

# Application of Threshold Partitioning of Sparse Matrices to Markov Chains

Hwajeong Choi and Daniel B. Szyld

Department of Mathematics  
Temple University  
Philadelphia, Pennsylvania 19122-2585

## Abstract

*Three algorithms which permute and partition a sparse matrix are presented as tools for the improved solution of Markov chain problems. One is the algorithm PABLO [SIAM J. Sci. Stat. Computing, Vol. 11, pp. 811–823, 1990] while the other two are modifications of it. In the new algorithms, in addition to the location of the nonzeros, the values of the entries are taken into account. The permuted matrices are well suited for block iterative methods that find the corresponding probability distribution, as well as for block diagonal preconditioners of Krylov-based methods. Also, if the partition obtained from the ordering algorithm is used as an aggregation scheme, an iterative aggregation method performs better with this partition than with others found in the literature. Numerical experiments illustrate the performance of the iterative methods with the new orderings.*

## 1 Introduction

In this contribution we explore the application of certain ordering and partitioning algorithms to the problem of finding stationary probability distributions of Markov chains. The main idea of the paper is to symmetrically permute the corresponding matrix, so that the permuted matrix has an almost block diagonal form. Then, the blocks in the diagonal are used for each of the following solution methods: (a) a classical iterative method, such as block Jacobi or block Gauss-Seidel, (b) as a block diagonal preconditioner for a Krylov-based iterative method, such as GMRES [18], or (c) to determine the aggregation scheme of an iterative aggregation method (also called aggregation / disaggregation method). We refer the reader, e.g., to the texts by Berman and Plemmons [3], W. Stewart [24], or Varga [28] for description of the block iterative methods, to the survey by Philippe, Saad and Stewart [17] for the use of Krylov-based methods, and

to the surveys by Schweitzer [19], [20] for the aggregation methods; see also [4], [5], [10], [14], [25], [26].

The idea of looking for an almost block diagonal form of a matrix representing a Markov chain is not new in itself. This has been done especially for nearly completely decomposable (NCD) matrices in many contexts; see, e.g., [4], [7], [12], [21], [22]. The new approach here is, on the one hand, the application to matrices that are not necessarily NCD, and on the other, the use of a different type of partitioning algorithm to achieve the almost block diagonal form. Furthermore, the algorithms discussed are very flexible in the sense that by a suitable choice of the input parameters, different blocks can be obtained, e.g., with denser blocks along the diagonal, or with entries within the diagonal blocks that are larger in magnitude. Another important property of the algorithms studied here is that their complexity is linear in the order of the matrix and the number of nonzeros. In other words, in general, the time required to find the partition adds very little time to the overall computation needed to find the stationary probability distribution.

The algorithm PABLO was presented by O’Neil and Szyld [16] with the goal of permuting sparse matrices so that the diagonal blocks are dense, i.e., that they have a large proportion of nonzeros. They used the new permuted matrices to improve the performance of classical block iterative methods for nonsingular systems, i.e., when the spectral radius of the iteration matrix is less than one. Here we apply the algorithm to iteration matrices whose spectral radius is one, and obtain good improvements of convergence as well. This algorithm is combinatorial in nature, i.e., it takes into account the zero-nonzero structure or graph of the matrix, but not the values of its entries. In Section 2 we review the algorithm PABLO, and we describe in detail two other variants which use a threshold to determine, depending on the magnitude of the entries, if the

corresponding row and column should be added to the current block along the diagonal. Since the threshold algorithms have an additional search through several nodes, their complexity is higher than that of PABLO, but it is still linear in the number of nonzeros and the order of the matrix, as shown in Section 2.

In Section 3 we present some numerical experiments. We have found, first, that the ordering generated by the original PABLO is a good partition for the matrices corresponding to Markov chains problems. Furthermore, the orderings produced by the threshold algorithms are obtained in similar time, and generally give rise to even faster convergence of classical iterative block methods. The performance of the block diagonal preconditioner obtained after the matrix is permuted and partitioned with the new threshold algorithm is much better than when the blocks are taken from the original orderings. They are also competitive with other preconditioners used for these problems, e.g., in [17].

We have also used the partitions as inputs to aggregation / disaggregation methods. It turns out, that with a certain choice of parameters the partitions generated by the new algorithms are the same as that generated in the package MARCA due to W. Stewart [23], [25]. With the use of some other parameters, the partitions generated provide an aggregation scheme with which the aggregation / disaggregation methods converge in less computational time; see Section 3.

In the rest of this section we briefly review some concepts related to the classical block iterative methods. Let  $B$  be the column stochastic matrix of interest. Let  $A = I - B$ . We are looking for the positive column vector  $\pi$  such that  $A\pi = 0$ , normalized so that  $\pi^T e = 1$ , where  $e^T = (1, 1, \dots, 1)$ . Any of the classical block methods can be described by a splitting of  $A$  of the form  $A = M - N$ , where  $M$  is nonsingular. The matrix  $T = M^{-1}N$  is called the iteration matrix of the method. Since  $B$  is column stochastic, the spectral radius  $\rho(B) = 1$ ,  $A$  is singular, and this implies that  $\rho(T) = 1$ . Given an initial guess  $\pi_0$ , the iteration is  $\pi_k = \alpha_k T \pi_{k-1}$ , where  $\alpha_k \in \mathbb{R}$  provides the appropriate normalization. The method converges if  $T$  satisfies certain properties, and the asymptotic rate of convergence is governed by the magnitude of the second largest eigenvalue, i.e., by the quantity  $\delta(T) = \max\{|\lambda|, \lambda \in \sigma(T), \lambda \neq 1\}$ , where  $\sigma(T)$  is the set of eigenvalues of  $T$ ; see, e.g., [2], [3], [13], [15], [24], for more details.

## 2 Threshold Orderings

In this section we review the algorithm PABLO (PARAMETRIZED BLOCK ORDERING) and describe the

threshold variants. Given an  $n \times n$  matrix  $A = (a_{ij})$ , let  $G = (V, E)$  be its associated graph, i.e.,  $V = \{v_1, \dots, v_n\}$  is the set of  $n$  vertices and  $E$  is the set of edges, where  $(v_i, v_j) \in E$  if and only if  $a_{ij} \neq 0$ ; see, e.g., [8], [9], [28]. Given this graph, PABLO constructs  $q$  section subgraphs  $G_k = (V_k, E_k)$ ,  $k = 1, \dots, q$ , where  $E_k \subset E$ ,  $k = 1, \dots, q$ , and the sets  $V_k$ ,  $k = 1, \dots, q$  are a partition of  $V$ , i.e., they are disjoint and  $V = \cup_{k=1}^q V_k$ . The number of subgraphs,  $q$ , i.e., the number of the corresponding diagonal blocks, is not known a priori, but is determined by the algorithm and it depends on the structure of the graph  $G$  and the input parameters. In the algorithm PABLO (as well as in the threshold algorithms described in this section), a first node is taken from the set of unmarked nodes and this node starts a new current set of vertices  $P$ . This first node is chosen to be one of minimum degree. Then additional nodes are taken from the queue of nodes adjacent to nodes in  $P$ , and added to the set  $P$  if they satisfy certain criteria, or sent back to the set of unmarked nodes if not.

Two criteria are used in PABLO to determine if a node  $v$  should be added to a current set  $P \subset V$ , corresponding to a diagonal block, by measuring how full  $v \cup P$  is, and to what degree the vertices in  $v \cup P$  are connected to each other. The first criterion is measured by the ratio of the total number of edges corresponding to  $v \cup P$  to the number of edges that subgraph would have if it were complete (corresponding to a full submatrix). In other words, if  $\phi_P$  is the ratio corresponding to the set  $P$ , the algorithm tests if  $\phi_{v \cup P} \geq \alpha \phi_P$ , where  $\alpha > 0$  is a user-provided parameter. If it is satisfied then the new node is added. The second test is that the new node  $v$  must be adjacent to at least a certain proportion of nodes in the subgraph corresponding to  $P$  and more than outside the subgraph. This proportion,  $0 \leq \beta \leq 1$ , is also provided by the user. In [16], it is recommended to have default values  $\alpha = 1$ ,  $\beta = 0.5$ , or to be reset by the user. The linearity of the algorithm is obtained by selecting a set of eligible nodes, those adjacent to nodes in  $P$ , thus restricting the search to a relatively small set of nodes. For further details, see [16].

In the two variants of threshold PABLO (TPABLO) presented here, a third additional criterion is used to decide if a new vertex  $v$  is added to the current subgraph being formed. Let  $\gamma \geq 0$  be the given threshold, let  $P$  be the set of nodes of the current subgraph, and  $v_j$  be the vertex being tested for addition to  $P$  (corresponding to the  $j$ th row and column of the matrix). In each of the TPABLO versions,  $v_j$  is added to  $P$  if, in addition to either of the two criteria in PABLO, the

```

Given a set  $C$ , set  $q = 0$ 
repeat until  $C = \emptyset$ 
  let  $P = Q = \emptyset$ 
  repeat until number of nodes in  $P \geq minbs$ 
    choose from  $C$  a node  $c$  of minimum degree, mark it, and add it to  $P$ 
    move to  $Q$  all nodes in  $C$  adjacent to  $c$ 
    repeat until  $Q = \emptyset$ 
      chose the node  $p$  from the head of  $Q$ 
      calculate connectivity information
      if either the fullness or the connectivity criteria of PABLO is satisfied
      and the threshold criterion holds (1 or 2, depending of the algorithm)
      then
        mark  $p$  and move it to  $P$ 
        add to the rear of  $Q$  all nodes in  $C$  adjacent to  $p$ 
        update connectivity information
      else
        move node  $p$  from  $Q$  to  $C$ 
      endif
    designate those nodes in  $P$  to be in a block (and set  $q = q + 1$ )

```

Figure 1: Pseudo-code for TPABLO

following holds.

Criterion 1.  $|a_{ij}| > \gamma$  or  $|a_{ji}| > \gamma$  for at least one  $i \in P$ .

Criterion 2.  $|a_{ij}| > \gamma$  and  $|a_{ji}| > \gamma$  for all  $i \in P$ .

The use of criterion 1 produces a permuted matrix in which every entry in the off-diagonal blocks is smaller than the threshold in absolute value. This criterion is also useful to find NCD matrices; see, e.g., [4], [11], [25]. The use of criterion 2 produces a permuted matrix in which every entry in the diagonal block is larger than the threshold in absolute value, with the possibility that some entries in the off-diagonal blocks are also larger than the threshold in absolute value. In addition, in order to avoid extremely small blocks, a parameter *minbs* with the minimum block size is introduced. If a block with fewer nodes is produced, the algorithm continues to add nodes to the block. A value of *minbs* = 0 has no effect on the algorithm, while a very large value is obviously not recommended. We should point out that if *minbs* > 0, this implies the following upper bound for the number of blocks  $q \leq n/minbs$ .

These two versions of TPABLO, called TPABLO1 and TPABLO2, respectively, are particularly useful for classical block iterative methods, such as block Jacobi and Gauss Seidel, as well as to provide good block diagonal preconditioners for Krylov-based methods. They are also useful to provide the partitions for an iterative aggregation method, also called aggrega-

tion/disaggregation method; see Section 3 for numerical experiments.

The pseudo-code in Figure 1 summarizes a portion of the algorithm TPABLO, cf. [16]. In it, the set  $C \subset V$  is the set of vertices which have yet not been assigned to a block, the set  $P \subset V$  is the current set, i.e., the one corresponding to a diagonal block being constructed, and the set  $Q \subset V$  is the set of nodes in  $C$  which are adjacent to nodes in  $P$ . Thus, at the beginning of the algorithm, we have  $C = V$ ,  $P = Q = \emptyset$ .

The complexity of PABLO is  $O(n + \nu)$ , where  $\nu$  is the number of nonzeros in  $A$ , i.e., the number of edges in  $E$  [16]. Let us analyze the added complexity introduced by the additional criterion. Each nonzero in the matrix is searched to compare its value with the threshold  $\gamma$  at least once, adding computing time proportional to  $\nu$ . In addition, if a node  $v_j$  is not added to the current set  $P$ , it is sent back to the set  $C$  and searched again, say when looking at another node adjacent to it. This rejection can happen at most  $d$  times, where  $d$  is the maximum degree of a node in  $E$ , i.e., the maximum number of nonzeros in any given row or column. Thus, the total additional time is at most proportional to  $(1+d)\nu$ . The algorithms studied are designed for large sparse matrices, and thus we expect  $d$  to be small, namely  $d \ll n$ , and therefore the complexity of TPABLO is also  $O(n + \nu)$ , though with a larger constant. In practice, the complexity

	order	$\gamma$	ngr	$\omega$	it	Tp	Ta	Tb	Tt	rnorm
TPABLO1	1771	0.01	21	1.5	4	0.27	0.00	2.36	9.45	1.38e-09
MARCA		0.01	21	1.5	5	0.02	0.00	2.34	11.80	2.80e-09
TPABLO1	5456	0.05	496	1.3	4	0.55	2.32	0.28	10.82	5.07e-09
		0.01	31	1.5	4	1.75	0.00	17.06	68.41	5.06e-10
TPABLO2		0.05	496	1.3	4	0.57	2.36	0.27	10.97	5.07e-09
		0.01	31	1.5	4	1.75	0.00	17.06	68.41	5.06e-10
MARCA		0.05	496	1.3	5	0.05	2.39	0.28	13.88	2.53e-10
		0.01	31	1.5	5	0.12	0.00	17.43	87.40	8.57e-10
TPABLO1	23426	0.05	1326	1.3	7	3.25	18.03	2.37	147.06	2.31e-11
		0.01	51	1.4	5	20.45	0.03	206.25	1032.39	8.64e-11
MARCA		0.05	1326	1.3	6	0.37	17.93	2.32	125.53	2.17e-10
		0.01	51	1.4	6	0.55	0.03	213.84	1284.43	8.24e-11

Table 1: Aggregation/Disaggregation for NCD matrices,  $\alpha = \beta = 0.5$ ,  $minbs = 0$

order	$\gamma$	ngr	$\omega$	iter	Tp	Ta	Tb	Tt	rnorm
1771	0.01	19	1.5	16	0.28	0.00	2.39	38.45	9.61e-09
			1.6	8	0.30	0.00	2.47	19.90	9.72e-09
			1.7	5	0.28	0.00	2.38	11.95	1.01e-09
5456	0.01	29	1.5	4	1.68	0.00	15.97	64.10	6.84e-11

Table 2: Aggregation/Disaggregation using TPABLO1 with  $\alpha = \beta = 0$ ,  $minbs = 10$

of the algorithm is as low as PABLO's as it can be appreciated in the numerical experiments presented in Section 3. This is in part because the criterion for threshold is tested after either of two criterions in PABLO has been satisfied, in other words, not every vertex needs to be tested for the given threshold.

### 3 Numerical experiments

We present numerical experiments illustrating how the permutations generated by the two versions of TPABLO are useful to find stationary probability vectors of Markov chains. We begin by comparing the ordering and permutations obtained with these, to those generated by the algorithm provided in the package MARCA due to W. Stewart. We will call the latter algorithm MARCA for short. In it, given a threshold, all elements of magnitude below it are discarded, and the strongly connected components of the remaining graph are determined; see [23], [25], for a detailed description of this partitioning algorithm. The matrices used in these experiments correspond to an interactive computer system, described as Example 1 in [17] or in [25]. The resulting NCD matrices corresponding to 20, 30 and 50 users have 1771, 5,456 and 23,426 states (number of variables) and 11,011, 35,216, and 156,026 nonzeros, respectively.

In Table 1 and in the tables that follow it, the parameter  $\gamma$  is the threshold used in the MARCA algorithm and in the two versions of TPABLO, "ngr"

is the number of aggregation groups, i.e., the number of blocks along the diagonal after the permutation,  $\omega$  is the relaxation parameter used in the SOR method for the solution of the linear system corresponding to each block if its order is larger than 50 (Gaussian elimination is used for the smaller blocks), and "it" is the number of aggregation and disaggregation steps needed for convergence. The times reported are CPU seconds of a SUN Microsystems Sparc 20 at the Department of Mathematics at Temple University. "Tp" is the time for the ordering algorithm, either MARCA or one of the versions of TPABLO, "Ta" is the average solution time for the aggregated matrix, "Tb" is the average solution time for the diagonal blocks, while "Tt" is the total CPU time for convergence. In the last column we present the Euclidean norm of the residual of the computed probability vector.

We observe that with the parameters  $\alpha = \beta = 0.5$ , TPABLO1 and TPABLO2 obtain the same groups of states as MARCA does, but the order of the variables (states) within each block is different, and also, the blocks appear in different order. This accounts for the difference in performance of the methods with the different partitions. In other words, the SOR method converges faster for the blocks in the diagonal (of order larger than 50) with the ordering produced by the TPABLOs. As can be observed, for these NCD matrices, the times obtained with the TPABLOs are of the

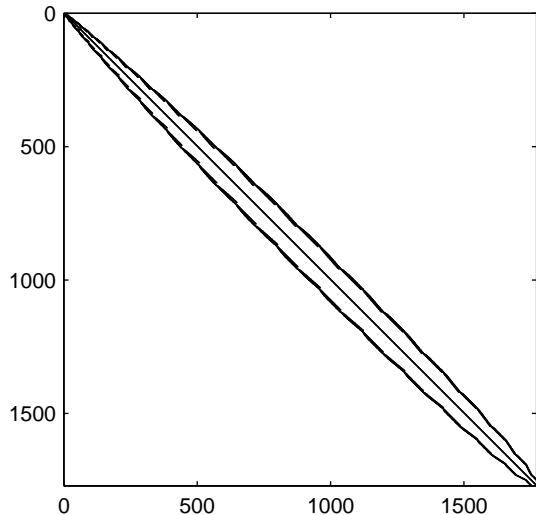


Figure 2: NCD Matrix of order 1771 in its original order

same order of magnitude, and in general better than those obtained with the MARCA partition. Furthermore, as can be seen in Table 2, different PABLO connectivity parameters give rise to different partitions which can produce, in some instances, better convergence time.

To illustrate the effect of the partition algorithm TPABLO we present in Figures 2 and 3 the zero-nonzero structure of the NCD matrix of order 1771 (20 users) in its original order, and permuted by TPABLO1, respectively.

While the MARCA algorithm was specifically created for NCD matrices, and thus is not expected to produce good partitionings for non-NCD Markov chains, the threshold PABLO versions consider other connectivity criteria, and thus can perform reasonably well for aggregation / disaggregation, even for non-NCD matrices (a proof that aggregation / disaggregation converges for non-NCD matrices can be found in [14]). For example, in Table 3 we consider the infinitesimal generator matrix of the continuous-time Markov chain underlying the Generalized Stochastic Petri Net model described in [1], which studies the processor-memory interference in a multiprocessor system. The matrix has 2,680 states and 32,775 nonzeros, and was produced with the package SPNP [6].

In Table 4 we present results on experiments using GMRES( $k$ ) [18], i.e., with restarts every  $k$  GMRES iterations. We use the same NCD matrices as in Table

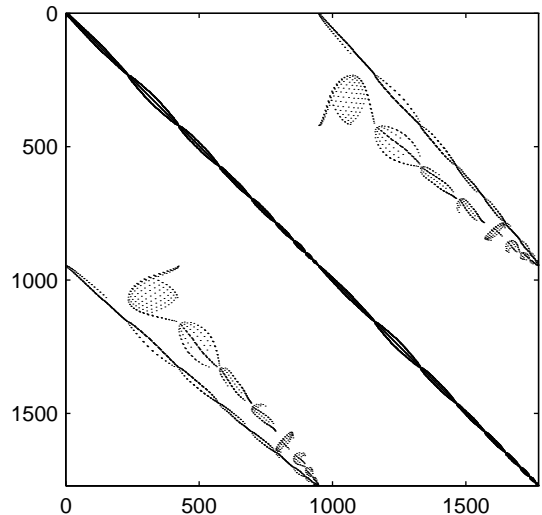


Figure 3: Matrix from Figure 2 after TPABLO permutation

1 with 1,771 and 23,426 states. We use three different preconditioners: (a) none, i.e., GMRES with no preconditioner, (b) ILU(0), i.e., incomplete factorizations as done, e.g., in [17], and (c) block diagonal preconditioners with the blocks obtained with TPABLO1 using  $\gamma = 0.01$ . We should point out that, in our experiments, GMRES with no preconditioner, as well as with block diagonal preconditioners obtained from the natural (original) order of the matrices, fails to converge. In the larger example, the same was the case for the incomplete factorization preconditioner. As can be appreciated, the block diagonal preconditioner performs very well. The time in parenthesis is the time for the factorization of the blocks in the diagonal (included in the total time). Of course the use of the block diagonal preconditioner implies the use of additional storage. For the case  $n = 1,771$ , the original matrix has 11,011 nonzeros, the diagonal blocks obtained by TPABLO has 7,591 nonzeros (69% of the total), and the factors of the diagonal blocks have 27,414 nonzeros. The corresponding numbers for the case  $n = 23,426$  are 156,026, 105,686 (68%), and 595,407.

We turn now to numerical experiments using Block Gauss Seidel. We compare the results obtained with PABLO and its threshold variants with the standard (point) Gauss Seidel, and to block versions obtained by taking 2 by 2 and 4 by 4 blocks along the diagonal. In addition to CPU time, we report the total number of operations, as well as  $\delta(T)$ , the second largest eigen-

Ordering	order	$\gamma$	ngr	iter	Tp	Ta	Tb	Tt	rnorm
TPABLO1	2680	0.001	235	11	4.07	0.50	0.09	7.07	8.27e-09
MARCA		0.001	456	4	0.15	1.92	53.81	223.26	5.36e-09

Table 3: Example ‘ProcMem20’ by Aggregation/Disaggregation ( $\alpha = 1.0$ ,  $\beta = 0.5$ ,  $minbs = 10$ )

order	preconditioner	no. of blocks	restart	no. of iter	time	rnorm
1771	None		10/15			no convergence
	TPABLO1(0.01)	40	10	160	9.60(0.97)	7.27e-09
			15	88	5.27(0.97)	6.34e-09
			20	77	4.37(0.97)	6.25e-09
	ILU(0)		10	243	4.55	9.78e-09
			15	255	5.89	8.70e-09
20			254	6.23	8.72.e-09	
23426	None/ILU(0)		10/15/20			no convergence
	TPABLO1(0.01)	199	15	350	569.96(63.51)	9.10e-09
			20	238	400.33(63.51)	8.29e-09

Table 4: GMRES( $k$ ) for NCD matrices,  $\alpha = \beta = 0.0$ ,  $minbs = 0$

value of the corresponding iteration matrix. “Tc” is the total computing time, excluding the time for the partition algorithm. As before “Tp” is the time for the partition algorithm (PABLO or TPABLO). The example in Table 5 is the infinitesimal generator matrix of the continuous-time Markov chain corresponding to the well-known “central server” queuing network model, described, e.g., in [27], with 1,771 states and 9,240 nonzeros. It was also produced with the package SPNP [6]. The example in Table 6 is the same as the one used in Table 3. It can be readily appreciated that the partitions generated by PABLO and the two versions of TPABLO provide better performance, combined with a block method, than point Gauss-Seidel, as well as better than the 2 by 2 and 4 by 4 blocks.

In order to illustrate how the threshold parameter may capture different proportion of the nonzeros in a matrix, we conclude with Table 7, where we report the percentage of nonzeros larger in magnitude than certain selected values, for four of the examples used in our experiments.

The experiments presented show that PABLO and its variants can be part of effective tools for finding stationary probability distributions of Markov chains.

### Acknowledgements.

We wish to thank Gianfranco Ciardo and William Stewart for generously providing us with the data used for our experiments. We also thank William Stewart for sending us an advance copy of [11], and Michele Benzi and Gianfranco Ciardo for their comments on

an earlier version of the manuscript. In addition, we thank the anonymous referees, whose suggestions have led to several improvements of the presentation. This work was supported by National Science Foundation grants DMS-9201728 and DMS-9625865.

### References

- [1] Marco Ajmone-Marsan, Gianfranco Balbo, and Giovanni Conte, “A class of Generalized Stochastic Petri Nets for the performance evaluation of multiprocessor systems,” *ACM Transactions on Computer Systems*, Vol. 2, pp. 93–122, 1984.
- [2] Vincent A. Barker, “Numerical solution of sparse singular systems of equations arising from ergodic Markov chains,” *Communications in Statistics, Stochastic Models*, Vol. 5, pp. 355–381, 1989.
- [3] Abraham Berman and Robert J. Plemmons, *Non-negative Matrices in the Mathematical Sciences*, Academic Press, New York, third edition, 1979, reprinted by SIAM, Philadelphia, 1994.
- [4] Wei-Lu Cao and William J. Stewart, “Iterative aggregation/disaggregation techniques for nearly uncoupled Markov chains,” *Journal of the Association for Computing Machinery*, Vol. 32, pp. 702–719, 1985.
- [5] Françoise Chatelin, “Iterative aggregation / disaggregation methods,” in G. Iazeolla, P. J. Courtois, and A. Hordijk, editors, *Mathematical Computer Performance and Reliability*, pp. 199–207,

	no. of blocks	no. of iter	total no. of operations	$\delta$	Tp	Tc	rnorm
Point GS	1771	467	27927365	.9677		13.35	9.87e-09
2 by 2	886	467	29577664	.9567		13.24	9.87e-09
4 by 4	443	467	32878821	.9460		13.14	9.87e-09
PABLO	539	268	17350803	.9445	.09	7.30	9.79e-09
TPABLO1(0.05)	599	226	14150797	.9320	.11	5.96	9.86e-09
TPABLO1(0.01)	541	266	17268875	.9438	.11	7.24	9.61e-09
TPABLO1(0.001)	539	268	17350803	.9445	.09	7.08	9.79e-09
TPABLO2(0.05)	624	208	12847129	.9255	.13	5.88	9.53e-09

Table 5: Example ‘CentralServer20’,  $\alpha=1.0$ ,  $\beta=0.5$ ,  $minbs = 2$

	no. of blocks	no. of iter	total no. of operations	$\delta$	Tp	Tc	rnorm
Point GS	2680	35	5793401	.3106		2.18	9.32e-09
2 by 2	1340	32	5477537	.2984		1.69	8.50e-09
4 by 4	443	28	5106098	.2817		1.47	6.40e-09
PABLO	825	18	3190312	.1923	4.14	0.97	4.42e-09
TPABLO1(0.05)	847	17	3031902	.1541	4.43	0.88	5.30e-09
TPABLO1(0.01)	832	18	3176405	.1917	4.33	0.93	5.33e-09

Table 6: Example ‘ProcMem20’,  $\alpha=1.0$ ,  $\beta=0.5$ ,  $minbs = 2$

North-Holland, Amsterdam – New York – Oxford, 1984.

- [6] Gianfranco Ciardo, Kishor S. Trivedi, and Jyoti Muppala, “SPNP: stochastic Petri net package,” in *Proceedings of the Third International Workshop on Petri Nets and Performance Models (PNPM’89)*, pp. 142–151, Kyoto, Japan, 1989, IEEE Computer Society Press.
- [7] Pierre-Jackes Courtois, *Decomposability, Queuing and Computer System Applications*, Academic Press, New York – San Francisco – London, 1977.
- [8] Iain S. Duff, Albert M. Erisman, and John K. Reid, *Direct Methods for Sparse Matrices*, Clarendon Press, Oxford, 1986.
- [9] Alan George and Joseph W. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.
- [10] Moshe Haviv, “Aggregation / disaggregation methods for computing the stationary distribution of a Markov chain,” *SIAM Journal on Numerical Analysis*, Vol. 14, pp. 952–966, 1987.
- [11] Richard L. Klevans and William J. Stewart, “From queuing networks to Markov chains: The XMARCA interface,” *Performance Evaluation*, Vol. 24, pp. 23–45, 1995.
- [12] Scott T. Leutenegger and Graham Horton, “On the utility of the multi-level algorithm for the solution of nearly completely decomposable Markov chains,” in William J. Stewart, editor, *Computations with Markov Chains: Proceedings of the 2nd International Workshop on the Numerical Solution of Markov Chains*, pp. 425–442, Kluwer Academic, Dordrecht, 1995.
- [13] Ivo Marek and Daniel B. Szyld, “Iterative and semi-iterative methods for computing stationary probability vectors of Markov operators,” *Mathematics of Computation*, Vol. 61, pp. 719–731, 1993.
- [14] Ivo Marek and Daniel B. Szyld, “Local convergence of the (exact and inexact) iterative aggregation method for linear systems and Markov operators,” *Numerische Mathematik*, Vol. 69, pp. 61–82, 1994.
- [15] Carl D. Meyer and Robert J. Plemmons, “Convergence powers of a matrix with applications to iterative methods for singular systems,” *SIAM Journal on Numerical Analysis*, Vol. 14, pp. 699–705, 1977.

Example	order	$\nu$	$\gamma$	$> \gamma$	$> 0.1$	$> .01$	$> 10^{-3}$	$> 10^{-4}$	$> 10^{-5}$
NCD-1	1771	11011	.05	44	44	72	88	98	100
NCD-2	5456	35216	.01	72	44	72	90	98	100
c20	1771	9240	.05	73	68	81	100		
p20	2680	32775	.001	75	25	43	75	94	97

Table 7: Percentage of nonzero elements of magnitude above certain values

- [16] James O’Neil and Daniel B. Szyld, “A block ordering method for sparse matrices,” *SIAM Journal on Scientific and Statistical Computing*, Vol. 11, pp. 811–823, 1990.
- [17] Bernard Philippe, Yousef Saad, and William J. Stewart, “Numerical methods in Markov chain modeling,” *Operations Research*, Vol. 40, pp. 1156–1179, 1992.
- [18] Yousef Saad and Martin H. Schultz, “GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems,” *SIAM Journal on Scientific and Statistical Computing*, Vol. 7, pp. 856–869, 1986.
- [19] Paul J. Schweitzer, “Aggregation methods for large Markov chains,” in G. Iazeolla, P. J. Courtois, and A. Hordijk, editors, *Mathematical Computer Performance and Reliability*, pp. 275–286, North-Holland, Amsterdam – New York – Oxford, 1984.
- [20] Paul J. Schweitzer, “A survey of aggregation-disaggregation in large Markov chains,” in William J. Stewart, editor, *Numerical Solution of Markov Chains*, pp. 63–88, Marcel Dekker, New York - Basel - Hong Kong, 1991.
- [21] G. W. Stewart, W. J. Stewart, and D. F. McAllister, “A two-stage iteration for solving nearly uncoupled Markov chains,” in G. Golub, A. Greenbaum, and M. Luskin, editors, *Recent Advances in Iterative Methods*, IMA Volumes in Mathematics and its Applications, Vol. 60, pp. 201–216, Springer Verlag, New York – Berlin, 1994.
- [22] G.W. Stewart and G. Zhang, “On a direct method for the solution of nearly uncoupled Markov chains,” *Numerische Mathematik*, Vol. 59, pp. 1–11, 1991.
- [23] William J. Stewart, “MARCA: Markov chain analyzer,” in William J. Stewart, editor, *Numerical Solution of Markov Chains*, pp. 37–61, Marcel Dekker, New York - Basel - Hong Kong, 1991.
- [24] William J. Stewart, *Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, Princeton, New Jersey, 1994.
- [25] William J. Stewart and Wei Wu, “Numerical experiments with iteration and aggregation for Markov chains,” *ORSA Journal on Computing*, Vol. 4, pp. 336–350, 1992.
- [26] Daniel B. Szyld, “Local convergence of (exact and inexact) iterative aggregation,” in Carl D. Meyer and Robert J. Plemmons, editors, *Linear Algebra, Markov Chains and Queuing Models*, IMA Volumes in Mathematics and its Applications, Vol. 48, pp. 137–143, Springer Verlag, New York – Berlin, 1993.
- [27] Kishor S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.
- [28] Richard S. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, 1962.