A Parallel Solver for Large-Scale Markov Chains

Michele Benzi [a,1] and Miroslav Tůma [b,2]

[a] *Department of Mathematics and Computer Science, Emory University Atlanta, GA 30322, USA*

[b] *Institute of Computer Science, Academy of Sciences of the Czech Republic Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic*

Dedicated to the memory of Rüdiger Weiss

Abstract

We consider the parallel computation of the stationary probability distribution vector of ergodic Markov chains with large state spaces by preconditioned Krylov subspace methods. The parallel preconditioner is obtained as an explicit approximation, in factorized form, of a particular generalized inverse of the generator matrix of the Markov process. Graph partitioning is used to parallelize the whole algorithm, resulting in a two-level method.

Conditions that guarantee the existence of the preconditioner are given, and the results of a parallel implementation are presented. Our results indicate that this method is well suited for problems in which the generator matrix can be explicitly formed and stored.

Key words: parallel preconditioning, iterative methods, discrete Markov chains, generalized inverses, singular matrices, graph partitioning, AINV, Bi-CGSTAB

[1] Email: benzi@mathcs.emory.edu. Part of this author's work was supported by the US Department of Energy through grant W-7405-ENG-36 with Los Alamos National Laboratory.

[2] Email: tuma@cs.cas.cz. Part of this author's work was supported by Grant Agency of the Czech Republic grant No. 2030801 and by Grant Agency of the Czech Republic grant No. 101/00/1035.

1 Introduction

Discrete Markov chains with large state spaces arise in many applications, including for instance reliability modeling, queueing network analysis, large scale economic modeling and computer system performance evaluation. The stationary probability distribution vector of an ergodic Markov process with $n \times n$ transition probability matrix P is the unique $1 \times n$ vector π which satisfies

$$\pi = \pi P, \quad \pi_i > 0, \quad \sum_{i=1}^{n} \pi_i = 1.$$

Letting $x = \pi^T$ and $A = I - P^T$, the computation of the stationary vector reduces to finding a nontrivial solution to the homogeneous linear system $Ax = 0$. The ergodicity assumption means that P (and therefore A) is irreducible. Perron–Frobenius theory [9] guarantees that A has rank $n-1$, and that the (one-dimensional) null space $\mathcal{N}(A)$ of A is spanned by a vector x with positive entries. Upon normalization in the ℓ^1-norm, this is the stationary distribution vector of the Markov process.

The coefficient matrix A is a singular M-matrix, called the *generator* of the Markov process. [3] The matrix A is nonsymmetric, although it is sometimes structurally symmetric. See [26] for a good introduction to Markov chains and their numerical solution.

Due to the very large number n of states typical of many real-world applications, there has been increasing interest in recent years in developing parallel algorithms for Markov chain computations; see [2], [5], [10], [17], [19], [24]. Most of the attention so far has focused on (linear) stationary iterative methods, including block versions of Jacobi and Gauss–Seidel [10], [19], [24], and on (nonlinear) iterative aggregation/disaggregation schemes specifically tailored to stochastic matrices [10], [17]. In contrast, little work has been done with parallel preconditioned Krylov subspace methods. Partial exceptions are [5], where a symmetrizable stationary iteration (Cimmino's method) was accelerated using conjugate gradients on a Cray T3D, and [19], where an out-of-core, parallel implementation of Conjugate Gradient Squared (with no preconditioning) was used to solve very large Markov models with up to 50 million states. The suitability of preconditioned Krylov subspace methods for solving Markov models has been demonstrated, e.g., in [25], although no discussion of parallelization issues is given there.

In this paper we investigate the use of a parallel preconditioned iterative method for large, sparse linear systems in the context of Markov chain com-

[3] Strictly speaking, the generator matrix is $Q = P - I = -A^T$. We work with A instead of Q to conform to the familiar notation of numerical linear algebra.

putations. The preconditioning strategy is a two-level method based on sparse approximate inverses, first introduced in [3]. However, due to the singularity of the generator matrix A, the applicability of approximate inverse techniques in this context is not obvious. That this is indeed possible is a consequence of the fact that A is a (singular) M-matrix.

The paper is organized as follows. In section 2 we discuss the problem of preconditioning singular equations in general, and we establish a link between some standard preconditioners and generalized inverses. Sections 3–5 are devoted to AINV preconditioning for Markov chain problems, including a discussion of the parallel implementation and a theoretical analysis of the existence of the preconditioner. Numerical tests are reported in section 6, and some conclusions are presented in section 7.

2 Preconditioning Markov chain problems

In the Markov chain context, preconditioning typically amounts to finding an easily invertible nonsingular matrix M (the preconditioner) which is a good approximation to A; a Krylov subspace method is then used to solve $M^{-1}Ax = 0$ (for left preconditioning) or $AM^{-1}y = 0$, $x = M^{-1}y$ (for right preconditioning). Notice that even if A itself is singular, the preconditioner must be nonsingular so as not to change the solution set, i.e., the null space $\mathcal{N}(A)$ of A. Preconditioners can be generated by means of splittings $A = M - N$, such as those used in stationary iterative methods including Jacobi, Gauss–Seidel, SOR and block versions of these schemes; see [26]. Also in this class are the popular incomplete LU (ILU) factorization preconditioners. ILU-type methods have been successfully applied to Markov chain problems by Saad [25] in a sequential environment. The existence of incomplete factorizations for nonsingular M-matrices was already proved in [20]; an investigation of the existence of ILU factorizations for singular M-matrices can be found in [11].

Incomplete factorization methods work quite well on a wide range of problems, but they are not easily implemented on parallel computers. For this and other reasons, much effort has been put in recent years into developing alternative preconditioning strategies that have natural parallelism while being comparable to ILU methods in terms of robustness and convergence rates. This work has resulted in several new techniques known as sparse approximate inverse preconditioners; see [7] for a recent survey and extensive references. Sparse approximate inverse preconditioners are based on directly approximating the inverse of the coefficient matrix A with a sparse matrix $G \approx A^{-1}$. The application of the preconditioner only requires matrix-vector products, which are easily parallelized. Until now, these techniques have been applied almost exclusively to nonsingular systems of equations $Ax = b$. The only exception seems to be [13], where the SPAI preconditioner [16] was used in connection with Fast Wavelet Transform techniques on singular systems stemming from discretizations of the Neumann problem for Poisson's equation.

The application of approximate inverse techniques in the singular case raises several interesting theoretical and practical questions. Because the inverse of A does not exist, it is not clear what matrix G is an approximation of. It should presumably be some generalized inverse of A, but which one? Note that this question can be asked of M^{-1} for any preconditioner $M \approx A$. In [26], page 143, it is stated that M^{-1} should be an approximation of the group generalized inverse $A^\sharp$, and that an ILU factorization $A \approx \bar{L}\bar{U}$ implicitly yields such an approximation: $(\bar{L}\bar{U})^{-1} \approx A^\sharp$. As we will see, this interpretation is not entirely correct and is somewhat misleading. The group inverse (see [12]) is only one of many possible generalized inverses. It is well known [21] that the group inverse plays an important role in the modern theory of finite Markov chains. However, it is seldom used as a computational tool, in part because its computation requires knowledge of the stationary distribution vector π.

As it turns out, different preconditioners result (implicitly or explicitly) in approximations M^{-1} to different generalized inverses of A, which are typically not the group inverse $A^\sharp$. Let us consider ILU preconditioning first. If A is a $n\times n$ irreducible, singular M-matrix, then A has the LDU factorization $A = LDU$ where L and U are unit lower and upper triangular matrices (respectively) and D is a diagonal matrix of rank $n-1$:

$$D = \mathrm{diag}(d_1, d_2, \ldots, d_{n-1}, 0), \quad d_i > 0 \;\text{ for } 1 \le i \le n-1$$

(see [26]). Notice that L and U are nonsingular M-matrices; in particular, L^{-1} and U^{-1} have nonnegative entries. Define the matrix

$$A^- = U^{-1}D^-L^{-1}, \quad \text{where } D^- = \mathrm{diag}(d_1^{-1}, d_2^{-1}, \ldots, d_{n-1}^{-1}, 0).$$

It can be easily verified that A^- satisfies the first two of Penrose's four conditions [12]:

$$AA^-A = A \;\text{ and }\; A^-AA^- = A^-.$$

The first identity states that A^- is an *inner inverse* of A and the second that A^- is an *outer inverse* of A. A generalized inverse satisfying these two conditions is called a $(1,2)$-inverse of A or an *inner-outer inverse*. Another term that is found in the literature is *reflexive inverse*; see [12]. Because A^- does not necessarily satisfy the third and fourth Penrose conditions, it is not the Moore–Penrose pseudoinverse $A^\dagger$ of A in general. Because $A^\dagger$ is obviously a $(1,2)$-inverse, this kind of generalized inverse is non-unique. Indeed, there are infinitely many such $(1,2)$-inverses in general. Each pair R, N of subspaces of $\mathbb{R}^n$ that are complements of the null space and range of A (respectively) uniquely determines a $(1,2)$-inverse $G_{N,R}$ of A with null space $\mathcal{N}(G_{N,R}) = N$ and range $\mathcal{R}(G_{N,R}) = R$; see [12]. In the case of A^- it is readily verified that

$N = \text{span}\{e_n\}$ and $R = \text{span}\{e_1, e_2, \ldots, e_{n-1}\}$ where e_i denotes the i-th unit basis vector in $\mathbb{R}^n$. It is easy to see that R is complementary to $\mathcal{N}(A)$ and N is complementary to $\mathcal{R}(A)$. The pseudoinverse $A^\dagger$ corresponds to $R = \mathcal{R}(A^T)$, $N = \mathcal{N}(A^T)$.

The $(1,2)$-inverse A^- is also different from the group inverse $A^\sharp$, in general. This can be seen from the fact that in general $AA^- \neq A^-A$, whereas the group inverse always satisfies $AA^\sharp = A^\sharp A$. Also notice that for a singular irreducible M-matrix A the $(1,2)$-inverse $A^- = U^{-1}D^-L^{-1}$ is always a nonnegative matrix, which is not true in general for either the group or the Moore–Penrose inverse.

Let now $A \approx \bar{L}\bar{D}\bar{U}$ be an incomplete LDU factorization of A, with $\bar{L} \approx L$ unit lower triangular, $\bar{U} \approx U$ unit upper triangular and $\bar{D} \approx D$ a nonsingular diagonal matrix with positive entries on the main diagonal. Then clearly

$$(\bar{L}\bar{D}\bar{U})^{-1} = \bar{U}^{-1}\bar{D}^{-1}\bar{L}^{-1} \approx A^-.$$

Hence, an ILU factorization of A yields an implicit approximation to A^- rather than to $A^\sharp$. This can also be seen from the fact that $(\bar{L}\bar{D}\bar{U})^{-1}$, like A^-, is always nonnegative, which is not true of $A^\sharp$.

It is straightfoward to check that A^-A is the oblique projector onto $R = \text{span}\{e_1, e_2, \ldots, e_{n-1}\}$ along $\mathcal{N}(A)$ and that AA^- is the oblique projector onto $\mathcal{R}(A)$ along $N = \text{span}\{e_n\}$. Therefore A^-A has eigenvalues 0 with multiplicity 1, and 1 with multiplicity $n-1$; likewise for AA^-. Hence, it makes good sense to construct preconditioners based on approximating A^- (either implicitly or explicitly), since in this case most eigenvalues of the preconditioned matrix will be clustered around 1.

Next we consider the approximate inverse preconditioner AINV; see [4], [6]. This method is based on the observation that if Z and W are matrices whose columns are A-biorthogonal, then $W^TAZ = D$, a diagonal matrix. When all the leading principal minors of A (except possibly the last one) are nonzeros, Z and W can be obtained by applying a generalized Gram–Schmidt process to the unit basis vectors $e_1, e_2, \ldots, e_n$. In this case Z and W are unit upper triangular. It follows from the uniqueness of the LDU factorization that $Z = U^{-1}$ and $W = L^{-T}$ where $A = LDU$ is the LDU factorization of A. The diagonal matrix D is the same in both factorizations. An approximate inverse in factorized form $G = \bar{Z}\bar{D}^{-1}\bar{W}^T$ can be obtained by dropping small entries in the course of the generalized Gram–Schmidt process. Similar to ILU, this incomplete inverse factorization is guaranteed to exist for nonsingular M-matrices [4]; see the next section for the singular M-matrix case. In either case, $G = \bar{Z}\bar{D}^{-1}\bar{W}^T$ is a nonnegative matrix. Hence, this preconditioner can be interpreted as a direct (explicit) approximation to the $(1,2)$-inverse A^- of A: $\bar{Z}\bar{D}^{-1}\bar{W}^T \approx A^-$.

Lastly, we take a look at sparse approximate inverse techniques based on Frobenius norm minimization; see, e.g., [16] and [14]. With this class of methods, an approximate inverse G is computed by minimizing the functional $F(X) = ||I - AX||_F$ subject to some sparsity constraints. Here $||\cdot||_F$ denotes the Frobenius matrix norm. The sparsity constraints could be imposed *a priori*, or dynamically in the course of the algorithm. In either case, it is natural to ask what kind of generalized inverse is being approximated by G when A is a singular matrix. It can be shown that the Moore–Penrose pseudoinverse $A^\dagger$ is the matrix of smallest Frobenius norm that minimizes $||I - AX||_F$; see for instance [23], page 428. Hence, in the singular case the SPAI preconditioner can be seen as a sparse approximate Moore–Penrose inverse of A. This is generally very different from the approximate $(1, 2)$-inverses obtained by either ILU or AINV. For instance, SPAI will not produce a nonnegative preconditioner in general.

In the next section we restrict our attention to the AINV preconditioner and its application to Markov chain problems.

3 The AINV method for singular matrices

The AINV preconditioner [4], [6] is based on A-biorthogonalization. This is a generalized Gram–Schmidt process applied to the unit basis vectors e_i, $1 \le i \le n$. In this generalization the standard inner product is replaced by the bilinear form $h(x, y) = x^T Ay$. This process is well defined, in exact arithmetic, if the leading principal minors of A are nonzero, otherwise some form of pivoting (row and/or column interchanges) may be needed. If A is a nonsingular M-matrix, all the leading principal minors are positive and the process is well defined with no need for pivoting. This is perfectly analogous to the LU factorization of A, and indeed in exact arithmetic the A-biorthogonalization process computes the inverses of the triangular factors of A. When A is a singular irreducible M-matrix, all the leading principal minors of A except the n-th one (the determinant of A) are positive, and the process can still be completed.

In order to obtain a sparse preconditioner, entries (fill-ins) in the inverse factors Z and W less than a given drop tolerance in magnitude are dropped in the course of the computation, resulting in an incomplete process. The stability of the incomplete process for M-matrices was analyzed in [4]. In particular, if $\bar{d}_i$ denotes the i-th pivot, i.e., the i-th diagonal entry of $\bar{D}$ in the incomplete process, then (Proposition 3.1 in [4]) $\bar{d}_i \ge d_i$ for all $i = 1, \dots, n$. Because $d_i > 0$ for an M-matrix, no pivot breakdown can occur.

Exactly the same argument applies to the case where A is an irreducible, singular M-matrix. In this case there can be no breakdown in the first $n-1$ steps of the incomplete A-biorthogonalization process, since the first $n-1$ leading principal minors are positive, and the pivots in the incomplete process cannot be smaller than the exact ones. And even if the n-th pivot $\bar{d}_n$ happened to be zero, it could simply be replaced by a positive number in order to have a nonsingular preconditioner. The argument in [4] shows that $\bar{d}_n$ must be a nonnegative number, and it is extremely unlikely that it will be exactly zero in the incomplete process.

Another way to guarantee the nonsingularity of the preconditioner is to perturb the matrix A by adding a small positive quantity to the last diagonal entry. This makes the matrix a nonsingular M-matrix, and the incomplete A-biorthogonalization process can then be applied to this slightly perturbed matrix to yield a well defined, nonsingular preconditioner. In practice, however, this perturbation is not necessary, since dropping in the factors typically has an equivalent effect (see Theorem 3 below).

The AINV preconditioner has been extensively tested on a variety of symmetric and nonsymmetric problems in conjunction with standard Krylov subspace methods like conjugate gradients (for symmetric positive definite matrices) and GMRES, Bi-CGSTAB and TFQMR (for unsymmetric problems). The preconditioner has been found to be comparable to ILU methods in terms of robustness and rates of convergence, with ILU methods being somewhat faster on average on sequential computers. The main advantage of AINV over the ILU-type methods is that its application within an iterative process only requires matrix-vector multiplies, which are much easier to vectorize and to parallelize than triangular solves.

Unfortunately, the computation of the preconditioner using the incomplete A-biorthogonalization process is inherently sequential. One possible solution to this problem, adopted in [8], is to compute the preconditioner sequentially on one processor and then to distribute the approximate inverse factors among processors in a way that minimizes communication costs while achieving good load balancing. This approach is justified in applications, like those considered in [8], in which the matrices are small enough to fit on the local memory of one processor, and where the preconditioner can be reused a number of times. In this case the time for computing the preconditioner is negligible relative to the overall costs. In the Markov chain setting, however, the preconditioner cannot be reused in general and it is imperative that set-up costs be minimized. Furthermore, Markov chain problems can be very large, and it is desirable to be able to compute the preconditioner in parallel.

4 The parallel preconditioner

In the present section we describe how to achieve a fully parallel preconditioner. The strategy used to parallelize the preconditioner construction is based on the use of graph partitioning. This approach was first proposed in [3] in the context of solving nonsingular linear systems arising from the discretization of partial differential equations.

The idea can be illustrated as follows. If p processors are available, graph partitioning can be used to decompose the adjacency graph associated with the sparse matrix A (or with $A + A^T$ if A is not structurally symmetric) in p subgraphs of roughly equal size in such a way that the number of edge cuts is approximately minimized. Nodes which are connected by cut edges are removed from the subgraphs and put in the *separator set*. By numbering the nodes in the separator set last, a symmetric permutation Q^TAQ of A is obtained. The permuted matrix has the following structure:

$$Q^TAQ = \begin{pmatrix} A_1 & & & & B_1 \\ & A_2 & & & B_2 \\ & & \ddots & & \vdots \\ & & & A_p & B_p \\ C_1 & C_2 & \dots & C_p & A_S \end{pmatrix}.$$

The diagonal blocks $A_1, \dots, A_p$ correspond to the interior nodes in the graph decomposition, and should have approximately the same order. The off-diagonal blocks B_i, C_i represent the connections between the subgraphs, and the diagonal block A_S the connections between nodes in the separator set. The order of A_S is equal to the cardinality of the separator set and should be kept as small as possible. Note that because of the irreducibility assumption, each block A_i is a nonsingular M-matrix, and each of the LDU factorizations $A_i = L_iD_iU_i$ exists, $1 \le i \le p$.

Let $Q^TAQ = LDU$ be the LDU factorization of Q^TAQ. Then it is easy to see that

$$L^{-1} = \begin{pmatrix} L_1^{-1} & & & & \\ & L_2^{-1} & & & \\ & & \ddots & & \\ & & & L_p^{-1} & \\ F_1 & F_2 & \dots & F_p & L_S^{-1} \end{pmatrix}$$

where $F_i := -L_S^{-1}C_iA_i^{-1}$ $(1 \le i \le p)$ and L_S^{-1} is the inverse of the unit lower triangular factor of the Schur complement matrix

$$S = A_S - \sum_{i=1}^{p} C_iA_i^{-1}B_i.$$

In the next section we show that S is a singular, irreducible M-matrix, hence it has a well defined LDU factorization $S = L_S D_S U_S$.

Likewise,

$$U^{-1} = \begin{pmatrix} U_1^{-1} & & & & E_1 \\ & U_2^{-1} & & & E_2 \\ & & \ddots & & \vdots \\ & & & U_p^{-1} & E_p \\ & & & & U_S^{-1} \end{pmatrix}$$

where $E_i := -A_i^{-1} B_i U_S^{-1}$ $(1 \leq i \leq p)$ and U_S^{-1} is the inverse of the unit upper triangular factor of S. It is important to observe that L^{-1} and U^{-1} preserve a good deal of sparsity, since fill-in can occur only within the nonzero blocks. The matrix D is simply defined as

$$D = \text{diag}(D_1,\ D_2, \ldots, D_p,\ D_S);$$

note that all diagonal entries of D are positive except for the last one, which is zero. The $(1,2)$-inverse D^- of D is defined in the obvious way.

Hence, we can write the (generally dense) generalized inverse $(Q^T AQ)^-$ of $Q^T AQ$ as a product of sparse matrices L^{-1}, U^{-1} and D^-. In practice, however, the inverse factors L^{-1} and U^{-1} contain too many nonzeros. Since we are only interested in computing a preconditioner, we just need to compute sparse approximations to L^{-1} and U^{-1}.

This is accomplished as follows. With graph partitioning, the matrix is distributed so that processor P_i holds A_i, C_i and B_i $(1 \leq i \leq p)$. One of the processors, marked as P_S, should also hold A_S. Each processor then computes sparse approximate inverse factors $\bar{Z}_i$, $\bar{D}_i$ and $\bar{W}_i$ such that $\bar{Z}_i \bar{D}_i^{-1} \bar{W}_i^T \approx A_i^{-1}$ using the AINV algorithm. Once this is done, each processor computes the product $S_i := C_i \bar{Z}_i \bar{D}_i^{-1} \bar{W}_i^T B_i \approx C_i A_i^{-1} B_i$. Until this point the computation proceeds in parallel with no communication. The next step is the accumulation of the approximate Schur complement $\hat{S} := A_S - \sum_{i=1}^{p} S_i$ on processor P_S. This accumulation is done in $k = \log_2 p$ steps with a fan-in across the processors. In the next section we show that although the exact Schur complement S is singular, the approximate Schur complement $\hat{S}$ is a nonsingular M-matrix under rather mild conditions.

As soon as $\hat{S}$ is computed, processor P_S computes a factorized sparse approximate inverse $\bar{Z}_S \bar{D}_S^{-1} \bar{W}_S^T \approx \hat{S}^{-1}$ using the AINV algorithm. This is a sequential bottleneck, and explains why the size of the separator set must be kept small. Once the approximate inverse factors of $\hat{S}$ are computed, they are broadcast to all remaining processors. (Actually, the preconditioner application can be implemented in such a way that only the $\bar{W}_S$ factor needs to be broadcast.) Notice that because only matrix-vector products are required in the application of the preconditioner, there is no need to form $-\bar{Z}_i \bar{D}_i^{-1} \bar{W}_i^T B_i \bar{Z}_S \approx E_i$ or

$-\bar{W}_S^T C_i \bar{Z}_i \bar{D}_i^{-1} \bar{W}_i^T \approx F_i$ explicitly. In this way, a factorized sparse approximate $(1,2)$-inverse of $Q^T AQ$ is obtained.

This is a two-level preconditioner, in the sense that the computation of the preconditioner involves two phases. In the first phase, sparse approximate inverses of the diagonal blocks A_i are computed. In the second phase, a sparse approximate inverse of the approximate Schur complement $\hat{S}$ is computed. Without this second step the preconditioner would reduce to a block Jacobi method with inexact block solves (in the terminology of domain decomposition methods, this is additive Schwarz with inexact solves and no overlap). It is well known that for a fixed problem size, the rate of convergence of this preconditioner tends to deteriorate as the number of blocks (subdomains) grows. Hence, assuming that each block is assigned to a processor in a parallel computer, this method would not be scalable. However, the approximate Schur complement phase provides a global exchange of information across the processors, acting as a "coarse grid" correction in which the "coarse grid" nodes are interface nodes (i.e., they correspond to vertices in the separator set). As we will see, this prevents the number of iterations from growing as the number of processors grows. As long as the cardinality of the separator set is small compared to the cardinality of the subdomains (subgraphs), the algorithm is scalable in terms of parallel efficiency. Indeed, in this case the application of the preconditioner at each step of a Krylov subspace method like GMRES or Bi-CGSTAB is easily implemented in parallel with relativeley little communication needed.

5 The approximate Schur complement

In this section we investigate the existence of the approximate $(1,2)$-inverse of the generator matrix A. The key role is played by the (approximate) Schur complement.

First we briefly review the situation for the case where A is a nonsingular M-matrix. Assume A is partitioned as

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}. \tag{1}$$

Then it is well-known that the Schur complement

$$S = A_{22} - A_{21} A_{11}^{-1} A_{12}$$

is also a nonsingular M-matrix; see, e.g., [1]. Moreover, the same is true of any approximate Schur complement

$$\hat{S} = A_{22} - A_{21}X_{11}A_{12}$$

provided that $O \le X_{11} \le A_{11}^{-1}$, where the inequalities hold componentwise; see [1], page 264.

In the singular case, the situation is slightly more complicated. In the following we will examine some basic properties of the exact Schur complement of a singular, irreducible M-matrix A corresponding to an ergodic Markov chain. Recall that $A = I - P^T$ where P is the irreducible row-stochastic transition probability matrix.

Lemma 1 *Let P be an irreducible row-stochastic matrix partitioned as*

$$\begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}.$$

Assume that

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} I - P_{11}^T & -P_{21}^T \\ -P_{12}^T & I - P_{22}^T \end{pmatrix}.$$

Then the Schur complement S of A_{11} in A is a singular, irreducible M-matrix with a one-dimensional null space.

Proof: Consider the stochastic complement [22] Σ of P_{22} in P:

$$\Sigma = P_{22} + P_{21}(I - P_{11})^{-1}P_{12}.$$

Note that $I - P_{11}$ is invertible since P is irreducible. From the theory developed in [22], we know that Σ is row stochastic and irreducible since P is. Consider now the Schur complement S of A_{11} in A:

$$S = A_{22} - A_{21}A_{11}^{-1}A_{12} = (I - P_{22}^T) - P_{12}^T(I - P_{11}^T)^{-1}P_{21}^T = I - \Sigma^T.$$

Clearly, S is an irreducible singular M-matrix. It follows (Perron–Frobenius theorem) that S has a one-dimensional null space. □

The previous lemma is especially useful in cases where the exact Schur complement is used. In the context of preconditioning it is often important to know properties of approximate Schur complements. As shown in the previous section, graph partitioning induces a reordering and block partitioning of the matrix A in the form (1) where

$$A_{11} = \mathrm{diag}(A_1,\ A_2, \ldots, A_p), \quad A_{22} = A_S$$

and

$$A_{12} = [B_1^T\ B_2^T\ \ldots\ B_p^T]^T, \quad A_{21} = [C_1\ C_2\ \ldots\ C_p].$$

We are interested in properties of the approximate Schur complement $\hat{S}$ obtained by approximating the inverses of the diagonal blocks A_i with AINV:

$$\hat{S} = A_S - \sum_{i=1}^{p} C_i(\bar{Z}_i \bar{D}_i^{-1} \bar{W}_i^T) B_i. \tag{2}$$

In particular, we are interested in conditions that guarantee that $\hat{S}$ is a nonsingular M-matrix, in which case the AINV algorithm can be safely applied to $\hat{S}$, resulting in a well defined preconditioner. We begin with a lemma. Recall that a Z-matrix is a matrix with nonpositive off-diagonal entries [9].

Lemma 2 *Let S be a singular, irreducible M-matrix. Let $C \geq O$, $C \neq O$ be such that $\hat{S} = S + C$ is a Z-matrix. Then $\hat{S}$ is a nonsingular M-matrix.*

Proof: Since S is a singular M-matrix, we have $S = \rho(B)I - B$ with $B \geq O$, where $\rho(B)$ denotes the spectral radius of B. Let $D = \mathrm{diag}(C)$. Since $S + C$ is a Z-matrix, we have that $B \geq C - D$. Therefore we can write the modified matrix $\hat{S}$ as $\hat{S} = D + \rho(B)I - (B + D - C)$ with $B + D - C \geq O$.

We distinguish the two following simple cases: Assume first that $D = O$. Then $\hat{S}$ is a nonsingular M-matrix since it can be written as $\rho(B)I - (B - C)$ with $\rho(B) > \rho(B - C)$. The last inequality follows from the irreducibility of B and properties of nonnegative matrices; see [9], page 27, Cor. 1.5 (b). Assume, on the other hand, that $C = D$. Hence, $\hat{S} = D + \rho(B)I - B$. Note that by assumption, at least one of the diagonal entries d_{ii} of D must be positive. Let δ denote the largest such positive diagonal entry. Since $B + \delta I$ is irreducible, $\rho(B + \delta I - D) < \rho(B + \delta I) = \rho(B) + \delta$, similar to the previous case. It follows that $\hat{S} = (\rho(B) + \delta)I - (B + \delta I - D)$ is a nonsingular M-matrix.

Finally, if both $D \neq O$ and $C \neq D$ the result follows by combining the two previous arguments. $\square$

In the context of our parallel preconditioner, this lemma says that if the inexactness in the approximate inverses of the diagonal blocks A_i results in an approximate Schur complement $\hat{S}$ that is still a Z-matrix and furthermore if $\hat{S} - S$ is nonnegative and nonzero, then $\hat{S}$ is a nonsingular M-matrix.

The following theorem states sufficient conditions for the nonsingularity of the approximate Schur complement. B_{i*} and B_{*j} denote the i-th row and the j-th column of matrix B, respectively.

Theorem 3 *Let*

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

be a singular irreducible M-matrix, with $A_{11} \in \mathbb{R}^{m\times m}$. Assume that $\tilde{A}_{11}^{-1}$ is an approximation to A_{11}^{-1} such that $O \leq \tilde{A}_{11}^{-1} \leq A_{11}^{-1}$, $\tilde{A}_{11}^{-1} \neq A_{11}^{-1}$. Furthermore, assume that there exist indices $i, j \in \{1, \ldots, m\}$ such that the following three conditions are satisfied: $(A_{11}^{-1} - \tilde{A}_{11}^{-1})_{ij} \neq 0$, $(A_{21})_{i*} \neq 0$ and $(A_{12})_{*j} \neq 0$. Then the approximate Schur complement $\hat{S} = A_{22} - A_{21}\tilde{A}_{11}^{-1}A_{12}$ is a nonsingular M-matrix.

Proof: From Lemma 1 we know that the exact Schur complement S of A_{11} in A is a singular, irreducible M-matrix. Note that the approximate Schur complement $\hat{S}$ induced by an approximation $\tilde{A}_{11}^{-1}$ of the block A_{11}^{-1} and the exact Schur complement S are related as follows:

$$\hat{S} = A_{22} - A_{21}\tilde{A}_{11}^{-1}A_{12} = S + A_{21}(A_{11}^{-1} - \tilde{A}_{11}^{-1})A_{12} = S + C$$

with $C \geq O$ since $A_{21} \leq O, A_{12} \leq O$, and $A_{11}^{-1} - \tilde{A}_{11}^{-1} \geq O$. Clearly, $\hat{S}$ is a Z-matrix by its definition. From the assumption that $\tilde{A}_{11}^{-1} \neq A_{11}^{-1}$ and that there exists at least one nonzero entry α_{ij} of $\tilde{A}_{11}^{-1}$ not equal to (in fact, strictly less than) the corresponding entry of A_{11}^{-1}, we see that if the corresponding row i of A_{21} and column j of A_{12} are both nonzero, there is a nonzero entry in C. The result now follows from Lemma 2. □

Let us now apply these results to the preconditioner described in the previous section. In this case, A_{11} is block diagonal and the AINV algorithm is used to approximate the inverse of each diagonal block separately, in parallel. The approximate Schur complement (2) is the result of subtracting p terms of the form $C_i(\bar{Z}_i\bar{D}_i^{-1}\bar{W}_i^T)B_i$ from A_S, $1 \leq i \leq p$. We refer to these terms as (Schur complement) *updates*. Each one of these updates is nonnegative and approximates the exact update $C_iA_i^{-1}B_i$ from below in the (entrywise) nonnegative ordering, since $O \leq \bar{Z}_i\bar{D}_i^{-1}\bar{W}_i^T \leq A_i^{-1}$; see [6]. Theorem 3 says that as long as at least one of these updates has an entry that is strictly less than the corresponding entry in $C_iA_i^{-1}B_i$, the approximate Schur complement $\hat{S}$ is a nonsingular M-matrix.

In practice, these conditions are satisfied as a result of dropping in the approximate inversion of the diagonal blocks A_i. It is nevertheless desirable to have rigorous conditions that ensure nonsingularity. The following theorem gives a sufficient condition for having a nonsingular approximate Schur complement as a consequence of dropping in AINV. Namely, it specifies conditions under which any dropping forces $\hat{S}$ to be nonsingular. Note that the conditions of this theorem do not apply to the global matrix (1), since A_{11} is block diagonal and therefore reducible. However, they can be applied to any individual Schur complement update for which the corresponding diagonal block A_i is irreducible, making the result fairly realistic.

Theorem 4 *Let $A_{11} \in \mathbb{R}^{m \times m}$ and the singular M-matrix*

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

be both irreducible. Let $A_{11} = L_{11}U_{11}$ be the LU factorization of A_{11}. Assume that in each column of L_{11} (except the last one) and in each row of U_{11} (except the last one) there is at least one nonzero entry in addition to the diagonal one:

$$(\forall i \in \{1, \ldots, m-1\})(\exists j > i)((L_{11})_{ji} \neq 0) \tag{3}$$

and

$$(\forall i \in \{1, \ldots, m-1\})(\exists j > i)((U_{11})_{ij} \neq 0). \tag{4}$$

Denote by $\bar{Z}_{11}\bar{D}_{11}^{-1}\bar{W}_{11}^T \approx A_{11}^{-1}$ the factorized sparse approximate inverse of A_{11} obtained with the AINV algorithm. Then the approximate Schur complement $\hat{S}$ is a nonsingular M-matrix provided that $\bar{Z}_{11} \neq Z_{11}$ and $\bar{W}_{11} \neq W_{11}$.

Proof: First note that the two conditions for nonzero entries in L_{11} and U_{11} are implied by similar conditions for the entries in the lower and upper triangular parts of A_{11}. Namely, it is easy to see that (barring fortuitous cancellation)

$$\text{for } i > j: \quad \text{tril}(A_{11})_{ij} \neq 0 \Rightarrow (L_{11})_{ij} \neq 0$$

$$\text{and for } i < j: \quad \text{triu}(A_{11})_{ij} \neq 0 \Rightarrow (U_{11})_{ij} \neq 0.$$

Here tril(B) and triu(B) denote the lower and upper triangular part of matrix B, respectively. These conditions are easier to check than the weaker ones on the triangular factors of A_{11}. Conditions (3) and (4) imply that there is a path $i \to m$ in the graph of L_{11}^T and there is a path $i \to m$ in the graph of U_{11} for all $i \in \{1, \ldots, m-1\}$.

Because the matrix is irreducible, then $A_{12} \neq O$ and $A_{21} \neq O$. This means that there exist indices i, j, r, s such that $(A_{12})_{ij} \neq 0$ and $(A_{21})_{rs} \neq 0$. The existence of the previously mentioned paths implies that

$$(W_{11}^T A_{12})_{m*} \neq 0 \text{ and } (A_{21}Z_{11})_{*m} \neq 0,$$

where W_{11} and Z_{11} are the exact inverse factors of A_{11}. The approximate inverse factors from the AINV algorithm satisfy [6]

$$O \leq \bar{Z}_{11}\bar{D}_{11}^{-1}\bar{W}_{11}^T \leq A_{11}^{-1}.$$

Hence, the conditions of Theorem 3 are satisfied and the result is proved. □

It is instructive to consider two extreme cases. If A_{11} is diagonal, then the approximate Schur complement is necessarily equal to the exact one, and is therefore singular. In this case, of course, the conditions of the last theorem are violated. On the other hand if A_{11} is irreducible and tridiagonal, its inverse factors are completely dense and by the last theorem it is enough to drop a single entry in each inverse factor to obtain a nonsingular approximate Schur complement.

The purpose of the theory developed here is to shed light on the observed robustness of the proposed preconditioner rather than to serve as a practical tool. In other words, it does not seem to be necessary to check these conditions in advance. Indeed, thanks to dropping the approximate Schur complement was always found to be a nonsingular M-matrix in actual computations.

6 Numerical experiments

In this section we report on results obtained with a parallel implementation of the preconditioner on several Markov chain problems. The underlying Krylov subspace method was Bi-CGSTAB [27], which was found to perform well for Markov chains in [15]. Our FORTRAN implementation uses MPI and dynamic memory allocation. The package METIS [18] was used for the graph partitioning, working with the graph of $A + A^T$ whenever A was not structurally symmetric.

The test problems arise from real Markov chain applications and were provided by T. Dayar. These matrices have been used in [15] to compare different methods in a sequential environment. A description of the test problems is provided in Table 1 below. Here n is the problem size and nnz the number of nonzeros in the matrix. All the test problems are structurally nonsymmetric except *ncd* and *mutex*. Most matrices are unstructured.

Tables 2–11 contain the test results. All runs were performed on an SGI Origin 2000 at Los Alamos National Laboratory (using up to 64 processors), except for those with matrices *leaky*, *ncd* and $2d$ which were performed on an Origin 2000 at the Helsinki University of Technology (using up to 8 processors). In all cases, the initial guess was a constant nonzero vector; similar results were obtained with a randomly generated initial guess. In the tables, "P-time" denotes the time to compute the preconditioner, "P-density" the ratio of the number of nonzeros in the preconditioner to the number of nonzeros in the matrix A, "Its" denotes the number of iterations needed to reduce the ℓ^2-norm of the initial residual by eight orders of magnitude, "It-time" the time to perform the iterations, and "Tot-time" the sum of "P-time" and "It-time." All timings are in seconds. Furthermore, "Sep-size" is the cardinality of the

Table 1
Information on test problems.

Matrix	n	nnz	Application
hard	20301	140504	Complete buffer sharing in ATM networks
leaky	8258	197474	Multiplexing model of a leaky bucket
$2d$	16641	66049	A two-dimensional Markov chain model
telecom	20491	101041	A telecommunication model
ncd	23426	156026	NCD queueing network
mutex	39203	563491	Resource sharing model
qn	104625	593115	A queueing network

separator set (i.e., the order of the Schur complement matrix) and "Avg-dom" the average number of vertices in a subdomain (subgraph) in the graph partitioning of the problem. The drop tolerance τ in the AINV algorithm was the same at both levels of the preconditioner (approximate inversion of A_i for $1 \leq i \leq p$ and approximate inversion of the approximate Schur complement $\hat{S}$), except for the *mutex* problem (see below).

Tables 2–4 present results for the matrix *hard*, using three different values of the drop tolerance in the AINV algorithm. It can be seen that changing the value of τ changes the density of the preconditioner and the number of iterations. However, the total timings are scarcely affected, especially if at least 8 processors are being used. See [8] for a similar observation in a different context. It is also clear from these runs that good speed-ups are obtained so long as the size of the separator set is small compared to the average subdomain size. As soon as the separator set is comparable in size to the average subdomain or larger, the sequential bottleneck represented by the Schur complement part of the computation begins to dominate and performance deteriorates. The number of iterations remains roughly constant (with a slight downward trend) as the number of processors grows. This is due to the influence of the approximate Schur complement.

The same problem was also solved using Bi-CGSTAB with diagonal preconditioning. This required approximately 700 iterations and 16.4 seconds on one processor. If implemented in parallel, this method would probably give results only slightly worse than those obtained with AINV. A similar observation applies to matrices *qn* and *mutex*. On the other hand, diagonally preconditioned Bi-CGSTAB did not converge on the *telecom* problem. Hence, AINV is a more robust approach. Furthermore, the ability to reduce the number of iterations, and therefore the total number of inner products, is an advantage on distributed memory machines, on which inner products incur an additional penalty due to the need for global communication.

Table 2
Matrix *hard*, $\tau = 0.02$

p	2	4	8	16	32
P-time	2.35	1.19	0.65	0.40	0.34
P-density	6.21	5.90	5.67	5.14	4.52
Its	66	67	65	62	64
It-time	9.43	2.91	1.42	0.99	0.88
Tot-time	11.8	4.10	2.07	1.39	1.22
Sep-size	156	321	540	900	1346
Avg-dom	10073	5155	2470	1213	592

Table 3
Matrix *hard*, $\tau = 0.05$

p	2	4	8	16	32
P-time	1.19	0.60	0.35	0.22	0.21
P-density	3.10	3.02	2.95	2.78	2.52
Its	106	109	99	98	97
It-time	6.85	2.90	1.59	1.05	1.19
Tot-time	8.04	3.50	1.94	1.27	1.40

Table 4
Matrix *hard*, $\tau = 0.1$

p	2	4	8	16	32
P-time	0.73	0.38	0.22	0.15	0.14
P-density	1.39	1.36	1.34	1.28	1.19
Its	170	167	159	151	153
It-time	6.03	2.52	1.50	1.30	1.64
Tot-time	6.76	2.90	1.72	1.45	1.78

Results for matrices *leaky* and *2d* are reported in Tables 5 and 6. These two matrices are rather small, so only up to 8 processors were used. Note that the speed-ups are better for *2d* than for *leaky*. Also notice that the preconditioner is very sparse for *leaky*, but rather dense for *2d*.

Tables 7 and 8 refer to the *telecom* test problem. Here we found that very small values of τ (and, consequently, very dense preconditioners) are necessary in order to achieve convergence in a reasonable number of iterations. This problem is completely different from the matrices arising from the solution of elliptic partial differential equations. Notice the fairly small size of the sep-

Table 5
Matrix *leaky*, $\tau = 0.1$

p	2	4	8
P-time	0.26	0.17	0.09
P-density	0.21	0.21	0.20
No. its	134	134	132
It-time	1.39	0.84	0.62
Tot-time	1.65	1.01	0.71
Sep-size	48	144	335
Avg-dom	4105	2028	990

Table 6
Matrix $2d$, $\tau = 0.1$

p	2	4	8
P-time	0.38	0.16	0.09
P-density	8.60	7.12	6.54
No. its	33	36	37
It-time	1.46	0.56	0.38
Tot-time	1.84	0.72	0.47
Sep-size	129	308	491
Avg-dom	8256	4083	2018

Table 7
Matrix *telecom*, $\tau = 0.001$

p	2	4	8	16	32	64
P-time	24.9	10.0	3.35	1.21	1.05	1.37
P-density	167	116	70	44	28	16
Its	11	12	14	13	12	12
It-time	36.5	14.0	6.04	0.96	0.38	0.43
Tot-time	61.4	24.0	9.39	2.17	1.43	1.80
Sep-size	34	97	220	471	989	1603
Avg-dom	10229	5099	2534	1251	609	295

arator set, which causes the density of the preconditioner to decrease very fast as the number of processors (and corresponding subdomains) grows. As a result, speed-ups are quite good (even superlinear) up to 32 processors. For a sufficiently high number of processors, the density of the preconditioner be-

Table 8
Matrix *telecom*, $\tau = 0.005$

p	2	4	8	16	32	64
P-time	7.04	3.78	1.37	0.61	0.49	0.56
P-density	46	44	37	29	21	10
No. its	60	55	81	60	68	84
It-time	59.4	27.0	12.8	2.18	1.36	1.71
Tot-time	66.4	30.8	14.2	2.79	1.86	2.27

Table 9
Matrix *ncd*, $\tau = 0.1$

p	2	4	8
P-time	1.42	0.69	0.31
P-density	4.13	2.65	1.90
No. its	292	288	285
It-time	17.0	8.45	6.38
Tot-time	18.4	9.14	6.69
Sep-size	3911	6521	12932
Avg-dom	9758	4226	1379

Table 10
Matrix *mutex*, $\tau = 0.1$

p	2	4	8
P-time	1.19	0.40	0.10
P-density	0.14	0.14	0.14
No. its	16	14	15
It-time	1.64	1.19	1.51
Tot-time	2.83	1.59	1.61
Sep-size	13476	17749	20654
Avg-dom	12864	5363	2319

comes acceptable, and the convergence rate is the same or comparable to that obtained with a very dense preconditioner on a small number of processors.

Tables 9 and 10 give results for matrices *ncd* and *mutex*, respectively. For the first matrix we see that the separator set is larger than the average subdomain already for $p = 4$ subdomains; nevertheless, it is possible to use effectively up to 8 processors. Matrix *mutex* exhibits a behavior that is radically different

Table 11
Matrix qn, $\tau = 0.1$

p	2	4	8	16
P-time	4.12	2.37	2.26	2.82
P-density	1.27	1.23	1.18	1.14
No. its	49	49	48	45
It-time	13.4	6.33	4.58	5.68
Tot-time	17.5	8.70	6.84	8.50
Sep-size	2879	6579	13316	20261
Avg-dom	50873	24511	11414	5273

from that of matrices arising from PDE's in two or three space dimensions. The separator set is huge already for $p = 2$. This is due to the fact that the problem has a state space (graph) of high dimensionality, leading to a very unfavorable surface-to-volume ratio in the graph partitioning. In order to solve this problem, we had to use two different values of τ in the two levels of AINV; at the subdomain level we used $\tau = 0.1$, but when forming the approximate Schur complement we dropped everything outside the main diagonal, resulting in a diagonal $\hat{S}$. In spite of this, convergence was very rapid. Nevertheless, it does not pay to use more than $p = 4$ processors.

In Table 11 we report results with the largest example in our data set, qn. This model consists of a network of three queues, and is analogous to a three-dimensional PDE problem. Because of the fairly rapid growth of the separator set, it does not pay to use more than $p = 8$ processors.

The test problems considered so far, although realistic, are relatively small. Hence, it is difficult to make efficient use of more than 16 processors, with the partial exceptions of matrices *hard* and *telecom*. To test the scalability of the proposed solver on larger problems, we generated some simple reliability problems analogous to those used in [2] and [5]; see also [26], page 135. These problems have a closed form solution. In Table 12 we show timing results for running 100 preconditioned Bi-CGSTAB iterations on a reliability problem of size $n = 250,000$ with $1,248,000$ nonzero entries. This problem is sufficiently large to show the good scalability of the algorithm up to $p = 64$ processors.

We conclude this section on numerical experiments by noting that in virtually all the runs, the preconditioner construction time has been quite modest and the total solution time has been dominated by the cost of the iterative phase.

Table 12
Reliability model, $\tau = 0.1$

p	2	4	8	16	32	64
P-time	9.53	4.86	2.49	1.31	0.90	0.90
P-density	4.05	4.02	3.97	3.90	3.80	3.70
It-time	138.8	70.5	37.2	16.8	9.47	7.96
Tot-time	148.3	75.4	39.7	18.1	10.4	8.86
Sep-size	542	1229	2186	3268	4919	7336
Avg-dom	124729	62193	30977	15421	7659	3792

7 Conclusions

We have investigated the use of a parallel preconditioner for Krylov subspace methods in the context of Markov chain problems. The preconditioner is a direct approximation, in factorized form, of a $(1, 2)$-inverse of the generator matrix A, and is based on an A-biorthogonalization process. Parallelization is achieved through graph partitioning, although other approaches are also possible. The existence of the preconditioner has been justified theoretically, and numerical experiments on a parallel computer have been carried out in order to assess the effectiveness and scalability of the proposed technique. The numerical tests indicate that the preconditioner construction costs are modest, and that good scalability is possible provided that the amount of work per processor is sufficiently large compared to the size of the separator set.

The method appears to be well suited for problems in which the generator matrix $A = I - P^T$ can be explicitly formed and stored. Parallelization based on graph partitioning is usually effective, with the possible exception of problems with a state space of high dimensionality (i.e., a large state descriptor set). For such problems, a different parallelization strategy is needed in order to achieve scalability of the implementation.

Acknowledgements. We would like to thank Professors M. Gutknecht and W. Schönauer for their kind invitation to take part in this commemoration of our friend and colleague Rüdiger Weiss. We are indebted to Tuğrul Dayar for providing the test matrices used in the numerical experiments and for useful information about these problems, as well as for his comments on an early version of the paper. Thanks also to Carl Meyer for his valuable input on generalized inverses.

References

[1] O. Axelsson, *Iterative Solution Methods* (Cambridge University Press, Cambridge, 1994).

[2] M. Benzi and T. Dayar, The arithmetic mean method for finding the stationary vector of Markov chains, *Parallel Algorithms Appl.* 6 (1995) 25–37.

[3] M. Benzi, J. Marín and M. Tůma, A two-level parallel preconditioner based on sparse approximate inverses, in: D. R. Kincaid and A. C. Elster, eds., *Iterative Methods in Scientific Computation IV* (IMACS Series in Computational and Applied Mathematics, vol. 5, IMACS, New Brunswick, NJ, 1999) 165–178.

[4] M. Benzi, C. D. Meyer and M. Tůma, A sparse approximate inverse preconditioner for the conjugate gradient method, *SIAM J. Sci. Comput.* 17 (1996) 1135–1149.

[5] M. Benzi, F. Sgallari and G. Spaletta, A parallel block projection method of the Cimmino type for finite Markov chains, in: W. J. Stewart, ed., *Computations with Markov Chains* (Kluwer Academic Publishers, Boston/London/Dordrecht, 1995) 65–80.

[6] M. Benzi and M. Tůma, A sparse approximate inverse preconditioner for nonsymmetric linear systems, *SIAM J. Sci. Comput.* 19 (1998) 968–994.

[7] M. Benzi and M. Tůma, A comparative study of sparse approximate inverse preconditioners, *Appl. Numer. Math.* 30 (1999) 305–340.

[8] L. Bergamaschi, G. Pini and F. Sartoretto, Approximate inverse preconditioning in the parallel solution of sparse eigenproblems, *Numer. Linear Algebra Appl.* 7 (2000) 99–116.

[9] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences* (Academic Press, New York, 1979. Reprinted by SIAM, Philadelphia, PA, 1994).

[10] P. Buchholz, M. Fischer and P. Kemper, Distributed steady state analysis using Kronecker algebra, in: B. Plateau, W. J. Stewart and M. Silva, eds., *Numerical Solutions of Markov Chains (NSMC'99)* (Prensas Universitarias de Zaragoza, Zaragoza, Spain, 1999) 76–95.

[11] J. J. Buoni, Incomplete factorization of singular M-matrices, *SIAM J. Alg. Discrete Methods* 7 (1986) 193–198.

[12] S. L. Campbell and C. D. Meyer, *Generalized Inverses of Linear Transformations* (Pitman Publishing Ltd., London and San Francisco, 1979. Reprinted by Dover Publishing Co., New York, 1991).

[13] R. Choquet and V. Perrier, Fast wavelet iterative solvers applied to the Neumann problem, *Aerosp. Sci. Technol.* 4 (2000) 135–145.

[14] E. Chow, A priory sparsity patterns for parallel sparse approximate inverse preconditioners, *SIAM J. Sci. Comput.* 21 (2000) 1804–1822.

[15] T. Dayar and W. J. Stewart, Comparison of partitioning techniques for two-level iterative solvers on large, sparse Markov chains, *SIAM J. Sci. Comput.* 21 (2000) 1691–1705.

[16] M. J. Grote and T. Huckle, Parallel preconditioning with sparse approximate inverses, *SIAM J. Sci. Comput.* 18 (1997) 838–853.

[17] M. Jarraya and D. El Baz, Asynchronous iterations for the solution of Markov systems, in: B. Plateau, W. J. Stewart and M. Silva, eds., *Numerical Solutions of Markov Chains (NSMC'99)* (Prensas Universitarias de Zaragoza, Zaragoza, Spain, 1999) 335–338.

[18] G. Karypis and V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, *SIAM J. Sci. Comput.* 20 (1999) 359–392.

[19] W. J. Knottenbelt and P. G. Harrison, Distributed disk-based solution techniques for large Markov models, in: B. Plateau, W. J. Stewart and M. Silva, eds., *Numerical Solutions of Markov Chains (NSMC'99)* (Prensas Universitarias de Zaragoza, Zaragoza, Spain, 1999) 58–75.

[20] J. A. Meijerink and H. A. van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Math. Comp.* 31 (1977) 148–162.

[21] C. D. Meyer, The role of the group generalized inverse in the theory of finite Markov chains, *SIAM Rev.* 17 (1975) 443–464.

[22] C. D. Meyer, Stochastic complementation, uncoupling Markov chains, and the theory of nearly reducible systems, *SIAM Rev.* 31 (1989) 240–272.

[23] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra* (SIAM, Philadelphia, PA, 2000).

[24] V. Migallón, J. Penadés and D. B. Szyld, Experimental studies of parallel iterative solutions of Markov chains with block partitions, in: B. Plateau, W. J. Stewart and M. Silva, eds., *Numerical Solutions of Markov Chains (NSMC'99)* (Prensas Universitarias de Zaragoza, Zaragoza, Spain, 1999) 96–110.

[25] Y. Saad, Preconditioned Krylov subspace methods for the numerical solution of Markov chains, in: W. J. Stewart, ed., *Computations with Markov Chains* (Kluwer Academic Publishers, Boston/London/Dordrecht, 1995) 49–64.

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

[27] H. A. van der Vorst, BiCGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.* 13 (1992) 631–644.