{j1} & -\sigma_{j2} \\ -\sigma_{j1} & \sigma_{j2} \end{pmatrix}, \quad \Lambda_j = \begin{pmatrix} \lambda_j & 0 \\ 0 & 0 \end{pmatrix}.$$

Here σ_{j1} , σ_{j2} and λ_j , $1 \leq j \leq q$, are positive MMPP parameters.

The input of the main queue comes from the superposition of several independent MMPPs, which is still an MMPP and is parameterized by two $2^q \times 2^q$ matrices (Q, Ω) . Here

$$Q = (Q_1 \otimes I_2 \otimes \cdots \otimes I_2) + (I_2 \otimes Q_2 \otimes I_2 \otimes \cdots \otimes I_2) + (I_2 \otimes \cdots \otimes I_2 \otimes Q_q) \\ \Lambda = (\Lambda_1 \otimes I_2 \otimes \cdots \otimes I_2) + (I_2 \otimes \Lambda_2 \otimes I_2 \otimes \cdots \otimes I_2) + (I_2 \otimes \cdots \otimes I_2 \otimes \Lambda_q)$$

and $\Omega = \Lambda + \lambda I_{2^q}$, where I_2 and I_{2^q} are the 2×2 and $2^q \times 2^q$ identity matrices, respectively.

Now the (MMPP/M/s/l) network can be regarded as a Markov process on the state space

$$\{(i, j) | 0 \leq i \leq l - 1, 1 \leq j \leq 2^q\},$$

where the number i corresponds to the number of calls at the destination, and the number j corresponds to the state of the Markov process with generator matrix Q . Hence, it is a two-dimensional queueing system, and the generator matrix can be shown to be the following $l2^q \times l2^q$ tridiagonal block matrix:

$$A = \begin{pmatrix} Q + \Omega & -\mu I & & & & 0 \\ -\Omega & Q + \Omega + \mu I & -2\mu I & & & \\ & \ddots & \ddots & \ddots & & \\ & & -\Omega & Q + \Omega + s\mu I & -s\mu I & \\ & & & \ddots & \ddots & \ddots \\ & 0 & & & -\Omega & Q + \Omega + s\mu I & -s\mu I \\ & & & & & -\Omega & Q + s\mu I \end{pmatrix}.$$

It can be rewritten as in the tensor product form,

$$A = I_l \otimes Q + B \otimes I_{2^q} + R \otimes \Omega, \quad (5)$$

where matrices B and R has the same form as Q_i and R in (2) and (3), respectively. Let $n = l2^q$, then steady state probability distribution vector $p = (p_1, p_2, \dots, p_n)^T$ is the solution of the linear system $Ap = 0$ with the same constraints as (4).

Similarly, the matrix A in (5) is irreducible, has zero column sums, positive diagonal elements and nonpositive off-diagonal elements. According to the famous Perron-Frobenius theory, the matrix A has a one-dimensional null space with a positive null vector [28, 29]. Hence, the steady state probability distribution vector p exists.

Remark 2.1. To solve the linear system $Ap = 0$ efficiently, it is necessary to make certain modifications to the matrix A since it is singular. One possible way to obtain the steady state probability distribution vector p is to normalize the solution x of the following nonsingular linear system:

$$\mathcal{A}x = (A + \epsilon I)x = b, \quad (6)$$

where $b = (0, \dots, 0, 1)^T$ is an $n \times 1$ vector, I is an $n \times n$ identity matrix, and $\epsilon > 0$ can be chosen as small as possible. The matrix $\mathcal{A}$ is nonsingular because it is irreducible strictly diagonally dominant. The linear system (6) can be solved by GMRES. It is clear that solving the linear system (6) may induce a small error of $O(\epsilon)$, but fortunately Ching et al. have proved that the 2-norm of the error induced by the regularization tends to zero [2, 4].

3 Multipreconditioned GMRES

This section will give a brief description of the preconditioned GMRES, and then discuss the GMRES with multiple preconditioners (MPGMRES). At last, to decrease the computing cost of the MPGMRES, a selective version of the MPGMRES is considered.

3.1 Preconditioned GMRES

GMRES is often used to solve the non-symmetric linear system (6). Given an initial vector x_0 , then the corresponding initial residual is $r_0 = b - Ax_0$, and the Krylov subspace is:

$$\mathcal{K}_m(\mathcal{A}, r_0) = \text{span}\{r_0, \mathcal{A}r_0, \mathcal{A}^2r_0, \dots, \mathcal{A}^{m-1}r_0\}.$$

An orthonormal basis for the Krylov subspace $\mathcal{K}_m(\mathcal{A}, r_0)$ can be computed by the modified Gram-Schmidt orthogonalization procedure with the first vector being $r_0/\|r_0\|_2$. This generates a useful relation:

$$\mathcal{A}V_m = V_{m+1}\tilde{H}_m, \quad (7)$$

where $V_m \in \mathcal{R}^{n \times m}$ and $V_{m+1} \in \mathcal{R}^{n \times (m+1)}$ has orthonormal columns, and $\tilde{H}_m \in \mathcal{R}^{(m+1) \times m}$ is an upper Hessenberg matrix. Thus at the k -th step, an approximate solution of (6) is computed as:

$$x_k = x_0 + V_m y_m \in x_0 + \mathcal{K}_m(\mathcal{A}, r_0). \quad (8)$$

where $y_m = \min_{y \in \mathcal{R}^m} \|\beta e_1 - \tilde{H}_m y\|_2$ with $\beta = \|r_0\|_2$ and $e_1 = (1, 0, \dots, 0)^T$.

The preconditioning technique is a key ingredient for the successful applications of GMRES. Let M be a preconditioner, then for the linear system (6), the right preconditioned GMRES is considered in Algorithm 1, refer to Ref. [7].

In line 11 of Algorithm 1, the approximate solution x_m is expressed as a linear combination of the preconditioned vectors $z_i = M^{-1}v_i, i = 1, 2, \dots, m$, where the preconditioner M is fixed, i.e., it does not change from step to step. Now suppose that the preconditioner is able to change at every step:

$$z_j = M_j^{-1}v_j, j = 1, \dots, m.$$

Then the approximate solution is computed by $x_m = x_0 + Z_m y_m$, where $Z_m = (z_1, \dots, z_m)$. Such kind of GMRES is called Flexible GMRES (FGMRES) since different preconditioners are allowed to be used at each iteration [30].

Algorithm 1 The right preconditioned GMRES

-
- 1: Give an initial guess x_0 , and compute $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$, $v_1 = r_0/\beta$.
 - 2: **for** $j = 1, \dots, m$ **do**
 - 3: Compute $w = AM^{-1}v_j$.
 - 4: **for** $i = 1, \dots, j$ **do**
 - 5: $h_{i,j} = (w, v_i) = v_i^T w$.
 - 6: $w = w - h_{i,j}v_i$.
 - 7: **end for**
 - 8: Compute $h_{j+1,j} = \|w\|_2$ and $v_{j+1} = w/h_{j+1,j}$.
 - 9: **end for**
 - 10: Let $V_m = (v_1, \dots, v_m)$, $\tilde{H}_m = \{h_{ij}\}$, $1 \leq i \leq j+1$, $1 \leq j \leq m$.
 - 11: Compute $y_m = \min_{y \in \mathcal{R}^m} \|\beta e_1 - \tilde{H}_m y\|_2$, and $x_m = x_0 + M^{-1}V_m y_m$.
 - 12: If satisfied stop, else set $x_0 = x_m$ and go to step 1.
-

3.2 Multiple preconditioned GMRES

Based on the idea of using different preconditioners in the FGMRES, the multipreconditioned GMRES with two or more preconditioners being applied simultaneously was proposed by Greif et. al [23]. Assume there are k preconditioners, i.e., $M_1, \dots, M_k$, $k \geq 2$, then at the beginning, for the initial residual r_0 , we have $v_1 = r_0/\|r_0\|_2$ and

$$Z_1 = (M_1^{-1}v_1, \dots, M_k^{-1}v_1) \in \mathcal{R}^{n \times k}, \quad (9)$$

such that the first iteration is computed by $x_1 = x_0 + Z_1 y_1$, where the vector $y_1 \in \mathcal{R}^k$ is chosen to minimize $\|b - Ax_1\|_2$. From (9), it is easy to see that using all preconditioners simultaneously has enlarged the space where the solution is sought. Similar to Algorithm 1, the multipreconditioned GMRES algorithm (i.e., Algorithm 2) is given as follows.

Algorithm 2 Multipreconditioned GMRES (MPGMRES)

-
- 1: Give an initial guess x_0 , and compute $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$, $V_1 = r_0/\beta$.
 - 2: Compute the research directions $Z_1 = (M_1^{-1}V_1, \dots, M_k^{-1}V_1)$.
 - 3: **for** $j = 1, \dots, m$ **do**
 - 4: $W = AZ_j$.
 - 5: **for** $i = 1, \dots, j$ **do**
 - 6: $H_{i,j} = (W, V_i) = V_i^T W$.
 - 7: $W = W - V_i H_{i,j}$.
 - 8: **end for**
 - 9: $W = V_{j+1} H_{j+1,j}$. (QR factorization)
 - 10: Compute $y_j = \min_{y \in \mathcal{R}^m} \|\beta e_1 - \tilde{H}_m y\|_2$, and $x_j = x_0 + [Z_1, \dots, Z_j] y_j$.
 - 11: If satisfied stop, otherwise continue;
 - 12: $Z_{j+1} = (M_1^{-1}V_{j+1}, \dots, M_k^{-1}V_{j+1})$.
 - 13: **end for**
 - 14: If unsatisfied, set $x_0 = x_m$ and go to step 1.
-

Comparing with Algorithm 1, the search space increases at each iteration, and the relation in the equation (7) has been replaced by:

$$AZ_m = \tilde{V}_{m+1} \tilde{H}_m, \quad (10)$$

where

$$\tilde{Z}_m = (Z_1, \dots, Z_m), \quad \tilde{V}_{m+1} = (V_1, \dots, V_{m+1}),$$

and

$$\tilde{H}_m = \begin{pmatrix} H_{1,1} & H_{1,2} & \cdots & H_{1,m} \\ H_{2,1} & H_{2,2} & \cdots & H_{2,m} \\ & \ddots & & \vdots \\ & & H_{m,m-1} & H_{m,m} \\ & & & H_{m+1,m} \end{pmatrix},$$

in which the matrices V_{j+1} and $H_{j+1,j}$ ($1 \leq j \leq m$) are computed by QR factorization in the line 8 of Algorithm 2. Note that matrices Z_j and V_{j+1} have k^j columns, $j = 1, \dots, m$. Thus the matrix $\tilde{V}_{m+1}$ has:

$$\theta_m = \sum_{j=0}^m k^j = \frac{k^{m+1} - 1}{k - 1} \quad (11)$$

columns, the matrix $\tilde{Z}_m$ has $\theta_m - 1 = (k^{m+1} - k)/(k - 1)$ columns, and the size of the upper hessenberg matrix $\tilde{H}_m$ is $\theta_m \times (\theta_m - 1)$.

Let $\mathcal{P}_{m-1} = \mathcal{P}_{m-1}(X_1, \dots, X_k)$ be the space of all possible polynomials of matrices in k variables of at most degree $m - 1$, then at the j -th step of the MPGMRES, the approximate solution can be represented by:

$$x_j = x_0 + \sum_{i=1}^k \omega_j^i (M_1^{-1} \mathcal{A}, \dots, M_k^{-1} \mathcal{A}) M_i^{-1} r_0, \quad (12)$$

where $\omega_j^i \in \mathcal{P}_{m-1}(X_1, \dots, X_k)$, see [23] for details. Furthermore, from (12), the corresponding residual can be computed as:

$$\begin{aligned} r_j &= r_0 + \sum_{i=1}^k \omega_j^i (\mathcal{A} M_1^{-1}, \dots, \mathcal{A} M_k^{-1}) \mathcal{A} M_i^{-1} r_0 \\ &= \sum_{i=1}^k (\tau_i + \omega_j^i (\mathcal{A} M_1^{-1}, \dots, \mathcal{A} M_k^{-1} \mathcal{A} M_i^{-1})) r_0 \\ &= \sum_{i=1}^k \beta_{j+1}^i (\mathcal{A} M_1^{-1}, \dots, \mathcal{A} M_k^{-1}) r_0, \end{aligned} \quad (13)$$

where $\beta_{j+1}^i \in \mathcal{P}_m(X_1, \dots, X_k)$, and τ_i satisfies

$$\sum_{i=1}^k \beta_{j+1}^i(0, \dots, 0) = \sum_{i=1}^k \tau_i = 1,$$

and

$$\frac{\partial^{j+1} \beta_{j+1}^i}{\partial X_s^{j+1}} = 0, \quad 1 \leq i, s \leq k, \quad i \neq s,$$

which implies that, in the matrix polynomial β_{j+1}^i , only the i -th variable may have the highest degree j , the degrees of all other variables are less than or equal to $j - 1$. From (13), the following result is established.

Theorem 3.1. Let r_0 be the initial residual of the linear system (6), and r_j be the residual at the j -th step of the MPGMRES with k preconditioners $M_1, \dots, M_k$. Then it has:

$$\frac{\|r_j\|}{\|r_0\|} \leq \min_{\substack{\beta_{j+1}^i \in \mathcal{P}_m(X_1, \dots, X_k), 1 \leq i \leq k \\ \sum_{i=1}^k \beta_{j+1}^i(0, \dots, 0) = 1, \frac{\partial^{j+1} \beta_{j+1}^i}{\partial X_s^{j+1}} = 0, i \neq s}} \left\| \sum_{i=1}^k \beta_{j+1}^i (\mathcal{A} M_1^{-1}, \dots, \mathcal{A} M_k^{-1}) \right\|. \quad (14)$$

Therefore, it is important to find an optimal combination of all the different preconditioners as they are used simultaneously at each iteration [23–25]. This problem still requires further research.

3.3 Selective MPGMRES and computing cost

The idea of the MPGMRES is to enlarge the Krylov subspace over which the GMRES minimizes the residual norm by using two or more preconditioners simultaneously at each iteration. From (14), it can be seen that the reason why the search space is so rich is that it not only has the polynomials of higher order of the variables X_i , but it also has many mixed terms [23]. For instance, when $k = 2$, the entire space of the matrix polynomials is

$$\mathcal{P}_2(X_1, X_2) = \text{span}\{I, X_1, X_2, X_1X_2, X_2X_1, X_1^2, X_2^2\}. \quad (15)$$

It is natural to think of reducing the dimension of the entire space. Specially, for the case when $\mathcal{A} = M_1 + M_2$, the following result has been proved [23].

Theorem 3.2. *If the variables X_1, X_2 satisfy the condition that $X_1X_2 = X_2X_1 = X_1 + X_2$, then*

$$\mathcal{P}_k(X_1, X_2) = \mathcal{P}_k(X_1) + \mathcal{P}_k(X_2), \quad k = 1, 2.$$

The mixed terms have been eliminated, which indicates that the dimension of the entire search space has been reduced successfully. Hence, it is a possible choice to construct an approximate search space by discarding some search directions at each iteration. The computing cost of the MPGMRES will then be decreased.

Several different strategies can be applied to obtain the approximate search space at each iteration, see [23, 31]. In this paper, a one simple method shown in [23], where line 11 of Algorithm 2 has been replaced by

$$Z_{j+1} = (M_1 V_{j+1}^1, M_2 V_{j+1}^2, \dots, M_k V_{j+1}^k),$$

i.e., the preconditioner M_1 is used in the first column of V_{j+1} , M_2 is used in the second column of V_{j+1} , and so on. This is called selective MPGMRES (sMPGMRES). Correspondingly, the column of the matrix Z_j , $j = 1, \dots, m$, remains k rather than k^j . For comparisons, at the j -th iteration, the computing cost of the MPGMRES and sMPGMRES is listed as follows [23].

Table 1. Comparisons of computing cost.

	matrix-vector products	inner products
MPGMRES	k^j	$\frac{k^{2j+1} + k^{2j} + k^{j+1} - 3k^j}{2(k-1)}$
sMPGMRES	k	$(j - \frac{1}{2})k^2 + \frac{3}{2}k$

From Table 1, it can be seen that the computing cost of the MPGMRES has an exponential growth, while that of the sMPGMRES only grows linearly.

4 Numerical experiments

In this section, Algorithms 1-2 are tested and the variant (sMPGMRES) introduced in this paper on two models of SANs. One example is from queueing systems, the other from telecommunication systems. In particular, we compare the numerical performance of the MPGMRES, sMPGMRES, standard right preconditioned GMRES and unpreconditioned GMRES (with restart being 10) in terms of the total computing time and iteration steps. For convenience, GMRES(M) is used to denote the right preconditioned GMRES method with the preconditioner M in this section. Convergence histories are shown in figures with the number of iterations on the horizontal axis, and Relres (defined as $\log_{10}(\|r_j\|_2/\|r_0\|_2)$) on the vertical axis. All experiments are carried out with a serial MATLAB implementation of the code, in which parts of the MATLAB code are from https://github.com/tyronerees/krylov_solvers/tree/master/mpgmres.

Throughout our numerical experiments, the stopping criteria is set as $\|r_j\|_2/\|r_0\|_2 \leq 10^{-6}$, where r_j is the current residual and r_0 is the initial residual. The initial vector for all methods is chosen to be $x_0 =$

$(0, \dots, 0, 1)^T$. The parameter ϵ in the equation (6) is given as $\epsilon = 0.5$, other numbers can also be used if they can make the coefficient matrix A be nonsingular.

Example 4.1. The two-queue overflow network [2, 26]. This test problem was introduced in Section 2.1. The corresponding matrices can be found in the equations (1), (2) and (3). Here set $\lambda_1 = \mu_1 = s_1 = 1$, and $\lambda_2 = \mu_2 = s_2 = 2$. The size of the matrix A is $n = l_1 l_2$. To find the steady state probability distribution, different ways can be applied to construct different preconditioners in order to solve the linear system (6).

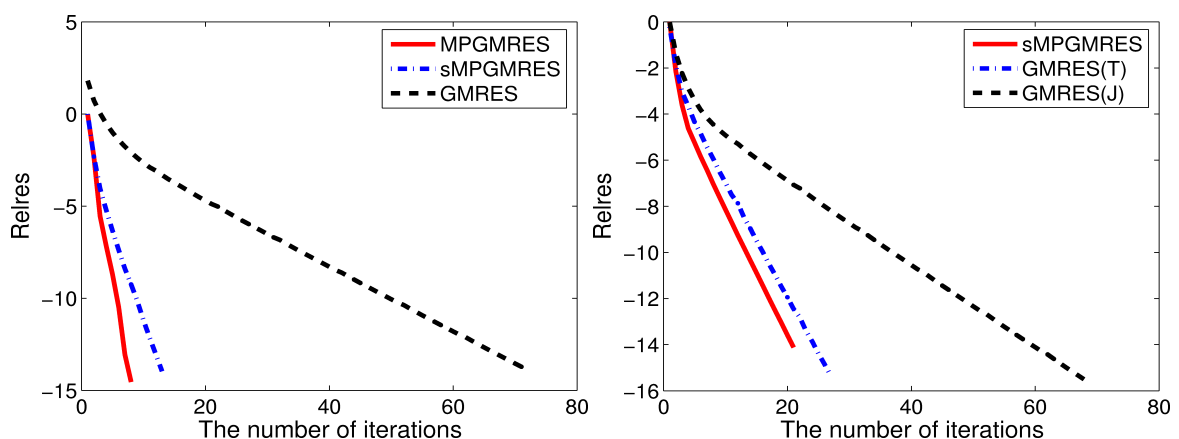
Case one: Consider the incomplete LU factorization with “droptol = 0.001” for the coefficient matrix A , i.e., $A = LU + E$, where L is an unit lower triangular matrix and U is an upper triangular matrix. Hence the matrices L and U are used as two preconditioners for the GMRES. Numerical results for this case are shown in Table 2.

Table 2. The number of iterations and computing time in seconds (in brackets) for Example 4.1 when L and U are used as preconditioners. The last column Ratio is defined as (computing time of sMPGMRES)/(computing time of GMRES).

l_1	l_2	MPGMRES	sMPGMRES	GMRES	Ratio
128	64	7 (8.8789)	13 (0.0619)	72 (0.0872)	0.71
128	128	7 (24.1441)	12 (0.1136)	71 (0.1353)	0.84
256	256	7 (106.1802)	12 (0.5188)	71 (0.7215)	0.72
512	256	7 (328.3111)	13 (1.9379)	71 (2.6381)	0.73
512	512	7 (634.2202)	12 (3.5193)	71 (6.7186)	0.52

Table 2 shows that the iteration counts of the MPGMRES and sMPGMRES with two preconditioners L and U being used simultaneously are less than those of the unpreconditioned GMRES. In particular, MPGMRES gives the best iteration steps, even though its computing time is the worst. However, it can be seen that the computing time of the sMPGMRES is superior to that of the unpreconditioned GMRES. The last column in Table 2 further confirms the fast convergence of the sMPGMRES. For instance, when $l_1 = 512$, $l_2 = 512$, the computing time needed by the sMPGMRES is almost only half of that needed by the unpreconditioned GMRES. Moreover, Fig. 2 (left) plots their convergent histories when $l_1 = 128$ and $l_2 = 128$.

Fig. 2. Convergent histories of the MPGMRES, sMPGMRES and unpreconditioned GMRES for Example 4.1 when $l_1 = 128$ and $l_2 = 128$ (left). Convergence histories of the sMPGMRES, GMRES(T) and GMRES(J) for Example 4.1 when $l_1 = 512$ and $l_2 = 512$ (right).



Case two: Let D and T be the diagonal and tridiagonal matrices of the matrix A , respectively. Then, correspondingly, the matrices J (equals to D) and T are considered as the Jacobi preconditioner and the tridiagonal preconditioner for the GMRES. Numerical results for this case are listed in Table 3.

Table 3. The number of iterations and computing time in seconds (in brackets) for Example 4.1 when J and T are used as preconditioners. Ratio1 and Ratio2 are defined as (computing time of sMPGMRES)/(computing time of GMRES(T)) and (computing time of sMPGMRES)/(computing time of GMRES(J)), respectively.

l_1	l_2	sMPGMRES	GMRES(T)	GMRES(J)	Ratio1	Ratio2
128	64	20 (0.1256)	26 (0.0581)	68 (0.0738)	2.16	1.70
128	128	20 (0.2043)	26 (0.1102)	67 (0.1146)	1.85	1.78
256	256	20 (0.5576)	26 (0.5593)	67 (0.6435)	1.00	0.87
512	256	20 (0.9771)	26 (1.5498)	68 (2.7222)	0.63	0.35
512	512	20 (1.9698)	26 (3.7703)	67 (6.6672)	0.52	0.30

From Table 3, it can see that the number of iterations required by the sMPGMRES is the best. In terms of the computing time, it is clear to find that the sMPGMRES is superior to the GMRES(T) and GMRES(J) when the size of this test problem becomes larger. While the sMPGMRES needs more computing time than the GMRES(T) and GMRES(J) when the size of this test problem is small. The last two columns of Table 3 illustrate the performance of the sMPGMRES with the size of the test problem increasing. Hence, it shows that, by discarding some search direction at each iteration, using different preconditioners for the GMRES simultaneously can be more effective than using a single preconditioner at each step. Fig. 2 (right) shows their convergent curves when $l_1 = 512$ and $l_2 = 512$.

Example 4.2. The telecommunication system [4, 5]. This example has been introduced in Section 2.2. The corresponding matrices can be found in the equations (2), (3) and (5). Here we set $\lambda = \mu = s = 2$, $q = 4$, $\sigma_{j2} = 1/3$ and $\lambda_j = 1/q$, $j = 1, \dots, q$. The size of the coefficient matrix A is $n = l2^q$. For finding the steady state probability distribution, similar ways to Example 4.1 have been used to construct different preconditioners.

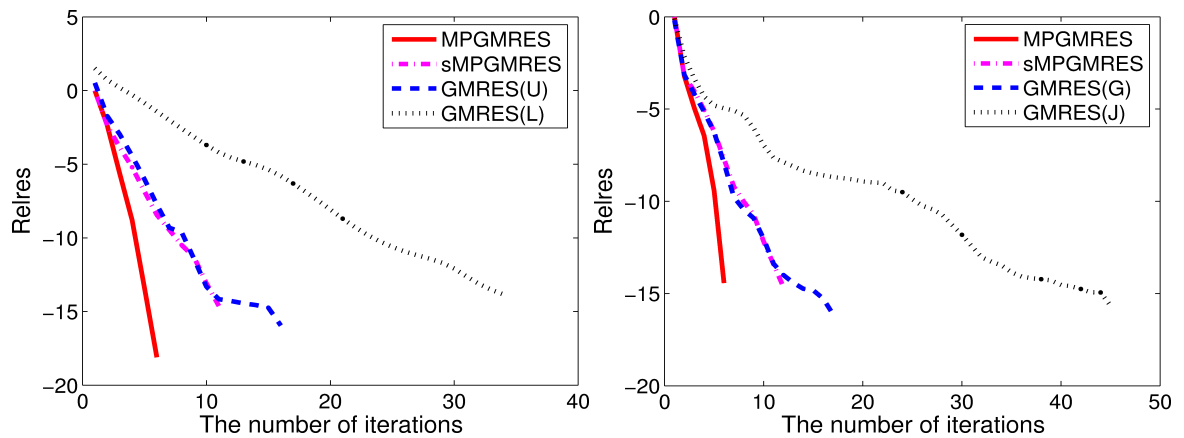
Case one: Using the same incomplete LU factorization as given above for the coefficient matrix A generated by (6). Then again, the matrices L and U are used as two preconditioners for the GMRES. The corresponding numerical results for this case are presented in Table 4.

Table 4. The number of iterations and computing time in seconds (in brackets) for Example 4.2, when L and U are used as preconditioners.

l	MPGMRES	sMPGMRES	GMRES(L)	GMRES(U)
512	5 (0.5675)	10 (0.0677)	33 (0.0545)	15 (0.0308)
1024	5 (1.9263)	10 (0.1765)	33 (0.1366)	15 (0.0752)
2048	5 (2.0571)	10 (0.2922)	33 (0.7859)	15 (0.2084)
4096	5 (2.2701)	10 (0.4457)	33 (3.3516)	15 (0.3071)
8192	5 (2.5883)	10 (0.6451)	33 (9.0279)	15 (0.7694)
16384	5 (3.2082)	10 (0.9496)	33 (19.7403)	15 (1.5723)

As seen from Table 4, in the sense of iteration counts, the MPGMRES and sMPGMRES with two preconditioners being applied simultaneously are superior to the GMRES with single preconditioner. And in terms of the computing time, the MPGMRES and sMPGMRES cost more than the GMRES(L) and GMRES(U) when the size of this test problem is small. The reason may be that the dimension of the Krylov subspaces has an increase when two preconditioners are used simultaneously. However, when the size of the test problem increases, it is not difficult to see that the sMPGMRES has given the best results, e.g., when $l = 16384$, the computing time of the GMRES(L) and GMRES(U) have been reduced by 95% and 40%, respectively. Fig. 3 (left) shows their convergent histories when $l = 8192$. Note that the computing time of the sMPGMRES is always less than that of the MPGMRES, which indicates that appropriately discarding some search direction from MPGMRES can lead to fast convergence, even though the iteration steps have a little increase. Therefore, the GMRES with multiple preconditioners is a competitive and robust choice for the large test problem.

Fig. 3. Convergent histories of the MPGMRES, sMPGMRES, GMRES(L) and GMRES(U) for Example 4.2 when $l = 8192$ (left). Convergent histories of the MPGMRES, sMPGMRES, GMRES(G) and GMRES(J) for Example 4.2 when $l = 8192$ (right).



Case two: Let us split the coefficient matrix A as $A = D - E - F$, where D is the diagonal of A , $-E$ and $-F$ are the strict lower triangular and strict upper triangular of A , respectively. There is also the Gauss-Seidel (GS) preconditioner $G = D - F$ and the Jacobi preconditioner J (equals to D). Numerical results for this case are supplied in Table 5.

Table 5. The number of iterations and computing time in seconds (in brackets) for Example 4.2, when G and J are used as preconditioners.

l	MPGMRES	sMPGMRES	GMRES(G)	GMRES(J)
512	5 (1.0332)	11 (0.2172)	16 (0.0588)	44 (0.0502)
1024	5 (1.6542)	11 (0.3605)	16 (0.1154)	44 (0.0897)
2048	5 (1.7015)	11 (0.3905)	16 (0.1994)	44 (0.2741)
4096	5 (1.8292)	11 (0.4545)	16 (0.4262)	44 (0.5751)
8192	5 (2.1279)	11 (0.6181)	16 (0.8906)	44 (2.1283)
16384	5 (2.7101)	11 (0.9103)	16 (2.0081)	44 (4.7786)

Again, from Table 5, it is shown that the iteration counts of the MPGMRES and sMPGMRES are less than those of the GMRES(G) and GMRES(J). With the increase of the size of this test problem, the computing time of the MPGMRES and sMPGMRES are superior to that of the GMRES(J), although the computing time of the MPGMRES is inferior to that of the GMRES(G). In particular, the sMPGMRES shows the best results, e.g., when $l = 16384$, the computing time of the GMRES(G) and GMRES(J) are reduced by 55% and 81%, respectively. Furthermore, Fig. 3 (right) plots their convergent histories when $l = 8192$, with G and J being two preconditioners.

Case three: Three preconditioners are applied to the GMRES. Observe the equation (5), it can be seen that the matrix A consists of three parts. According to the properties of the tensor product [9, 10, 27], the last part of the matrix A , i.e., $R \otimes \Omega$, is a lower triangular matrix. Since the last diagonal element of the matrix R is zero as shown in the form of (3), the tensor product $R \otimes \Omega$ is a singular matrix. To modify its singularity, let the last diagonal element of R be equal to λ , and denote the new matrix as $\tilde{R}$. Then let $M_3 = \tilde{R} \otimes \Omega$. It is nonsingular. On the other hand, using the tensor construct of the matrix A given in (5), we let M_1 and M_2 be the lower triangular matrices of the tensor product $I_1 \otimes Q$ and $B \otimes I_{2^q}$, respectively. Then three preconditioners M_1 , M_2 and M_3 for the linear system (6) are considered for the GMRES. Numerical results for this case are provided in Table 6.

As seen from Table 6, it finds that the MPGMRES has shown the best iteration steps. In addition, the number of iterations of the sMPGMRES is also less than those of the GMRES(M_1), GMRES(M_2) and GMRES(M_3). From the last column of Table 6, it also observes that the preconditioner M_3 is not effective since its iteration counts have a large change and the computing time has a dramatic increase with the size of this test problem increasing.

Table 6. The number of iterations and computing time in seconds (in brackets) for Example 4.2, when M_1 , M_2 and M_3 are used as preconditioners.

l	MPGMRES	sMPGMRES	GMRES(M_1)	GMRES(M_2)	GMRES(M_3)
512	20 (3.6583)	23 (0.1967)	38 (0.0451)	37 (0.0916)	194 (0.2613)
1024	20 (4.4979)	23 (0.2336)	38 (0.0919)	37 (0.2168)	336 (0.7794)
2048	20 (5.9025)	23 (2.3188)	38 (0.2335)	37 (0.3816)	595 (4.0581)
4096	20 (8.2599)	23 (0.4523)	38 (0.4792)	37 (0.6017)	1025 (14.2911)
8192	20 (11.6451)	23 (0.9709)	38 (1.7371)	37 (1.7914)	2080 (104.7501)
16384	20 (15.7614)	23 (1.8284)	38 (4.2532)	37 (4.2274)	4120 (493.6202)

However, the preconditioner M_3 does not affect the stability of the MPGMRES and sMPGMRES, i.e., their iteration steps have no changes and their computing time are acceptable. In particular, for large test problem, the sMPGMRES gives the best computing time, e.g., when $l = 16384$, the computing time of the GMRES(M_1), GMRES(M_2) and GMRES(M_3) are reduced by 57%, 57% and 99%, respectively. Hence, the acceptable efficiency of the GMRES with multiple preconditioners is illustrated again.

5 Conclusions

In the current study, the main contribution is to consider the MPGMRES for computing the steady state probability distribution of SANs. The idea of the MPGMRES is to enlarge the Krylov subspace over which the GMRES minimizes the residual norm by using two or more preconditioners simultaneously at each iteration. Since the dimension of the search space has an exponential growth at each iteration, a practical sMPGMRES is given by discarding some search directions at each iteration, such that the dimension of the search space only has a linear growth with the problem size. Numerical experiments on two models of SANs have illustrated that the MPGMRES is more effective than the GMRES with a single preconditioner in reducing the number of iterations. Moreover, for large test problems, the computing time of the sMPGMRES is the best.

Here it is worth noting that only the MPGMRES has been tested on two models of SANs. In fact, there are several other models of SANs, e.g., manufacturing systems [6, 32], node mobility in wireless networks [33] and FTS projects [34]. Hence, it would be interesting to extend method to these models in the future.

Acknowledgement: The authors would like to thank the editor Dr. Agnieszka Bednarczyk-Dragthe and anonymous referees for their constructive comments and helpful suggestions in revising the paper. This research is supported by the National Natural Science Foundation of China (Nos. 11501085 and 61772003) and the Fundamental Research Funds for the Central Universities (No. JBK1809003).

References

- [1] Buchholz P., A class of hierarchical queueing networks and their analysis, Queueing Sys., 1994, 15, 59-80.
- [2] Chan R.H., Ching W.K., Circulant preconditioners for stochastic automata networks, Numer. Math., 2000, 87, 35-57.
- [3] Stewart W.J., Probability, Markov Chains, Queues, and Simulation, 2009, Princeton University Press, Princeton, NJ.
- [4] Ching W.K., Chan R.H., Zhou X.Y., Circulant preconditioners for Markov-modulated Poisson process and their applications to manufacturing systems, SIAM J. Matrix Anal. Appl., 1997, 18, 464-481.
- [5] Meier-Hellstern K., The analysis of a queue arising in overflow models, IEEE Trans. Commun., 1989, 37, 367-372.
- [6] Buzacott J., Shanthikumar J., Stochastic Models of Manufacturing Systems, 1993, Prentice-Hall International Editions, Upper Saddle River, NJ.
- [7] Saad Y., Iterative Methods for Sparse Linear Systems, 2nd ed., 2003, SIAM, Philadelphia, PA.
- [8] Wen C., Huang T.-Z., Wang C., Triangular and skew-symmetric splitting method for numerical solutions of Markov chains, Comput. Math. Appl., 2011, 62, 4039-4048.

- [9] Langville A.N., Stewart W.J., The Kronecker product and stochastic automata networks, *J. Comput. Appl. Math.*, 2004, 167, 429-447.
- [10] Stewart W.J., Atif K., Plateau B., The numerical solution of stochastic automata networks, *Eur. J. Oper. Res.*, 1995, 86, 503-525.
- [11] Saad Y., Schultz M.H., GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.*, 1986, 7, 856-896.
- [12] Zhang H.-F., Huang T.-Z., Wen C., Shen Z.-L., FOM accelerated by an extrapolation method for solving PageRank problems, *J. Comput. Appl. Math.*, 2016, 296, 397-409.
- [13] Shen Z.-L., Huang T.-Z., Carpentieri B., Gu X.-M., Wen C., An efficient elimination strategy for solving PageRank problems, *Appl. Math. Comput.*, 2017, 298, 111-122.
- [14] Shen Z.-L., Huang T.-Z., Carpentieri B., Wen C., Gu X.-M., Block-accelerated aggregation multigrid for Markov chains with application to PageRank problems, *Commun. Nonlinear. Sci. Numer. Simulat.*, 2018, 59, 472-487.
- [15] Wu G., Wei Y., An Arnoldi-extrapolation algorithm for computing PageRank, *J. Comput. Appl. Math.*, 2010, 234, 3196-3212.
- [16] Wu G., Zhang Y., Wei Y., Accelerating the Arnoldi-type algorithm for the PageRank problem and the ProteinRank problem, *J. Sci. Comput.*, 2013, 57, 74-104.
- [17] Gu X.-M., Huang T.-Z., Yin G., Carpentieri B., Wen C., Du L., Restarted Hessenberg method for solving shifted nonsymmetric linear systems, *J. Comput. Appl. Math.*, 2018, 331, 166-177.
- [18] Wen C., Huang T.-Z., Shen Z.-L., A note on the two-step matrix splitting iteration for computing PageRank, *J. Comput. Appl. Math.*, 2017, 315, 87-97.
- [19] Langville A.N., Stewart W.J., A Kronecker product approximate preconditioner for SANs, *Numer. Linear. Algebra Appl.*, 2004, 11, 723-752.
- [20] Saad Y., Preconditioned Krylov subspace methods for the numerical solution of Markov chains, in Stewart W.J. (ed.), *Computations with Markov Chains*, 1995, 49-64, Springer, Boston, MA.
- [21] Langville A.N., Stewart W.J., Testing the nearest Kronecker product preconditioner on Markov chains and stochastic automata networks, *INFORMS J. Comput.*, 2004, 16, 300-315.
- [22] Davis P.J., *Circulant Matrices*, 1979, John Wiley & Sons, New York, NY.
- [23] Greif C., Rees T., Szyld D.B., GMRES with multiple preconditioners, *SeMA J.*, 2007, 74, 213-231.
- [24] Bridson R., Greif C., A multipreconditioned conjugate gradient algorithm, *SIAM J. Matrix Anal. Appl.*, 2006, 27, 1056-1068.
- [25] Dios B.A.D., Baker A.T., Vassilevski P.S., A combined preconditioning strategy for nonsymmetric systems, *SIAM J. Sci. Comput.*, 2014, 36, A2533-A2556.
- [26] Kaufman L., Matrix methods for queueing problems, *SIAM J. Sci. Stat. Comput.*, 1982, 4, 525-552.
- [27] Van Loan C.F., The ubiquitous Kronecker product, *J. Comput. Appl. Math.*, 2000, 123, 85-100.
- [28] Senta E., *Non-Negative Matrices*, 1973, John Wiley & Sons, New York, NY.
- [29] Varga R.S., *Matrix Iterative Analysis*, 1963, Prentice-Hall, Inc., Englewood Cliffs, NJ.
- [30] Saad Y., A flexible inner-outer preconditioned GMRES algorithm, *SIAM J. Sci. Comput.*, 1993, 14, 461-469.
- [31] Sturler E.D., Nested Krylov methods based on GCR, *J. Comput. Appl. Math.*, 1996, 67, 15-41.
- [32] Fernandes P., O'Kelly M.E.J., Papadopoulos C.T., Sales A., Analysis of exponential reliable production lines using Kronecker descriptors, *Int. J. Prod. Res.*, 2013, 51, 4240-4257.
- [33] Dotti F.L., Fernandes P., Nunes C.M., Structured Markovian models for discrete spatial mobile node distribution, *J. Braz. Comp. Soc.*, 2011, 17, 31-52.
- [34] Santos A.R., Sales A., Fernandes P., Using SAN formalism to evaluate Follow-The-Sun project scenarios, *J. Syst. Softw.*, 2015, 100, 182-194.