Technical Report

TR-2006-001

Product Preconditioning for Markov Chain Problems

by

Michele Benzi, Bora Ucar

Mathematics and Computer Science

EMORY UNIVERSITY

PRODUCT PRECONDITIONING FOR MARKOV CHAIN PROBLEMS

MICHELE BENZI[§*] AND BORA UÇAR[§†]

Abstract. We consider preconditioned Krylov subspace methods for computing the stationary probability distribution vector of irreducible Markov chains. We propose preconditioners constructed as the product of two fairly simple preconditioners. Theoretical properties of the proposed product preconditioners are briefly discussed. We use graph partitioning tools to partition the coefficient matrix in order to build the preconditioner matrices, and we investigate the effect of the partitioning on the proposed preconditioners. Numerical experiments with GMRES on various Markov chain problems generated with the MARCA software package demonstrate that the proposed preconditioners are effective in reducing the number of iterations to convergence. Furthermore, the experimental results show that the number of partitions does not severely affect the number of iterations.

Key words. preconditioning, discrete Markov chains, iterative methods, graph partitioning

AMS subject classifications. 05C50, 60J10, 60J22, 65F10, 65F50, 65F35

1. Introduction. Discrete Markov chains with large state spaces arise in many applications, including for instance reliability modeling, queuing network analysis, web-based information retrieval, and computer system performance evaluation [29]. As is well known, the long-run behavior of an ergodic (irreducible) Markov chain is described by the stationary distribution vector of the corresponding matrix of transition probabilities. Recall that the stationary probability distribution vector of a finite, ergodic Markov chain with $N \times N$ transition probability matrix P is the unique $1 \times N$ vector π which satisfies

$$\pi = \pi P, \quad \pi_i > 0 \ \text{ for } \ i = 1, \ldots N, \quad \sum_{i=1}^{N} \pi_i = 1 \ . \tag{1.1}$$

Here P is nonnegative ($p_{ij} \geq 0$ for $1 \leq i, j \leq N$), row-stochastic ($\sum_{j=1}^{N} p_{ij} = 1$ for $1 \leq i \leq N$), and due to ergodicity assumption it is irreducible.

The matrix $A = I - P^T$, where I is the $N \times N$ identity matrix, is called the generator of the Markov process. The matrix A is a singular, irreducible M-matrix of rank $N-1$. Letting $x = \pi^T$ and hence $x^T = x^T P$, the computation of the stationary vector reduces to finding a nontrivial solution to the homogeneous linear system

$$Ax = 0 \ , \tag{1.2}$$

where $x \in \mathbb{R}^N$, $x_i > 0$ for $i = 1, \ldots, N$, and $\sum_{i=1}^{N} x_i = 1$. Perron–Frobenius theory [7] implies that such a vector exists and is unique. We assume that P is large and sparse; hence, so is A. We further assume that P and A are partitioned as

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \text{ and } \quad A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} I_n - P_{11}^T & -P_{21}^T \\ -P_{12}^T & I_m - P_{22}^T \end{bmatrix} . \tag{1.3}$$

*The work of this author was supported in part by the National Science Foundation grant DMS-0511336.

†The work of this author was supported by The Scientific and Technological Research Council of Turkey (TÜBITAK).

§Department of Mathematics and Computer Science, Emory University, Atlanta, Georgia 30322, USA (benzi@mathcs.emory.edu, ubora@mathcs.emory.edu).

Here I_n and I_m are identity matrices of size $n \times n$ and $m \times m$, respectively, where $N = n + m$ and typically $n \gg m$.

Our goal is to develop efficient preconditioners for Krylov subspace methods for solving the system (1.2). The gist of the proposed preconditioner is to combine the effects of two simple and inexpensive preconditioners. One of the constituent preconditioners is the well known block Jacobi preconditioner. The block Jacobi preconditioner is known to deteriorate dramatically with the increasing size of the off-diagonal blocks. The other preconditioner is proposed here to have a corrective effect on the block Jacobi preconditioner. We combine the two preconditioners in a multiplicative fashion and obtain product preconditioners. The resulting algorithm can be regarded as a two-level method where the block Jacobi preconditioner plays the role of a smoothing relaxation with the second preconditioner playing the role of a "coarse grid" correction. Our approach, however, is different from two-level algebraic multigrid or Schwarz methods known in the literature; see, e.g., [19] and the references therein. It is also distinct from (and simpler than) the iterative aggregation-disaggregation (IAD) approach [29].

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 [3, 4, 8, 15, 18, 20]. Most of the attention so far has focused on (linear) stationary iterative methods, including block versions of Jacobi and Gauss–Seidel [8, 18, 20], and on (nonlinear) iterative aggregation/disaggregation schemes specifically tailored to stochastic matrices [8, 15]. In contrast, little work has been done with parallel preconditioned Krylov subspace methods. The suitability of preconditioned Krylov subspace methods for solving Markov models has been demonstrated, e.g., in [22, 25], although no discussion of parallelization aspects was given there. Parallel computing aspects can be found in [4], where a symmetrizable stationary iteration (Cimmino's method) was accelerated using the Conjugate Gradients method on a Cray T3D, and in [18], 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. We further mention [6], where parallel preconditioners based on sparse approximate pseudoinverses were used to speed-up the convergence of BiCGStab.

The paper is organized as follows. We briefly review background material on M-matrices, stationary iterative methods, matrix splittings, and graph partitioning in Section 2. Then, we discuss two simple preconditioners based on regular splittings and introduce the product preconditioners in Section 3. Section 4 contains materials on partitioning the matrices into the 2×2 block structure (1.3) with an eye to future parallel implementations of the proposed product preconditioner. In Section 5 we investigate the effect of the product preconditioner under various partitionings, the properties of the 2×2 block structure imposed by the graph partitioning, and the performance of the product preconditioner relative to that of some other well-known preconditioners. We present our conclusions in Section 6.

2. Background. Here we borrow some material from [5, 6, 7, 31] to provide the reader with a short summary of the concepts and results that are used in building the proposed preconditioners. We also give a brief description of graph partitioning by vertex separator, which can be used to obtain the 2×2 block structure (1.3).

2.1. Nonnegative matrices and M-matrices. A matrix $A_{N\times N}$ is nonnegative if all of its entries are nonnegative, i.e., $A \geq O$ if $a_{ij} \geq 0$ for all $1 \leq i, j \leq N$.

Any matrix A with nonnegative diagonal entries and nonpositive off-diagonal entries can be written in the form

$$A = sI - B, \quad s > 0, \quad B \geq O \; . \tag{2.1}$$

A matrix A of the form (2.1) with $s \geq \rho(B)$ is called an M-matrix. Here, $\rho(B)$ denotes the spectral radius of B. If $s = \rho(B)$ then A is singular, otherwise nonsingular. If A is a nonsingular M-matrix, then $A^{-1} \geq O$.

If A is a singular, irreducible M-matrix, then each $k \times k$ principal square submatrix of A, where $1 \leq k < N$, is a nonsingular M-matrix. If, furthermore, A is the generator of an ergodic Markov chain, then the Schur complement $S_{m\times m} = A_{22} - A_{21}A_{11}^{-1}A_{12}$ of A_{11} (Eq. (1.3)) is a singular, irreducible M-matrix with rank $m - 1$.

2.2. Stationary iterations and matrix splittings. Consider the solution of a linear system of the form $Ax = b$, where A is an $N \times N$ square matrix, possibly singular, and $x, b \in \mathbb{R}^N$. The representation $A = B - C$ is called a splitting if B is nonsingular. A splitting gives rise to the stationary iterative method

$$x^{k+1} = Tx^k + c, \quad k = 0, 1, \ldots , \tag{2.2}$$

where $T = B^{-1}C$ is called the iteration matrix, $c = B^{-1}b$, and $x^0 \in \mathbb{R}^N$ is a given initial guess. The splitting $A = B - C$ is called *regular* if $B^{-1} \geq O$ and $C \geq O$ [31], *weak regular* if $B^{-1} \geq O$ and $T \geq O$ [7], and an *M-splitting* if B is an M-matrix and $C \geq O$ [27].

It is well know that the convergence of the stationary iteration (2.2) depends on the convergence of the sequence T^k as $k \to \infty$; see, e.g., [31]. The matrix T is said to be convergent [21] if the powers T^k converge to a limiting matrix as $k \to \infty$. If the limit is the zero matrix, then T is called zero-convergent. For a nonsingular matrix A, a necessary and sufficient condition for the convergence of (2.2) for any x^0 is that T be zero-convergent, or equivalently, that $\rho(T) < 1$. In the singular case the situation is more involved [7, 30]. In this case, 1 is in the spectrum of T, i.e., $1 \in \sigma(T)$, and a necessary condition for convergence is that $\rho(T) = 1$ be the only eigenvalue on the unit circle, i.e., $\gamma(T) := \max\{|\lambda|, \lambda \in \sigma(T), \lambda \neq 1\} < 1$. If the original system $Ax = b$ is consistent and T is convergent, the iteration (2.2) converges to a solution which depends, in general, on the initial guess x^0.

A related approach is defined by the *alternating iterations*

$$\left\{ \begin{array}{rcl} x^{k+1/2} & = & M_1^{-1}N_1x^k + M_1^{-1}b \\ x^{k+1} & = & M_2^{-1}N_2x^{k+1/2} + M_2^{-1}b, \quad k = 0, 1, \ldots , \end{array} \right. \tag{2.3}$$

where $A = M_1 - N_1 = M_2 - N_2$ are splittings of A, and x^0 is the initial guess. The convergence of alternating iterations is analyzed by Benzi and Szyld [5]. They construct a single splitting $A = B - C$ associated with the iteration matrix by eliminating $x^{k+1/2}$ from the second equation in (2.3) and obtain

$$x^{k+1} = M_2^{-1}N_2M_1^{-1}N_1x^k + M_2^{-1}(N_2M_1^{-1})b, \quad k = 0, 1, \ldots , \tag{2.4}$$

which is in the form of (2.2) with $T = M_2^{-1}N_2M_1^{-1}N_1$. Using this formulation, they construct a unique splitting $A = B - C$ with $B^{-1}C = T$. The splitting is defined by (Eq. (10) in [5])

$$B^{-1} = M_2^{-1}(M_1 + M_2 - A)M_1^{-1} \; . \tag{2.5}$$

Clearly, the matrix $M_1 + M_2 - A$ must be nonsingular for (2.5) to be well-defined.

2.3. Graph partitioning. Given an undirected graph $G = (V, E)$, the problem of K-way graph partitioning by vertex separator (GPVS) asks for finding a set of vertices V_S of minimum size whose removal decomposes a graph into K disconnected subgraphs with balanced sizes. The problem is NP-hard [9]. Formally, $\Pi = \{V_1, \ldots, V_K; V_S\}$ is a K-way vertex partition by vertex separator V_S if the following conditions hold: $V_k \subset V$ and $V_k \neq \emptyset$ for $1 \leq k \leq K$; $V_k \cap V_\ell = \emptyset$ for $1 \leq k < \ell \leq K$ and $V_k \cap V_S = \emptyset$ for $1 \leq k \leq K$; $\bigcup_k V_k \cup V_S = V$; there is no edge between vertices lying in two different parts V_k and V_ℓ for $1 \leq k < \ell \leq K$; $W_{max}/W_{avg} \leq \epsilon$, where W_{max} is the maximum part size (defined as $\max_k |V_k|$), W_{avg} is the average part size (defined as $(|V| - |V_S|)/K$), and ϵ is a given maximum allowable imbalance ratio. See the works [1, 10, 14, 13, 16] for applications of the GPVS and heuristics for GPVS.

In the weighted GPVS problem, the vertices of the given undirected graph have weights. The weight of the separator or a part is defined as the sum of the weights of the vertices that they contain. The objective of the weighted GPVS problem is to minimize the weight of the separator and to maintain balance on the part weights.

3. Product splitting preconditioners. We will consider two preconditioners based on regular splittings of A and combine them as in the alternating iterations (2.4) to build an effective preconditioner for Krylov subspace methods.

The first preconditioner is the well-known block Jacobi preconditioner:

$$M_{BJ} = \begin{bmatrix} A_{11} & O \\ O & A_{22} \end{bmatrix} . \tag{3.1}$$

Note that A_{11} and A_{22} are nonsingular M-matrices, and $A = M_{BJ} - (M_{BJ} - A)$ is a regular splitting (in fact, an M-splitting).

Next, we consider another simple preconditioner:

$$M_{SC} = \begin{bmatrix} D_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} . \tag{3.2}$$

Here $D_{11} \neq A_{11}$ stands for an approximation of A_{11}; in practice, we take D_{11} to be the diagonal matrix formed with the diagonal entries of A_{11}. More generally, we assume that D_{11} is a matrix obtained from A_{11} by setting off-diagonal entries to zero. Thus, D_{11} is a nonsingular M-matrix [31, Theorem 3.12]. The Schur complement matrix $A_{22} - A_{21}D_{11}^{-1}A_{12}$ is therefore well-defined and under the (very mild) structural conditions given in [6, Theorem 3], it is also a nonsingular M-matrix. These conditions are satisfied for the problems considered in this paper. Therefore M_{SC} is a nonsingular M-matrix and $A = M_{SC} - (M_{SC} - A)$ is an M-splitting (hence, a regular splitting).

Since both M_{BJ} and M_{SC} define regular splittings, the product preconditioner M_{PS} given by

$$M_{PS}^{-1} = M_{SC}^{-1}(M_{BJ} + M_{SC} - A)M_{BJ}^{-1} , \tag{3.3}$$

(see (2.5)) implicitly defines a weak regular splitting [5, Theorem 3.4]. Note that since the matrix

$$M_{BJ} + M_{SC} - A = \begin{bmatrix} D_{11} & O \\ O & A_{22} \end{bmatrix} \tag{3.4}$$

is invertible, M_{PS}^{-1} is well-defined, and so is the corresponding splitting of A. We also have

$$\begin{aligned} M_{PS} &= M_{BJ}(M_{BJ} + M_{SC} - A)^{-1} M_{SC} \\ &= \begin{bmatrix} A_{11} & O \\ O & A_{22} \end{bmatrix} \begin{bmatrix} D_{11}^{-1} & O \\ O & A_{22}^{-1} \end{bmatrix} \begin{bmatrix} D_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \\ &= \begin{bmatrix} A_{11} & A_{11}D_{11}^{-1}A_{12} \\ A_{21} & A_{22} \end{bmatrix} , \end{aligned}$$

and therefore

$$A - M_{PS} = \begin{bmatrix} O & (A_{12} - A_{11}D_{11}^{-1}A_{12}) \\ O & O \end{bmatrix} .$$

It follows from the identity $M_{PS}^{-1}A = I + M_{PS}^{-1}(A - M_{PS})$ that $M_{PS}^{-1}A$ (or AM_{PS}^{-1}) has at least n eigenvalues all equal to 1. Exactly one eigenvalue is zero; the remaining $m-1$ all have positive real part and lie in a disk of radius $\rho \leq 1$ and center at the point $(1,0)$ in the complex plane, since $A = M_{PS} - (M_{PS} - A)$ is a weak regular splitting [21]. The better D_{11} approximates A_{11}, the smaller ρ is, and the more clustered the nonzero eigenvalues are around the point $(1,0)$. In general, a clustered spectrum near $(1,0)$ implies fast convergence of the preconditioned Krylov subspace iteration.

Since the order the A_{22} block m is an upper bound on the number of non-unit eigenvalues, it is important to keep this number as small as possible. This is also desirable from the point of view of parallel efficiency; see [6] and the next section.

Application of the M_{BJ} preconditioner requires solving two uncoupled linear systems with the A_{11} and A_{22} blocks:

$$\begin{cases} A_{11}x_1 &= b_1 \\ A_{22}x_2 &= b_2 \end{cases} . \tag{3.5}$$

Within a Krylov subspace method, these linear systems can be solved exactly or approximately. When the blocks are large, as they are bound to be in realistic applications, exact solves are inefficient in terms of both time and storage, and inexact solves must be used. Although the use of iterative methods is a possibility (leading to an inner-outer iterative scheme), in this paper we perform inexact solves by means of incomplete factorizations.

Application of the M_{SC} preconditioner requires solving coupled equations of the form

$$\begin{cases} D_{11}x_1 + A_{12}x_2 &= b_1 \\ A_{21}x_1 + A_{22}x_2 &= b_2 \end{cases} . \tag{3.6}$$

A convenient way to solve the above system is to eliminate x_1 from the second equation using the first one and to solve the Schur complement system

$$(A_{22} - A_{21}D_{11}^{-1}A_{12})\, x_2 = b_2 - A_{21}D_{11}^{-1}b_1 \tag{3.7}$$

for x_2. Then substituting x_2 into the first equation in (3.6) results in the system

$$D_{11}x_1 = b_1 - A_{12}x_2 , \tag{3.8}$$

which is easy to solve.

Application of the M_{PS} preconditioner requires the solution of a system of the form (3.5), a matrix-vector multiply with the matrix $M_{BJ} + M_{SC} - A$ given in (3.4), and the solution of two systems of the form (3.7) and (3.8). In analogy with domain decomposition methods for partial differential equations, (3.7) can be interpreted as a kind of "coarse grid" correction, even though there may be no underlying physical grid. Note that x_2 corresponds to the interface unknowns. On the other hand, x_1 corresponds to subdomain unknowns. Note, however, that (3.7) is embedded in the global solve (3.6), which differentiates our approach from standard two-level Schwarz methods.

4. Building the block structure. The first requirement to be met in permuting the matrix A into 2×2 block structure (1.3) is that the permutation should be symmetric. A symmetric permutation on the rows and columns of A guarantees that A_{11} is an $n \times n$ invertible M-matrix, since the transition probability matrix P is irreducible and $A = I - P^T$. Moreover, A_{11} is diagonally dominant by columns.

The second requirement, as already discussed in Section 3, is to keep the order m of A_{22} as small as possible. The requirement is important in order to have fast convergence of the Krylov subspace method, since m is an upper bound on the number of non-unit eigenvalues. Strictly speaking, this is true only if we assume exact solves in the application of the preconditioner. In practice we will use inexact solves, and rather than having n eigenvalues (or more) exactly equal to 1, there will be a cluster of at least n eigenvalues near the point $(1, 0)$. Still, we want this cluster to contain as many eigenvalues as possible.

The second requirement is also desirable from the point of view of parallel implementation. A possible parallelization approach would be constructing and solving the system (3.7) on a single processor and then solving the system (3.8) in parallel. This approach has been taken previously in parallelizing applications of approximate inverse preconditioners [6]. Another possible approach would be parallelizing the solution of (3.7) either by allowing redundancies in the computations (each processor can form the whole system or a part of it) or by running a parallel solver on (3.7) itself. In both cases, the solution with the Schur complement system constitutes a serial bottleneck and requires additional storage space.

The third requirement, also stemming from the discussion in Section 3, is to have D_{11} as close to A_{11} as possible in order to cluster the non-unit eigenvalues of $M_{PS}^{-1}A$. Meeting this requirements would likely help reduce the number of iterations to convergence for the Krylov subspace method.

The fourth requirement, not necessary for the convergence analysis but crucial for an efficient implementation, is that A_{11} should be block diagonal with subblocks of approximately equal size and density. Given K subblocks in the (1,1) block A_{11}, the application of the M_{BJ} preconditioner, i.e., solving the system (3.5), requires $K + 1$ independent solves: one with the A_{22} block and one with each subblock of A_{11}. Similarly, the application of the M_{SC} preconditioner after solving (3.7) requires K independent solves: one with the Schur complement and a diagonal solve (scaling) for each subblock of D_{11}. We note that this form of the M_{SC} preconditioner is a special case of a *domain decomposition splitting*, as defined in [32]. Meeting this requirement for a serial implementation will enable solution of very large systems, since the subblocks can be handled one at a time. In any admissible parallelization, each of these subblocks would more likely be assigned to a single processor. Therefore, maintaining balance on the sizes and the densities of the subblocks will relate to maintaining balance on computational loads of the processors. Furthermore, it is

desirable that sizes of these subblocks be larger than the order m of A_{22}, if possible, for the reasons given for the second requirement.

Meeting all of the above four requirements is a very challenging task. Therefore, as a pragmatic approach we totally ignore the third one and apply well established heuristics for addressing the remaining three requirements. As it is common, we adopt the standard undirected graph model to represent a square matrix $A_{N\times N}$. The vertices of the graph $G(A) = (V, E)$ correspond to the rows and columns of A and the edges correspond to the nonzeros of A. The vertex $v_i \in V$ represents the ith row and the ith column of A, and there exists an edge $(v_i, v_j) \in E$ if a_{ij} and a_{ji} is nonzero.

Consider a partitioning $\Pi = \{V_1, \ldots, V_K; V_S\}$ of $G(A)$ with vertex separator V_S. The matrix A can be permuted into the 2×2 block structure (1.3) by permuting the rows and columns associated with the vertices in $\bigcup_k V_k$ before the rows and columns associated with the vertices in V_S. That is, V_S defines the rows and columns of the (2,2) block A_{22}. Notice that the resulting permutation is symmetric, and hence the first requirement is met. Furthermore, since GPVS tries to minimize the size of the separator set V_S, it tries to minimize the order of the block A_{22}. Therefore, the permutation induced by Π meets the second requirement as well.

Consider the A_{11} block defined by the vertices in $\bigcup_k V_k$. The rows and columns that are associated with the vertices in V_k can be permuted before the rows and columns associated with the vertices in V_ℓ for $1 \le k < \ell \le K$. Such a permutation of A_{11} gives rise to diagonal subblocks. Since we have already constructed A_{22} using V_S, we end up with the following structure:

$$A = \begin{bmatrix} A_1 & & & & B_1 \\ & A_2 & & & B_2 \\ & & \ddots & & \vdots \\ & & & A_K & B_K \\ C_1 & C_2 & \cdots & C_K & A_S \end{bmatrix}.$$

The diagonal blocks $A_1, \ldots, A_K$ correspond to the vertex parts $V_1, \ldots, V_K$, and therefore 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 represents the connections between nodes in the separator set. Note that because of the irreducibility assumption, each block A_i is a nonsingular M-matrix. Thus, graph partitioning induces a reordering and block partitioning of the matrix A in the form (1.3) where

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

and

$$A_{12} = [B_1^T\ B_2^T\ \cdots\ B_K^T]^T, \quad A_{21} = [C_1\ C_2\ \cdots\ C_K].$$

Therefore, the permutation induced by the GPVS partially addresses the fourth requirement. Note that the GPVS formulation ignores the requirement of balancing the densities of the diagonal subblocks of A_{11}. In fact, obtaining balance on the densities of the diagonal blocks is a complex partitioning requirement that cannot be met before a partitioning takes place (see [23] for a possible solution) even with a weighted GPVS formulation.

If the matrix is structurally nonsymmetric, which is common for matrices arising from Markov chains, then A cannot be modeled with undirected graphs. In this case, a 2×2 block structure can be obtained by partitioning the graph of $A + A^T$.

TABLE 5.1
Properties of the generator matrices.

Matrix	number of rows/cols	number of nonzeros					
		total	average	row		col	
	N		row/col	min	max	min	max
mutex09	65535	1114079	17.0	16	17	16	17
mutex12	263950	4031310	15.3	9	21	9	21
ncd07	62196	420036	6.8	2	7	2	7
ncd10	176851	1207051	6.8	2	7	2	7
qnatm06	79220	533120	6.7	3	9	4	7
qnatm07	130068	875896	6.7	3	9	4	7
tcomm16	13671	67381	4.9	2	5	2	5
tcomm20	17081	84211	4.9	2	5	2	5
twod08	66177	263425	4.0	2	4	2	4
twod10	263169	1050625	4.0	2	4	2	4

5. Numerical experiments. In this section, we report on experimental results obtained with a Matlab 6 implementation on a 1.2GHz Sun Fire V880 with 2 Gbytes main memory. The main goal was to test the product splitting preconditioner and to compare it with a few other preconditioners. The Krylov method used was GMRES [26]. For completeness we performed experiments with the stationary iterations corresponding to the various splittings (without GMRES acceleration), but they were found to converge too slowly to be competitive with preconditioned GMRES. Therefore, we do not show these results.

The various methods were tested on the generator matrices of some Markov chain models provided in the MARCA (MARkov Chain Analyzer) collection [28]. The models are discussed in [12, 22, 24] and have been used to compare different solution methods in [6, 11] and elsewhere. These matrices are infinitesimal generators of time-continuous Markov chains, but can be easily converted (as we did) to the form $A = I - P^T$, with P row-stochastic, so that A corresponds to a discrete-time Markov chain, known as the *embedded Markov chain*; see [29, Chapter 1.4.3]. The preconditioning techniques described in this paper can be applied to either form of the generator matrix.

We performed a large number of tests on numerous matrices; here we present a selection of results for a few test matrices, chosen to be representative of our overall findings.

Table 5.1 displays the properties of the test matrices. Each matrix is named by its family followed by its index in the family. For example, `mutex09` refers to the 9th matrix in the `mutex` family. The matrices from the `mutex` and `ncd` families are structurally symmetric, the matrices from the `qnatm` and `twod` families are structurally nonsymmetric, and the matrices from the `tcomm` family are very close to being structurally symmetric—the nonzero patterns of `tcomm20` and `tcomm16` differ from the nonzero patterns of their transposes in only 60 locations.

We compared the product preconditioner (PS) with its factors block Jacobi (BJ) and Schur complement-based (SC) preconditioners. We also compared PS with the block Gauss-Seidel (BGS) and block successive overrelaxation (BSOR) preconditioners, where the preconditioner matrices are

$$M_{BGS} = \begin{bmatrix} A_{11} & A_{12} \\ O & A_{22} \end{bmatrix} \quad \text{or} \quad M_{BGS} = \begin{bmatrix} A_{11} & O \\ A_{21} & A_{22} \end{bmatrix},$$

$$M_{BSOR} = \begin{bmatrix} \frac{1}{\omega}A_{11} & A_{12} \\ O & \frac{1}{\omega}A_{22} \end{bmatrix} \quad \text{or} \quad M_{BSOR} = \begin{bmatrix} \frac{1}{\omega}A_{11} & O \\ A_{21} & \frac{1}{\omega}A_{22} \end{bmatrix}.$$

In agreement with the previously reported results [11] on the MARCA collection, we observed that $\omega = 1.0$ (which reduces the BSOR to BGS) or very close to 1.0 is nearly always the best choice of the relaxation parameter for BSOR. We also observed that the block lower triangular versions of the BGS and BSOR preconditioners are indistinguishable from the block upper triangular versions under either the storage or performance criteria. Therefore, we report only the experiments with the upper triangular BGS preconditioner. Note that application of the BGS preconditioner requires two linear system solutions (one with A_{11} and one with A_{22}) and a matrix-vector multiply with A_{12}.

5.1. Properties of the block structure and the preconditioners. We partitioned the matrix into the 2×2 block structure (1.3) using Metis [17]. For the structurally symmetric `mutex` and `ncd` matrices, we used the graph of A, and for the other matrices we used the graph of $A+A^T$ as mentioned in Section 4. As discussed in Section 4, we maintain balance on the size, rather than the densities, of the subblocks of A_{11}. We have conducted experiments with $K = 2, 4, 8, 16$, and 32 subblocks in the (1,1) block. For each K value, K-way partitioning of a test matrix constitutes a partitioning instance. Since Metis incorporates randomized algorithms, it was run 20 times starting from different random seeds for each partitioning instance. The maximum allowable imbalance ratio among the part weights was specified as 25%. In all partitioning instances except the `mutex` matrices, the imbalance ratios among the parts were within the specified limit. Therefore, the partition with the minimum separator size was chosen for those matrices. For the `mutex` matrices, we choose the best among the partitions that satisfies the imbalance ratio, if any. Otherwise, we chose the one with smallest imbalance (this was the case in $K = 16$- and 32-way partitioning of both of the matrices and the resulting imbalance was 35%). The properties of the 2×2 block structures corresponding to these best partitions are given in Table 5.2.

As seen from Table 5.2, only the `mutex` matrices have a large number of rows in the second row block of A, i.e., a large separator set. For these matrices, only for the $K = 2$-way partitioning instances the average part size is larger than the size of the separator set. For the other matrices, the average part size is larger than the size of the second row block in all partitioning instances with $K = 2, 4, 8$, and 16 except in $K = 16$-way partitioning of `ncd07` and `qnatm06`. The average separator sizes for all K values are given at the bottom of the table. These averages contain the `mutex` matrices; without them, the average separator and part sizes for $K = 16$ are 0.043 and 0.060, respectively.

An interesting observation is that in all partitioning instances, the (2,2) block is always very sparse; the minimum and maximum number of nonzeros per row in the (2,2) block are 1.0 (in $K = 2, 4$, and 8-way partitioning of `mutex12` and in almost all `ncd` partitioning instances) and 4.2 (in $K = 16$-way partitioning of `mutex09`) where the overall average is 1.67. All matrices, except `mutex12` and `mutex09` have small numbers of nonzeros per row. Therefore, a very sparse separator is most likely to occur in partitioning these matrices. However, the separator being highly sparse in different Markov chain models is noteworthy, especially from the parallelization perspective. For example, the (2,2) block may be duplicated on all processors to overcome the serial bottleneck of Schur complement solves with only a small storage overhead, or the off-diagonal entries of the (2,2) block may be selectively dropped to yield an approximate Schur complement in order to reduce the storage requirements.

TABLE 5.2

Properties of the partitions and the induced block structures for the matrices. The column "sep" refers to the number of rows in the 2nd row block of A normalized by the number of rows in A, i.e., m/N; the column "part" refers to the average part size normalized by the number of rows in A, i.e., $(n/K)/N$; the columns A_{ij} for $i, j = 1, 2$ refer to the number of nonzeros in the (i, j) block normalized by the number of nonzeros in A, i.e., $nnz(A_{ij})/nnz(A)$.

Matrix	K	Partition		Blocks			
		sep	part	A_{11}	A_{12}	A_{21}	A_{22}
mutex09	2	0.205	0.398	0.618	0.177	0.177	0.028
	4	0.337	0.166	0.363	0.300	0.300	0.037
	8	0.415	0.073	0.237	0.348	0.348	0.068
	16	0.469	0.033	0.177	0.354	0.354	0.116
	32	0.473	0.016	0.138	0.389	0.389	0.084
mutex12	2	0.141	0.429	0.621	0.185	0.185	0.009
	4	0.225	0.194	0.397	0.294	0.294	0.015
	8	0.282	0.090	0.243	0.369	0.369	0.018
	16	0.333	0.042	0.162	0.389	0.389	0.060
	32	0.343	0.021	0.124	0.411	0.411	0.053
ncd07	2	0.015	0.493	0.972	0.013	0.013	0.002
	4	0.028	0.243	0.946	0.025	0.025	0.004
	8	0.047	0.119	0.910	0.041	0.041	0.007
	16	0.071	0.058	0.866	0.062	0.062	0.010
	32	0.099	0.028	0.813	0.086	0.086	0.015
ncd10	2	0.012	0.494	0.976	0.011	0.011	0.002
	4	0.023	0.244	0.957	0.020	0.020	0.003
	8	0.036	0.121	0.933	0.031	0.031	0.005
	16	0.057	0.059	0.893	0.049	0.049	0.009
	32	0.076	0.029	0.856	0.066	0.066	0.012
qnatm06	2	0.012	0.494	0.979	0.009	0.009	0.003
	4	0.024	0.244	0.957	0.019	0.019	0.005
	8	0.041	0.120	0.927	0.032	0.032	0.009
	16	0.068	0.058	0.877	0.055	0.055	0.013
	32	0.100	0.028	0.819	0.081	0.081	0.019
qnatm07	2	0.009	0.496	0.986	0.005	0.005	0.004
	4	0.019	0.245	0.966	0.015	0.015	0.004
	8	0.034	0.121	0.940	0.026	0.026	0.008
	16	0.054	0.059	0.903	0.043	0.043	0.011
	32	0.081	0.029	0.854	0.065	0.065	0.017
tcomm16	2	0.002	0.499	0.996	0.002	0.002	0.001
	4	0.007	0.248	0.988	0.005	0.005	0.002
	8	0.016	0.123	0.973	0.011	0.011	0.005
	16	0.034	0.060	0.942	0.024	0.024	0.010
	32	0.059	0.029	0.898	0.042	0.042	0.018
tcomm20	2	0.002	0.499	0.997	0.001	0.001	0.001
	4	0.005	0.249	0.991	0.004	0.004	0.002
	8	0.013	0.123	0.978	0.009	0.009	0.004
	16	0.027	0.061	0.954	0.019	0.019	0.008
	32	0.054	0.030	0.908	0.038	0.038	0.016
twod08	2	0.002	0.499	0.997	0.001	0.001	0.001
	4	0.006	0.249	0.991	0.003	0.003	0.003
	8	0.013	0.123	0.980	0.007	0.007	0.007
	16	0.021	0.061	0.968	0.011	0.011	0.011
	32	0.035	0.030	0.947	0.018	0.018	0.018
twod10	2	0.002	0.499	0.997	0.001	0.001	0.001
	4	0.004	0.249	0.994	0.002	0.002	0.002
	8	0.007	0.124	0.989	0.004	0.004	0.004
	16	0.012	0.062	0.982	0.006	0.006	0.006
	32	0.019	0.031	0.972	0.009	0.009	0.009
Averages							
	2	0.040	0.480	0.914	0.040	0.040	0.005
	4	0.068	0.233	0.855	0.069	0.069	0.008
	8	0.090	0.114	0.811	0.088	0.088	0.013
	16	0.115	0.055	0.772	0.101	0.101	0.026
	32	0.134	0.027	0.733	0.121	0.121	0.026

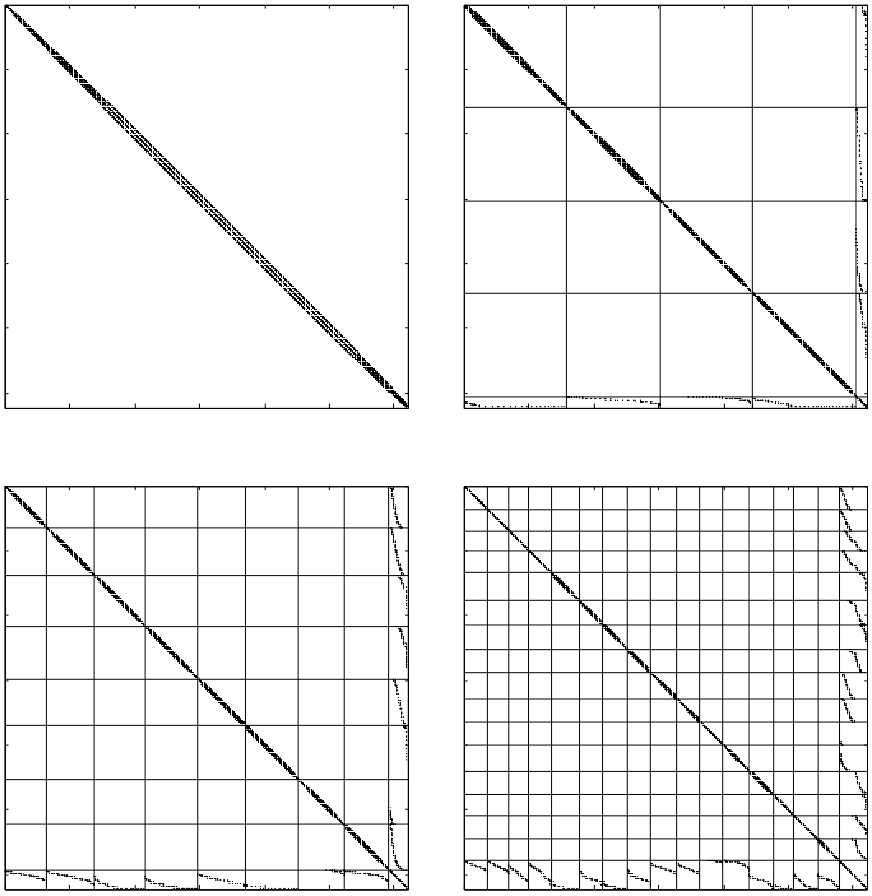

FIG. 5.1. *Sparsity pattern of the matrix* `ncd07` *and structure induced by* K*-way partitioning for* $K = 4, 8, 16$.

As an example of the partitioning outcomes, Figure 5.1 shows the sparsity pattern of the matrix `ncd07` followed by the block structure of the K-way partitionings for $K = 4, 8, 16$.

Each subblock of A_{11} and the (2,2) block A_{22} are factored using the incomplete LU factorization (ILUTH) with threshold parameter $\tau = 0.01$ for the `qnatm` matrices and $\tau = 0.001$ for the other matrices. The threshold of 0.001 was too small for the `qnatm` matrices: the resulting preconditioners had 8 times more nonzeros than the generator matrices. The densities of the preconditioners, i.e., the number of nonzeros in the matrices appearing in the preconditioner solve phase divided by the number of nonzeros in the corresponding generator matrices, are given in Table 5.3. The number of nonzeros in the preconditioner solve phase of BJ is the total number of nonzeros in the approximate ILU factors of A_{11} and A_{22}. The number of nonzeros in the preconditioner solve phase of SC is the total number of nonzeros in the approximate ILU factors of the Schur complement matrix in (3.7), plus the number of nonzeros in A_{21} and A_{12}. The number of nonzeros in the preconditioner solve phase of PS is computed by adding the number of nonzeros in $M_{BJ} + M_{SC} - A$ to that for the

BJ and SC preconditioners. The number of nonzeros for the BGS preconditioner is the total number of nonzeros in the ILU factors of A_{11} and A_{22} plus the number of nonzeros in A_{21}.

Since the number of nonzeros in the off-diagonal blocks A_{12} and A_{21} increases for increasing K, the number of nonzeros in the BJ preconditioners decreases for increasing K. By the same token, the number of nonzeros in the SC preconditioners increases for increasing K. Since the proposed preconditioner PS uses those two preconditioners and an additional $n+nnz(A_{22}) \approx N$ nonzeros, the number of nonzeros in the PS preconditioners is not heavily affected by the increasing number of parts. As seen from the averages given in the bottom of the table, the product preconditioners have around 2.5 times the nonzeros of the generator matrices for all K, whereas the BGS preconditioners contain around 1.97 times the nonzeros of the generator matrices, on the average. Thus, on average, the PS preconditioners contain 28% more nonzeros than the BGS preconditioners.

5.2. Performance comparisons. The underlying Krylov subspace method was GMRES, restarted every 50 iterations (if needed). Right preconditioning was used in all the tests. The stopping criterion was set as

$$\|r_k\|_2/\|r_0\|_2 < 10^{-10} ,$$

where r_k is the residual at the kth iteration and r_0 is the initial residual. We allow at most 250 iterations, i.e., 5 restarts, for the GMRES iteration. Therefore, the number 250 in the following tables marks the cases in which GMRES failed to deliver solutions with the prescribed accuracy within 250 iterations. For each matrix, only one initial solution is chosen at random (with positive entries) and normalized to have an ℓ_1-norm of 1.0. In other words, the initial guess is a random probability distribution vector. The same initial guess is used for all partitioning instances of a matrix with all preconditioners. Iteration counts for GMRES(50) on the various test matrices with no permutation or preconditioning (GMRES) and with the preconditioners BJ, SC, BGS, and PS are given in Table 5.4. The ℓ_1-norms of the residuals corresponding to the approximate solutions returned by GMRES(50) were between 1.9889e-17 and 1.6098e-11 for the converged instances. Note that without preconditioning, GMRES(50) converges only for the `mutex` matrices.

As seen from Table 5.4, the BGS preconditioner is consistently better than the BJ and SC preconditioners. The proposed PS preconditioner, although constructed from the two preconditioners outperformed by BGS, is in turn consistently better than the BGS preconditioner. The last five rows of the table show the performance of the proposed PS preconditioner with respect to the BGS preconditioner. These numbers are computed by first normalizing the number of iterations of BGS and PS by the number of iterations required by BGS preconditioner with $K = 2$-way partitioning for each matrix. Then, the averages of these numbers are displayed in the last five rows. We did not display the averages for the BJ and SC preconditioners because they do not lead to convergence in all instances. As seen from the normalized averages, the proposed PS preconditioner outperforms the BGS one by a factor of two in terms of iteration counts, at the expense of an increase of just 28% (approximately) in the number of nonzeros (see Table 5.3).

Recall from Section 3 that the PS preconditioner is given by $M_{PS}^{-1} = M_{BJ}^{-1}(M_{BJ} + M_{SC} - A)M_{SC}^{-1}$. Consider the matrix in the middle. It is composed of the diagonal of A_{11} and the (2,2) block A_{22}. Since the A_{22} block is very sparse, i.e., close to being a diagonal matrix, the effect of the multiply with $(M_{BJ} + M_{SC} - A)$ is merely a

TABLE 5.3

The densities of the preconditioners, i.e., the total number of nonzeros in the matrices appearing in the preconditioning matrices divided by the number of nonzeros in the corresponding generator matrices.

Matrix	K	Preconditioners			
		BJ	SC	BGS	PS
mutex09	2	0.75	0.67	0.93	1.50
	4	0.47	1.26	0.77	1.81
	8	0.37	1.50	0.72	1.97
	16	0.36	1.56	0.72	2.07
	32	0.28	1.78	0.67	2.17
mutex12	2	0.68	0.62	0.87	1.37
	4	0.47	1.01	0.76	1.55
	8	0.31	1.34	0.68	1.72
	16	0.29	1.51	0.68	1.90
	32	0.24	1.58	0.65	1.92
ncd07	2	0.88	0.18	0.89	1.21
	4	0.86	0.21	0.89	1.22
	8	0.82	0.25	0.86	1.22
	16	0.79	0.30	0.85	1.23
	32	0.75	0.36	0.84	1.25
ncd10	2	0.72	0.17	0.73	1.04
	4	0.71	0.19	0.73	1.05
	8	0.68	0.22	0.72	1.05
	16	0.66	0.26	0.71	1.07
	32	0.64	0.30	0.71	1.09
qnatm06	2	3.43	0.18	3.44	3.77
	4	3.31	0.22	3.33	3.68
	8	3.17	0.27	3.21	3.60
	16	2.93	0.36	2.98	3.44
	32	2.65	0.46	2.73	3.26
qnatm07	2	3.48	0.17	3.48	3.80
	4	3.38	0.20	3.39	3.73
	8	3.25	0.25	3.27	3.65
	16	3.06	0.31	3.11	3.53
	32	2.83	0.40	2.89	3.38
tcomm16	2	2.12	0.21	2.13	2.53
	4	2.10	0.22	2.11	2.52
	8	2.06	0.24	2.07	2.50
	16	1.97	0.27	2.00	2.45
	32	1.88	0.32	1.92	2.41
tcomm20	2	2.19	0.21	2.19	2.60
	4	2.17	0.21	2.18	2.59
	8	2.13	0.23	2.14	2.57
	16	2.06	0.26	2.08	2.52
	32	1.93	0.31	1.97	2.45
twod08	2	2.38	0.25	2.38	2.88
	4	2.36	0.26	2.36	2.87
	8	2.33	0.27	2.34	2.86
	16	2.30	0.29	2.31	2.84
	32	2.24	0.31	2.25	2.81
twod10	2	3.86	0.25	3.86	4.37
	4	3.84	0.26	3.84	4.35
	8	3.80	0.26	3.80	4.31
	16	3.73	0.27	3.74	4.26
	32	3.66	0.28	3.67	4.20
Averages					
	2	2.05	0.29	2.09	2.51
	4	1.97	0.40	2.04	2.54
	8	1.89	0.48	1.98	2.54
	16	1.82	0.54	1.92	2.53
	32	1.71	0.61	1.83	2.50

TABLE 5.4

Number of iterations to reduce the ℓ_2-norm of the initial residual by ten orders of magnitude using GMRES(50) with at most 5 restarts. The number 250 means that the method did not converge in 250 iterations.

Matrix	GMRES	Preconditioned GMRES				
		K	Preconditioners			
			BJ	SC	BGS	PS
mutex09	97	2	25	27	13	8
		4	29	23	15	9
		8	30	20	15	8
		16	26	22	14	9
		32	29	17	15	9
mutex12	91	2	27	23	14	8
		4	29	20	15	7
		8	29	18	15	7
		16	27	17	14	8
		32	28	17	15	8
ncd07	250	2	38	250	25	16
		4	201	250	69	17
		8	250	250	99	19
		16	250	250	99	21
		32	250	250	158	22
ncd10	250	2	42	250	29	19
		4	250	250	181	21
		8	205	250	101	22
		16	250	250	145	25
		32	250	250	188	26
qnatm06	250	2	64	250	41	39
		4	92	250	45	39
		8	120	250	50	42
		16	200	250	62	45
		32	250	250	86	49
qnatm07	250	2	65	250	45	44
		4	94	250	52	46
		8	138	250	71	48
		16	215	250	87	55
		32	245	250	98	68
tcomm16	250	2	33	250	19	15
		4	50	250	27	20
		8	100	250	36	26
		16	250	250	91	42
		32	250	250	79	42
tcomm20	250	2	31	250	19	16
		4	41	250	24	20
		8	101	250	36	26
		16	250	250	73	40
		32	250	250	200	101
twod08	250	2	26	250	14	9
		4	34	250	18	11
		8	43	250	22	16
		16	51	250	26	21
		32	58	250	30	25
twod10	250	2	35	250	18	12
		4	46	250	24	19
		8	52	250	27	22
		16	50	250	26	21
		32	87	250	37	29
Normalized averages with respect to 2-way BGS						
		2	-	-	1.00	0.74
		4	-	-	1.88	0.85
		8	-	-	1.93	0.98
		16	-	-	2.64	1.23
		32	-	-	3.82	1.67

TABLE 5.5
Running times (in seconds) for GMRES(50) without preconditioning (GMRES column) and with BJ, SC, BGS and PS preconditioning for the `mutex` *matrices.*

Matrix	GMRES		Preconditioned GMRES							
	Total time	K	Total time							
			Preconditioner construction				Solve			
	Solve		BJ	SC	BGS	PS	BJ	SC	BGS	PS
mutex09	19.6	2	1.8	0.7	2.0	3.2	11.0	8.2	5.8	5.0
		4	0.8	3.2	1.0	4.5	11.8	9.0	6.0	6.2
		8	0.5	4.1	0.7	5.2	11.9	8.8	5.8	6.0
		16	0.4	7.9	0.7	8.9	10.2	10.4	5.4	6.9
		32	0.3	5.2	0.5	6.1	11.2	8.4	5.8	7.0
mutex12	79.1	2	6.5	2.6	7.2	11.7	51.9	26.7	26.7	21.0
		4	3.6	6.5	4.3	12.4	52.0	28.5	26.9	19.8
		8	1.9	12.3	2.7	16.5	49.6	30.1	26.0	21.3
		16	1.7	42.4	2.5	46.4	46.4	30.7	23.9	25.3
		32	1.2	35.5	2.1	38.8	47.0	31.6	25.3	25.2

scaling. Therefore, we conducted experiments with the PS preconditioner without the matrix-vector multiplies with $(M_{BJ} + M_{SC} - A)$ and observed that in 41 partitioning instances (even with the `mutex` matrices that have relatively denser A_{22}) the number of iterations did not change at all, in 3 of the cases omitting the multiply decreased the number of iterations by 1, in 4 cases it increased the number of iterations by 1, and only in $K = 32$-way partitioning of `tcomm20` it increased the number of iterations by 5.

We close this section by discussing running times for GMRES. The timings are obtained using Matlab 6's `cputime` command and are in seconds. In Table 5.5, the total time for GMRES without preconditioning is the total time spent in performing the GMRES iterations. In Table 5.5 and 5.6, the total time for the preconditioned GMRES is dissected into the preconditioner construction and the solve phases. We first discuss the case of `mutex` matrices, since all the preconditioners lead to convergence for these matrices. In all partitioning instances of the `mutex` matrices, the proposed PS preconditioner's solve phase time is less than the solve phase times of its factors BJ and SC preconditioners. Furthermore, in half of the instances its solve phase time is smaller than the BGS preconditioner's solve phase time. On the other hand, the total running time of BGS preconditioner is always the minimum except in $K = 2$-way partitioning of `mutex12` in which case BJ gives the minimum total running time. We observe that the `mutex` matrices are the worst case for the construction of the PS preconditioner, since the size of the separator set is very large already for $K = 2$, thus forming the Schur complement is very time-consuming.

Table 5.6 contain the running time of the preconditioned GMRES with the BGS and PS preconditioners for the larger matrices in each matrix family. As seen from the table, for these matrices (whose partitions have small separators) the preconditioner construction phases' time are always smaller than the solve phases' time. Furthermore, the construction times for the BGS and PS preconditioners are almost the same, and PS is faster than BGS (due to smaller number of iterations) in all instances except for $K = 2$-way partitioning of `qnatm07`. Note that the ith iteration of GMRES after a restart requires i inner product computations with vectors of length N [2]. Therefore, the performance gains in the solve phase with the PS preconditioners are not only due to the savings in preconditioner solves and matrix-vector multiplies, but also due to the savings in the inner product computations. What is important in this

TABLE 5.6
Running times (in seconds) for GMRES(50) with BGS and PS preconditioners for the larger matrices in each family.

Matrix	K	Total time			
		Precond const		Solve	
		BGS	PS	BGS	PS
ncd10	2	1.3	1.7	28.1	19.8
	4	1.3	1.7	193.0	22.0
	8	1.2	1.6	108.6	23.4
	16	1.2	1.6	153.7	27.0
	32	1.1	1.6	198.4	28.5
qnatm07	2	20.9	21.2	46.5	49.6
	4	14.9	15.2	53.7	50.8
	8	9.6	10.0	70.3	53.4
	16	6.5	6.9	85.8	61.1
	32	4.9	5.3	93.6	70.9
tcomm20	2	0.2	0.2	1.3	1.2
	4	0.2	0.2	1.7	1.5
	8	0.2	0.2	2.6	2.0
	16	0.2	0.2	5.6	3.3
	32	0.1	0.2	15.8	8.8
twod10	2	17.3	17.6	31.1	22.6
	4	14.5	14.8	42.4	35.9
	8	6.6	6.9	47.3	42.0
	16	6.1	6.4	45.4	39.6
	32	5.2	5.5	67.1	56.4

table is how the total time with the PS preconditioner increases as the number of parts increases. These increases are fairly modest compared to the number of parts. Except for `tcommm20`, which is rather small, the solution times with 32 parts are at most 3 times larger than the solution times with 2 parts. This heralds considerable speedups in the total solution time in a parallel computing environment. Additionally, this also gives leeway in parallelizing the iterations.

6. Conclusions. We have described and investigated a new preconditioning technique for Markov chain problems. The idea is to combine two simple preconditioners to get a new, more effective one. This "product splitting" preconditioner relies on a block 2×2 structure of the generator matrix, which is obtained through graph partitioning. Our approach is somewhat similar to a two-level, non-overlapping additive Schwarz (block Jacobi) preconditioner with a multiplicative "coarse grid correction" represented by an approximate Schur complement solve. Numerical experiments with preconditioned GMRES using test matrices from MARCA indicate that the product preconditioner is much more effective than the two constituent preconditioners, at the expense of only a slight increase in storage requirements. The product splitting was also found to be competitive in most cases with an appropriate block Gauss–Seidel preconditioner. Furthermore, the numerical experiments indicate that, for most problems, the number of iterations grows slowly with the number of parts (subdomains). This suggests that the product splitting preconditioner should perform very well in a parallel implementation.

Acknowledgment. We thank Billy Stewart for making the MARCA software available.

REFERENCES

[1] C. Aykanat, A. Pinar, and U. V. Çatalyürek, *Permuting sparse rectangular matrices into block-diagonal form*, SIAM J. Sci. Comput., 25 (2004), pp. 1860–1879.

[2] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. A. Van Der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM, Philadelphia, 1994.

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

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

[5] M. Benzi and D. Szyld, *Existence and uniqueness of splittings for stationary iterative methods with applications to alternating methods*, Numer. Math., 76 (1997), pp. 309–321.

[6] M. Benzi and M. Tůma, *A parallel solver for large-scale Markov chains*, Appl. Numer. Math., 41 (2002), pp. 135–153.

[7] A. Berman and B. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*, Academic Press, New York, 1979.

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

[9] T. N. Bui and C. Jones, *Finding good approximate vertex and edge partitions is NP hard*, Inform. Process. Lett., 42 (1992), pp. 153–159.

[10] ———, *A heuristic for reducing fill in sparse matrix factorization*, in Proc. 6th SIAM Conf. Parallel Processing for Scientific Computing, Philadelphia, 1993.

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

[12] P. Fernandes, B. Plateau, and W. J. Stewart, *Efficient descriptor-vector multiplications in stochastic automata networks*, J. ACM, 45 (1998), pp. 381–414.

[13] B. Hendrickson and R. Leland, *A multilevel algorithm for partitioning graphs*, in Proc. 1995 ACM/IEEE Conference, High Performance Networking and Computing, Supercomputing '95, New York, 1995.

[14] B. Hendrickson and E. Rothberg, *Improving the run time and quality of nested dissection ordering*, SIAM J. Sci. Comput., 20 (1998), pp. 468–489.

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

[16] G. Karypis and V. Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput., 20 (1998), pp. 359–392.

[17] ———, *MeTiS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices version 4.0*, University of Minnesota, Department of Computer Science / Army HPC Research Center, Minneapolis, MN 55455, September 1998.

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

[19] I. Marek and D. B. Szyld, *Algebraic Schwarz methods for the numerical solution of Markov chains*, Linear Algebra Appl., 386 (2004), pp. 67–81.

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

[21] M. Neumann and R. J. Plemmons, *Convergent nonnegative matrices and iterative methods for consistent linear systems*, Numer. Math., 31 (1978), pp. 265–279.

[22] B. Philippe, Y. Saad, and W. J. Stewart, *Numerical methods in Markov chain modeling*, Oper. Res., 40 (1992), pp. 1156–1179.

[23] A. Pinar and B. Hendrickson, *Graph partitioning for complex objectives*, in Proceedings of Irregular 2001, April 2001, p. 121.

[24] P. K. Pollet and S. E. Stewart, *An efficient procedure for computing quasi-stationary distributions of Markov chains with sparse transition structure*, Adv. Appl. Probab., 26 (1994), pp. 68–79.

[25] Y. Saad, *Preconditioned Krylov subspace methods for the numerical solution of Markov chains*, in Computations with Markov Chains, W. J. Stewart, ed., Kluwer Academic Publishers, Boston/London/Dordrecht, 1995, pp. 49–64.
[26] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.
[27] H. Schneider, *Theorems on M-splittings of a singular M-matrix which depend on graph structure*, Linear Algebra Appl., 58 (1984), pp. 407–424.
[28] W. J. Stewart, *MARCA Models: A collection of Markov chain models*. url `http://www.csc.ncsu.edu/faculty/stewart/MARCA_Models/MARCA_Models.html`.
[29] ———, *Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, Princeton, NJ, 1994.
[30] D. B. Szyld, *Equivalence of convergence conditions for iterative methods for singular equations*, Numer. Linear Algebra Appl., 1 (1994), pp. 151–154.
[31] R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, 1962.
[32] R. E. White, *Domain decomposition splittings*, Linear Algebra Appl., 316 (2000), pp. 105–112.