Componentwise bounds for nearly completely decomposable Markov chains using stochastic comparison and reordering*

Nihal Pekergin [a], Tuğrul Dayar [b], Denizhan N. Alparslan [c]

[a] PR*i*SM, Université de Versailles-St.Quentin, 45 av. des Etats Unis, 78035 France, and CERMSEM, Université de Paris I, 106-112 Bld de l'Hopital, 75647 France
Nihal.Pekergin@prism.uvsq.fr

[b] Department of Computer Engineering, Bilkent University, 06533 Bilkent, Ankara, Turkey
tugrul@cs.bilkent.edu.tr

[c] Computer Science Telecommunications, University of Missouri - Kansas City, 454 Robert H. Flarsheim Hall, 5100 Rockhill Road, Kansas City, Missouri 64110, USA
dna5a0@umkc.edu

Abstract

This paper presents an improved version of a componentwise bounding algorithm for the state probability vector of nearly completely decomposable Markov chains, and on an application it provides the first numerical results with the type of algorithm discussed. The given two-level algorithm uses aggregation and stochastic comparison with the strong stochastic (st) order. In order to improve accuracy, it employs reordering of states and a better componentwise probability bounding algorithm given st upper- and lower-bounding probability vectors. A thorough analysis of the two-level algorithm from the point of view of irreducibility is provided. Results in sparse storage show that there are cases in which the given algorithm proves to be useful.

Keywords: Markov processes; Near complete decomposability; Stochastic comparison; Reorderings; Aggregation.

*This work is supported by TÜBİTAK-CNRS joint grant.

1 Introduction

Nearly completely decomposable (NCD) Markov chains (MCs) [2], [12], [18] are irreducible stochastic matrices that can be symmetrically permuted [4] to the block form

$$P_{n\times n} = \begin{array}{c} \begin{pmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,N} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{N,1} & P_{N,2} & \cdots & P_{N,N} \end{pmatrix} \end{array} \begin{array}{c} n_1 \\ n_2 \\ \vdots \\ n_N \end{array} \tag{1}$$

in which nonzero elements of the off-diagonal blocks are small compared with those of the diagonal blocks [18, p. 286]. Let $P = \mathrm{diag}(P_{1,1}, P_{2,2}, \ldots, P_{N,N}) + F$. The diagonal blocks $P_{i,i}$ are square, of order n_i, with $n = \sum_{i=1}^{N} n_i$. The quantity $\|F\|_\infty$ is referred to as the degree of coupling and is taken to be a measure of the decomposability of P. When the chain is NCD, it has eigenvalues close to 1, and the poor separation of the unit eigenvalue implies a slow rate of convergence for standard matrix iterative methods [7, p. 290]. Hence, NCD Markov chains are said to be ill-conditioned [12, p. 258]. On the other hand, if P were reducible, we would decompose the chain into its irreducible (i.e., isolated) and transient subclasses of states as in equation (1.20) of [18, p. 26] and continue our analysis on the irreducible subclasses.

Such matrices arise in queuing network analysis, large-scale economic modeling, and computer systems performance evaluation. The measures of interest for these systems may be obtained either from the long-run distribution of state probabilities by solving a homogeneous system of linear equations with a singular coefficient matrix under a normalization constraint (i.e., steady state analysis), or from the state probability distribution at a particular time instant by solving a set of first order ordinary differential equations using an initial state probability distribution (i.e., transient analysis).

To each NCD MC corresponds an irreducible coupling matrix [12], C, whose (i,j)th element is given by

$$c_{i,j} = \frac{\pi_i}{\|\pi_i\|_1} P_{i,j} e \qquad \forall i,j \in \{1,2,\ldots,N\}. \tag{2}$$

Here e represents a column vector of all ones and the steady state probability (row) vector π is partitioned conformally with P in equation (1) such that $\pi = (\pi_1, \pi_2, \ldots, \pi_N)$, where each π_i is a row vector having n_i elements. The coupling matrix shows the evolution of the system when each NCD partition is treated as a single aggregated state. In other words, C describes the coupling among NCD partitions, and its steady state vector gives the steady state probability of being in each NCD partition. Note that one needs to know π to compute C exactly.

For the partitioning in equation (1), the stochastic complement [12] of $P_{i,i}$ for $i \in \{1,2,\ldots,N\}$ is given by

$$\overline{P}_{i,i} = P_{i,i} + P_{i,:}(I - P_i)^{-1} P_{:,i},$$

where $P_{i,:}$ is the $n_i \times (n - n_i)$ matrix composed of the ith row of blocks of P with $P_{i,i}$ removed, $P_{:,i}$ is the $(n - n_i) \times n_i$ matrix composed of the ith column of blocks of P with $P_{i,i}$ removed, and P_i is the $(n - n_i) \times (n - n_i)$ principal submatrix of P with ith row and

ith column of blocks removed. The ith stochastic complement is the stochastic transition probability matrix of an irreducible MC of order n_i obtained by observing the original process in the ith NCD partition. The conditional steady state probability vector of the ith NCD partition is $\pi_i/\|\pi_i\|_1$, and it may be computed by solving for the steady state vector of $\overline{P}_{i,i}$ (see [12] for details). However, each stochastic complement has an embedded matrix inversion which may require excessive computation.

Stochastic comparison is a technique by which both transient and steady state performance measures of a MC may be bounded. There are several applications of this technique in different areas of applied probability [16] and in practical problems of engineering [13], [14], [9]. The stochastic comparison of MCs is discussed in detail in [10], [19], [11]. The comparison of two MCs may be established by the comparison of their transient probability vectors at each time instant. Obviously, if steady states exist, stochastic comparison between their steady state probability vectors is also possible.

Sufficient conditions for the existence of stochastic comparison of two time-homogeneous MCs are given by the stochastic monotonicity and bounding properties of their one step transition probability matrices [10], [11]. In [20], this idea is used to devise an algorithm that constructs an optimal st-monotone upper-bounding MC. Later, this algorithm is used to compute stochastic bounds on performance measures that are defined on a totally ordered and reduced state space [1]. Performance measures may be defined as reward functions of the underlying MC. In [1], states having the same reward are aggregated, so the state space size of the bounding MC is considerably reduced. However, the given algorithm may provide loose bounds in the general case. If the state space reduction (i.e., aggregation) procedure takes into account the dynamics of the underlying system, it is possible to provide tight stochastic bounds. Another way to provide tight bounds is to consider specific matrix structures which are suitable to aggregation procedures.

In [21], a componentwise bounding algorithm for the state probability vector of NCD MCs is given. The two-level algorithm of Truffet in [21] uses aggregation and stochastic comparison with the strong stochastic (st) order. The algorithm is different from the bounded aggregation method discussed in [3], [15] in that a smaller number of linear systems of about the same order as those in [3] are solved at the cost of lower accuracy. The algorithm of Courtois and Semal in [3] uses polyhedra theory to compute the best possible bounds for a given NCD MC. To the best of our knowledge, the algorithm of Truffet has not been implemented and tested on any applications yet. Furthermore, its theoretical analysis lacks essential components. In this paper, we present an improved and coherent version of the algorithm, and remedy the situation regarding analysis and implementation. We remark that even though the presentation of the algorithm in this paper is for computing the steady state probability vector of an NCD MC, it can also be used to carry out transient analysis with some modification. The bounding techniques in [3] and [21] both have tradeoffs. This work is not intended to be a comparative study between them, but it rather aims to develop a better understanding of stochastic comparison as a useful tool in performance analysis. We also remark that a continuous-time MC (CTMC) can be transformed through uniformization [18, p. 24] to a discrete-time MC. Hence, the algorithm presented in this paper may be used to compute bounds on the state probability vector of a CTMC.

In section 2, we provide background on stochastic comparison. In section 3, we introduce the improved algorithm and demonstrate the effect of the improvements on a small NCD MC. A thorough analysis of the new algorithm from the point of view of irreducibility

appears in section 4. In section 5, we provide numerical results in sparse storage on a current application in mobile communications. Therein, we also discuss an application from the same research area which does not favor the algorithm. In section 6, we conclude.

2 Background on stochastic comparison

There are different stochastic ordering relations and the most well known is the strong stochastic ordering (i.e., $\leq_{st}$). Intuitively speaking, two random variables X and Y which take values on a totally ordered space being comparable in the strong stochastic sense (i.e., $X \leq_{st} Y$) means that it is less probable for X to take larger values than Y (see [16], [19]).

First we give the definition of st-ordering used in this paper. For further information on the stochastic comparison method, we refer the reader to [19].

Definition 1 *Let X and Y be random variables taking values on a totally ordered space. Then X is said to be less than Y in the strong stochastic sense, that is, $X \leq_{st} Y$ iff*

$$E[f(X)] \leq E[f(Y)]$$

for all nondecreasing functions f whenever the expectations exist.

Definition 2 *Let X and Y be random variables taking values on the finite state space $\{1, 2, \ldots, n\}$. Let p and q be probability vectors such that*

$$p_i = Prob(X = i) \quad and \quad q_i = Prob(Y = i) \quad for \quad i \in \{1, 2, \ldots, n\}.$$

Then X is said to be less than Y in the strong stochastic sense, that is, $X \leq_{st} Y$ iff

$$\sum_{i=j}^{n} p_i \leq \sum_{i=j}^{n} q_i \quad for \quad j = n, n-1, \ldots, 1.$$

The comparison of MCs has been largely studied in [10], [19], [11]. We use the following definition (Definition 4.1.2 of [19, p. 59]) to compare MCs.

Definition 3 *Let $\{X(t),\ t \in \mathcal{T}\}$ and $\{Y(t),\ t \in \mathcal{T}\}$ be two time-homogeneous MCs. Then $\{X(t),\ t \in \mathcal{T}\}$ is said to be less than $\{Y(t),\ t \in \mathcal{T}\}$ in the strong stochastic sense, that is, $\{X(t)\} \leq_{st} \{Y(t)\}$ iff*

$$X(t) \leq_{st} Y(t) \qquad \forall t \in \mathcal{T}.$$

It is shown in Theorem 3.4 of [11, p. 355] that monotonicity and comparability of the probability transition matrices of time-homogeneous MCs yield sufficient conditions for their stochastic comparison, which is summarized in:

Theorem 1 *Let P and $\tilde{P}$ be stochastic matrices respectively characterizing time-homogeneous MCs $X(t)$ and $Y(t)$. Then $\{X(t),\ t \in \mathcal{T}\} \leq_{st} \{Y(t),\ t \in \mathcal{T}\}$ if*

- $X(0) \leq_{st} Y(0)$,

- *st-monotonicity of at least one of the probability transition matrices holds, that is,*

$$\textit{either} \quad P_{i,*} \leq_{st} P_{j,*} \quad \textit{or} \quad \tilde{P}_{i,*} \leq_{st} \tilde{P}_{j,*} \quad \forall i,j \quad \textit{such that} \;\; i \leq j,$$

- *st-comparability of the transition matrices holds, that is,*

$$P_{i,*} \leq_{st} \tilde{P}_{i,*} \qquad \forall i.$$

Here $P_{i,*}$ refers to row i of P.

3 Componentwise bounding algorithm

The componentwise bounding algorithm for the steady state vector of NCD MCs we present (see Algorithm 1) is based on the two-level algorithm in [21] that uses aggregation and stochastic comparison with the st-order.

3.1 The improved algorithm

Algorithm 1 Componentwise bounding algorithm for the steady state vector of NCD MCs:

0. Find a (balanced) NCD partitioning of P and symmetrically permute it to the form in equation (1). Let $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N\}$ be the resulting state space partition.

1. for $i = 1, 2, \ldots, N$,

   a. Choose a state from $\mathcal{S}_i$, say f_i, make it the last state and find the ordering of the remaining states in $\mathcal{S}_i$ with respect to f_i by the heuristic algorithm in Step 3.a of [5, p. 241]. Symmetrically permute $P_{i,i}$ according to the resulting ordering.

   b. Compute the two stochastic matrices $\overline{S}_i$ and $\underline{S}_i$ of order n_i corresponding to $P_{i,i}$ by Algorithms 2 and 3, respectively (see Remark 1).

   c. Compute the st-monotone upper-bounding matrix $\overline{Q}_i$ of order n_i corresponding to $\overline{S}_i$ by Algorithm 5 and the st-monotone lower-bounding matrix $\underline{Q}_i$ of order n_i corresponding to $\underline{S}_i$ by Algorithm 6.

   d. Extract the irreducible submatrices of $\overline{Q}_i$ and $\underline{Q}_i$ and solve the corresponding systems of equations for their steady state vectors $\overline{\pi}_i^{st}$ and $\underline{\pi}_i^{st}$, respectively. Place zero steady state probabilities for transient states in each vector.

   e. Compute the componentwise bounding vectors π_i^{sup} and π_i^{inf} on the conditional steady state probability vector corresponding to $\mathcal{S}_i$ from $\overline{\pi}_i^{st}$ and $\underline{\pi}_i^{st}$ by Algorithm 7.

2. a. Compute U and L of order N using π_i^{sup} and π_i^{inf}, $i \in \{1, 2, \ldots, N\}$, by Algorithms 8 and 9, respectively.

   b. Compute the two stochastic matrices $\overline{S}$ and $\underline{S}$ of order N corresponding to L and U by Algorithms 2 and 3, respectively.

c. Compute the st-monotone upper-bounding matrix $\overline{Q}$ of order N corresponding to $\overline{S}$ by Algorithm 5 and the st-monotone lower-bounding matrix $\underline{Q}$ of order N corresponding to $\underline{S}$ by Algorithm 6.

d. Extract the irreducible submatrices of $\overline{Q}$ and $\underline{Q}$ and solve the corresponding systems of equations for their steady state vectors $\overline{\xi}^{st}$ and $\underline{\xi}^{st}$, respectively. Place zero steady state probabilities for transient states in each vector.

e. Compute the componentwise bounding vectors ξ^{sup} and ξ^{inf} on the steady state probability vector corresponding to C from $\overline{\xi}^{st}$ and $\underline{\xi}^{st}$ by Algorithm 7.

3. Compute the componentwise steady state probability upper- and lower-bounding vectors for $\mathcal{S}_i$ respectively as $\xi_i^{sup}\pi_i^{sup}$ and $\xi_i^{inf}\pi_i^{inf}$, $i \in \{1, 2, \ldots, N\}$.

Remark 1 *When Algorithms 2 and 3 are invoked for the substochastic matrices $P_{i,i}$, $L = P_{i,i}$ and $U = L + \Delta$, where $d = e - Le$ and $\Delta = [d\ d\ \cdots\ d]$.*

Algorithm 2 Construction of stochastic matrix $\overline{S}$ corresponding to L and U of order m:

$\Delta = U - L$;
for $i = 1, 2, \ldots, m$,
  $\epsilon_i^{(0)} = 1 - \sum_{j=1}^{m} l_{i,j}$;
for $i = 1, 2, \ldots, m$,
  for $j = m, m-1, \ldots, 1$,
    $\overline{s}_{i,j} = l_{i,j} + \min(\delta_{i,j}, (\epsilon_i^{(m-j)})^+)$;
    $\epsilon_i^{(m-j+1)} = \epsilon_i^{(m-j)} - \delta_{i,j}$;

Algorithm 3 Construction of stochastic matrix $\underline{S}$ corresponding to L and U of order m:

$\Delta = U - L$;
for $i = 1, 2, \ldots, m$,
  $\epsilon_i^{(0)} = 1 - \sum_{j=1}^{m} l_{i,j}$;
for $i = 1, 2, \ldots, m$,
  for $j = 1, 2, \ldots, m$,
    $\underline{s}_{i,j} = l_{i,j} + \min(\delta_{i,j}, (\epsilon_i^{(j-1)})^+)$;
    $\epsilon_i^{(j)} = \epsilon_i^{(j-1)} - \delta_{i,j}$;

Algorithm 4 Construction of matrix B (to be used in Algorithms 5 and 6) corresponding to stochastic matrix S of order m:

for $i = 1, 2, \ldots, m$,
  $b_{i,m} = s_{i,m}$;
  for $j = m-1, m-2, \ldots, 1$,
    $b_{i,j} = b_{i,j+1} + s_{i,j}$;

Algorithm 5 Construction of st-monotone upper-bounding matrix $\overline{Q}$ corresponding to stochastic matrix $\overline{S}$ of order m:

Compute B by Algorithm 4 for $\overline{S}$ of order m.
$\overline{q}_{1,m} = b_{1,m}$;
for $i = 2, 3, \ldots, m$,
  $\overline{q}_{i,m} = \max(b_{i,m}, \overline{q}_{i-1,m})$;
for $l = m-1, m-2, \ldots, 1$,
  $\overline{q}_{1,l} = b_{1,l} - b_{1,l+1}$;
  for $i = 2, 3, \ldots, m$,
    $\overline{q}_{i,l} = \max(b_{i,l}, \sum_{j=l}^{m} \overline{q}_{i-1,j}) - \sum_{j=l+1}^{m} \overline{q}_{i,j}$;

Algorithm 6 Construction of st-monotone lower-bounding matrix $\underline{Q}$ corresponding to stochastic matrix $\underline{S}$ of order m:

Compute B by Algorithm 4 for $\underline{S}$ of order m.
for $l = 1, 2, \ldots, m-1$,
  $\underline{q}_{m,l} = b_{m,l} - b_{m,l+1}$;
  for $i = m-1, m-2, \ldots, 1$,
    $\underline{q}_{i,l} = \max(1 - b_{i,l+1}, \sum_{j=1}^{l} \underline{q}_{i+1,j}) - \sum_{j=1}^{l-1} \underline{q}_{i,j}$;
$\underline{q}_{m,m} = b_{m,m}$;
for $i = m-1, m-2, \ldots, 1$,
  $\underline{q}_{i,m} = 1 - \sum_{j=1}^{m-1} \underline{q}_{i,j}$;

Algorithm 7 Computation of componentwise probability bounding vectors v^{sup} and v^{inf} given st upper- and lower-bounding probability vectors $\overline{v}^{st}$ and $\underline{v}^{st}$ of length m:

$v_m^{sup} = \overline{v}_m^{st}$;
$v_m^{inf} = \underline{v}_m^{st}$;
for $j = m-1, m-2, \ldots, 1$,
  $v_j^{sup} = \sum_{k=j}^{m} \overline{v}_k^{st} - \sum_{k=j+1}^{m} \underline{v}_k^{st}$;
  $v_j^{inf} = (\sum_{k=j}^{m} \underline{v}_k^{st} - \sum_{k=j+1}^{m} \overline{v}_k^{st})^+$;

Algorithm 8 Computation of componentwise upper-bounding matrix U for C of order N using P and π_i^{sup}, $i \in \{1, 2, \ldots, N\}$:

for $i = 1, 2, \ldots, N$,
  for $j = 1, 2, \ldots, N$,
    $u_{i,j} = \min(\pi_i^{sup} P_{i,j} e, \max(P_{i,j} e))$;

Algorithm 9 Computation of componentwise lower-bounding matrix L for C of order N using P and π_i^{inf}, $i \in \{1, 2, \ldots, N\}$:

for $i = 1, 2, \ldots, N$,
  for $j = 1, 2, \ldots, N$,
    $l_{i,j} = \max(\pi_i^{inf} P_{i,j} e, \min(P_{i,j} e))$;

The preprocessing done in Step 0 of Algorithm 1 is self-descriptive. As suggested in Section 1, it is possible to use the algorithm in [4] to find NCD partitionings of P given a user specified decomposability parameter. By a balanced partitioning, we mean one in which the n_i, $i \in \{1, 2, \ldots, N\}$, in equation (1) do not differ significantly from each other. We argue why it is important to use balanced NCD partitionings after we analyze the complexity of Algorithm 1.

The ordering (i.e., numbering) of states in a MC affects the quality of bounds that may be obtained by the stochastic comparison approach [5] due to the conditions of st-monotonicity and st-comparability in Theorem 1. In order to obtain tighter probability bounds, Step 1.a of Algorithm 1 permutes one of the states within each NCD partition to be the last and orders the remaining states in the same partition using the heuristic given in [5, pp. 241–242]. The state to be permuted to the end of each NCD block is chosen as the state which has the largest self-transition probability among the states in the same NCD partition followed by a simple tie-breaking rule if needed. We do not reorder (aggregated) states in Step 2 of Algorithm 1 since the resulting matrices are highly diagonally dominant due to the NCD structure implying a small gain (if at all). Reordering of states is the first improvement over the algorithm in [21].

At the first level of Algorithm 1 (see Step 1), componentwise upper- and lower-bounds on the conditional steady state probability vector of each NCD partition are computed for the partitioning of P in equation (1). This is achieved in Steps 1.b and 1.c (see Algorithms 2–6) by computing st-monotone upper- and lower-bounding matrices for each stochastic complement. In Step 1.b, two stochastic matrices corresponding to the particular NCD block are obtained using Algorithms 2 and 3. The former (latter) of these matrices is computed by adding the off-diagonal block probability mass to the incoming transitions of the last (first) state in the NCD partition thereby ensuring that the resulting stochastic matrix satisfies the st-comparability relation greater (less) than or equal to with the stochastic complement (see Section 1) of the NCD block. The two stochastic matrices obtained in this way for each NCD partition are input to Step 1.c. In Step 1.c, we use the st-monotone upper-bounding matrix construction algorithm in [1] as in [21] (see Algorithm 5), but devise and use a new st-monotone lower-bounding matrix construction algorithm (see Algorithm 6) whose optimality is proved in Section 4. In [21], the st lower-bounding vector on the steady state distribution of a MC is computed by reversing the order of its states and running Algorithm 5 on the permuted MC. See [21, p. 847] for details. The new st-monotone lower-bounding matrix construction algorithm we present eliminates the need for a permutation vector to order the states of the input stochastic matrix in reverse.

Neither of the two st-monotone bounding matrices computed for each stochastic complement may be irreducible [1]. However, as we prove in the next section, both of these matrices have one irreducible subset of states. This is also true for the st-monotone bounding matrices computed for the coupling matrix, C (see Section 1), in Step 2. After identifying the transient states and removing them from each of the two st-monotone bounding matrices, the resulting irreducible stochastic matrices are solved for their steady state vectors in Step 1.d. This gives st upper- and lower-bounds on the conditional steady state probabilities of the particular NCD partition. In Step 1.e, componentwise bounds on the conditional steady state probability vector of the NCD partition are obtained from the st upper- and lower-bounding vectors using Algorithm 7. In Section 4, Algorithm 7 is shown to be better than Algorithm 10 used in [21]. This is the second improvement over the algorithm in [21].

At the second level (see Step 2), st-monotone upper- and lower-bounding matrices for C corresponding to the partitioning of P in equation (1) are computed using Algorithms 2–6, 8, and 9 in Steps 2.a–c. This is achieved by using the conditional steady state probability bounding vectors obtained for each NCD partition at the first level. Note that in Step 2.a, Algorithms 8 and 9 compute the matrices U and L which are respectively componentwise upper- and lower-bounding matrices for C. In Step 2.b, two stochastic matrices corresponding to C are obtained by Algorithms 2 and 3 using U and L. The former (latter) of these stochastic matrices is computed so as to satisfy the st-comparability relation greater (less) than or equal to with C. In Step 2.c, st-monotone bounding matrices corresponding to these two stochastic matrices are obtained. From the two st-monotone bounding matrices, two stochastic matrices corresponding to the irreducible subsets of states are extracted, and they are solved for their steady state vectors in Step 2.d. This gives st upper- and lower-bounds on the steady state probabilities of C. In Step 2.e, the st-bounding vectors obtained in the previous step are used by Algorithm 7 to compute componentwise bounds on the steady state probabilities of C.

Finally, in Step 3 of Algorithm 1, the componentwise upper-bounding (lower-bounding) vector on the conditional steady state distribution of each NCD partition is unconditioned by the corresponding upper-bound (lower-bound) on the steady state probability of C. Hence, we obtain componentwise bounds on the global steady state distribution, π.

We remark that Steps 1.d and 2.d should omit the removal of transient states and replace the steady solution process with a transient solution procedure when performing transient analysis.

The theoretical analysis of the algorithm in [21] lacks essential components. There is no mention of the existence of a single irreducible subset of states in the st-monotone upper-bounding matrix computed by Algorithm 5. The possibility of computing a reducible st-monotone bounding matrix for a given irreducible Markov chain is stated in [1]. For the proposed methodology, the existence of a single irreducible subset of states must be proved for the matrices obtained by Algorithms 5 and 6 at both levels. Furthermore, the componentwise bounding algorithm that takes in st upper- and lower-bounding probability vectors (see Algorithm 7) is superior to its counterpart:

Algorithm 10 Computation of componentwise probability bounding vectors w^{sup} and w^{inf} as in [21] given st upper- and lower-bounding probability vectors $\overline{v}^{st}$ and $\underline{v}^{st}$ of length m:

$$
\begin{aligned}
&w_m^{sup} = \overline{v}_m^{st};\\
&w_m^{inf} = \underline{v}_m^{st};\\
&\text{for } j = m-1, m-2, \ldots, 1,\\
&\quad w_j^{sup} = \min(1, (\textstyle\sum_{k=j}^{m} \overline{v}_k^{st} - \sum_{k=j+1}^{m} w_k^{inf})^+);\\
&\quad w_j^{inf} = \min(1, (\textstyle\sum_{k=j}^{m} \underline{v}_k^{st} - \sum_{k=j+1}^{m} w_k^{sup})^+);
\end{aligned}
$$

We address these theoretical issues in Section 4. Now we proceed with the complexity analysis of Algorithm 1.

3.2 Complexity analysis

We assume that P has nz nonzero elements distributed uniformly across the matrix. Note that nz is considered to be $O(n)$ for sparse matrices. Hence, there will be roughly $k = nz/n$ nonzero elements per row/column of P and $k_i = nz \times n_i/n^2$ nonzero elements per row/column of $P_{i,i}$. The uniform assumption is neither an optimistic nor a pessimistic one. Now, assuming that Algorithm 1 is implemented in sparse storage and a direct method is employed in solving the linear systems in Steps 1.d and 2.d, its space complexity other than the storage set aside for P is $\max\{O(nz), \max_i\{O(n_i^2), O(N^2)\}\}$ reals and integers from Steps 0, 1.b, 1.c, 2.b, and 2.c. Other steps contribute as lower order terms. As for the time complexity of the algorithm, we should account for floating-point comparisons and floating-point arithmetic operations separately. From Steps 0, 1.a–d, 2.a, and 2.d, we have $\max\{O(nk^2), \sum_{i=1}^{N} \max\{O(n_i^2), O(n_i k_i^2)\}, O(Nn), O(N^3)\}$ floating-point comparisons. From Steps 1.d, 2.a, and 2.d, we have $\max\{\sum_{i=1}^{N} O(n_i^3), O(nz),\ O(Nn), O(N^3)\}$ floating-point arithmetic operations. Other steps contribute as lower order terms. Now it is evident why one should opt for balanced NCD partitionings (cf. Step 0).

Now, let us compare the time complexity of Algorithm 1 with that of iterative aggregation-disaggregation (IAD), a method devised to compute the steady state vector of NCD MCs using successive approximations [18, Ch. 6]. In order to make a fair comparison, let us assume that both methods use the same NCD partitioning and that there is space to factorize its diagonal blocks at the outset [6]. We make the same assumption regarding sparsity as in Algorithm 1 and conclude that there are roughly $\sum_{i=1}^{N} n_i(k - k_i)$ nonzero elements in the off-diagonal blocks of P. Each iteration of IAD consists of two steps. In the aggregation step, an approximate coupling matrix is formed and solved for its steady state vector. In order to expedite the computation of this matrix, row sums of each block in the NCD partitioning are computed and stored at the outset. In the disaggregation step, a block Gauss-Seidel (BGS) iteration is performed. This requires the solution of N nonsingular linear systems whose coefficient matrices are the diagonal blocks factorized at the outset and whose right hand sides are computed using the nonzero elements in the off-diagonal blocks, the steady state vector of the approximate coupling matrix, and the previous steady state approximation. Hence, there are $\max\{\sum_{i=1}^{N} O(n_i^3), O(nz)\}$ floating-point arithmetic operations at the outset. The aggregation and disaggregation steps respectively cost $\max\{O(N^3), O(Nn)\}$ and $\max\{O(\sum_{i=1}^{N} n_i(k - k_i)), \sum_{i=1}^{N} O(n_i^2)\}$ floating-point arithmetic operations per IAD iteration. The number of iterations taken by IAD to converge to a tolerance of ϵ is $O(\log \epsilon / \log \|F\|_\infty)$ since each iteration of IAD reduces the error in the approximate solution by a factor of degree of coupling (see Theorem 6.6 in [18, p. 340]). It is clear that Algorithm 1 runs faster than IAD as long as the coefficient matrix is dense and the degree of coupling is not exceedingly small.

Now we show how Algorithm 1 executes on a small example.

3.3 An example

Consider the 8×8 Courtois matrix [2]

$$P = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \left(\begin{array}{ccc|cc|ccc} 0.85 & 0 & 0.149 & 0.0009 & 0 & 0.00005 & 0 & 0.00005 \\ 0.1 & 0.65 & 0.249 & 0 & 0.0009 & 0.00005 & 0 & 0.00005 \\ 0.1 & 0.8 & 0.0996 & 0.0003 & 0 & 0 & 0.0001 & 0 \\ \hline 0 & 0.0004 & 0 & 0.7 & 0.2995 & 0 & 0.0001 & 0 \\ 0.0005 & 0 & 0.0004 & 0.399 & 0.6 & 0.0001 & 0 & 0 \\ \hline 0 & 0.00005 & 0 & 0 & 0.00005 & 0.6 & 0.2499 & 0.15 \\ 0.00003 & 0 & 0.00003 & 0.00004 & 0 & 0.1 & 0.8 & 0.0999 \\ 0 & 0.00005 & 0 & 0 & 0.00005 & 0.1999 & 0.25 & 0.55 \end{array}\right)$$

whose steady state vector is given by

$$\pi = [0.089283, 0.092758, 0.040488, 0.158533, 0.118938, 0.120385, 0.277795, 0.101819].$$

In Step 0, we choose a degree of decomposability [4] of 0.001 and obtain the state space partitioning $\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$, where $\mathcal{S}_1 = \{1,2,3\}$, $\mathcal{S}_2 = \{4,5\}$ and $\mathcal{S}_3 = \{6,7,8\}$. This is an NCD partitioning with $\|F\|_\infty = 0.001$. The NCD blocks are

$$P_{1,1} = \begin{pmatrix} 0.85 & 0 & 0.149 \\ 0.1 & 0.65 & 0.249 \\ 0.1 & 0.8 & 0.0996 \end{pmatrix}, P_{2,2} = \begin{pmatrix} 0.7 & 0.2995 \\ 0.399 & 0.6 \end{pmatrix}, P_{3,3} = \begin{pmatrix} 0.6 & 0.2499 & 0.15 \\ 0.1 & 0.8 & 0.0999 \\ 0.1999 & 0.25 & 0.55 \end{pmatrix}.$$

In Step 1.a, states 1, 4, and 7 are chosen as the last states in their corresponding NCD blocks. Given these last states, the heuristic algorithm in [5] returns the orderings (3,2,1), (5,4), and (6,8,7) for NCD blocks 1, 2, and 3, respectively. When the $P_{i,i}$ for $i \in \{1,2,3\}$ are symmetrically permuted with respect to these orderings, they become

$$P_{1,1} = \begin{array}{c} 3 \\ 2 \\ 1 \end{array} \begin{pmatrix} 0.0996 & 0.8 & 0.1 \\ 0.249 & 0.65 & 0.1 \\ 0.149 & 0 & 0.85 \end{pmatrix}, \quad P_{2,2} = \begin{array}{c} 5 \\ 4 \end{array} \begin{pmatrix} 0.6 & 0.399 \\ 0.2995 & 0.7 \end{pmatrix},$$

$$P_{3,3} = \begin{array}{c} 6 \\ 8 \\ 7 \end{array} \begin{pmatrix} 0.6 & 0.15 & 0.2499 \\ 0.1999 & 0.55 & 0.25 \\ 0.1 & 0.0999 & 0.8 \end{pmatrix}.$$

Step 1.b computes two stochastic matrices for each of the (permuted) NCD blocks which are given by

$$\overline{S}_1 = \begin{pmatrix} 0.0996 & 0.8 & 0.1004 \\ 0.249 & 0.65 & 0.101 \\ 0.149 & 0 & 0.851 \end{pmatrix}, \quad \underline{S}_1 = \begin{pmatrix} 0.1 & 0.8 & 0.1 \\ 0.25 & 0.65 & 0.1 \\ 0.15 & 0 & 0.85 \end{pmatrix},$$

$$\overline{S}_2 = \begin{pmatrix} 0.6 & 0.4 \\ 0.2995 & 0.7005 \end{pmatrix}, \quad \underline{S}_2 = \begin{pmatrix} 0.601 & 0.399 \\ 0.3 & 0.7 \end{pmatrix},$$

$$\overline{S}_3 = \begin{pmatrix} 0.6 & 0.15 & 0.25 \\ 0.1999 & 0.55 & 0.2501 \\ 0.1 & 0.0999 & 0.8001 \end{pmatrix}, \quad \underline{S}_3 = \begin{pmatrix} 0.6001 & 0.15 & 0.2499 \\ 0.2 & 0.55 & 0.25 \\ 0.1001 & 0.0999 & 0.8 \end{pmatrix}.$$

Using these stochastic matrices, Step 1.c computes st-monotone upper- and lower-bounding matrices for each NCD partition which are given by

$$\overline{Q}_1 = \begin{pmatrix} 0.0996 & 0.8 & 0.1004 \\ 0.0996 & 0.7994 & 0.101 \\ 0.0996 & 0.0494 & 0.851 \end{pmatrix}, \ \underline{Q}_1 = \begin{pmatrix} 0.25 & 0.65 & 0.1 \\ 0.25 & 0.65 & 0.1 \\ 0.15 & 0 & 0.85 \end{pmatrix},$$

$$\overline{Q}_2 = \begin{pmatrix} 0.6 & 0.4 \\ 0.2995 & 0.7005 \end{pmatrix}, \ \underline{Q}_2 = \begin{pmatrix} 0.601 & 0.399 \\ 0.3 & 0.7 \end{pmatrix},$$

$$\overline{Q}_3 = \begin{pmatrix} 0.6 & 0.15 & 0.25 \\ 0.1999 & 0.55 & 0.2501 \\ 0.1 & 0.0999 & 0.8001 \end{pmatrix}, \ \underline{Q}_3 = \begin{pmatrix} 0.6001 & 0.15 & 0.2499 \\ 0.2 & 0.55 & 0.25 \\ 0.1001 & 0.0999 & 0.8 \end{pmatrix}.$$

We remark that both st-monotone bounding matrices corresponding to each NCD partition in the Courtois example turn out to be irreducible. In other words, they do not have any transient states.

Step 1.d solves the st-monotone bounding matrices for their steady state vectors, which are given by

$$\overline{\pi}_1^{st} = [0.099600, 0.496639, 0.403761], \ \underline{\pi}_1^{st} = [0.210000, 0.390000, 0.400000],$$

$$\overline{\pi}_2^{st} = [0.428163, 0.571837], \ \underline{\pi}_2^{st} = [0.429185, 0.570815],$$

$$\overline{\pi}_3^{st} = [0.240679, 0.203597, 0.555724], \ \underline{\pi}_3^{st} = [0.240882, 0.203616, 0.555502].$$

Using these vectors, Step 1.e computes componentwise bounds on the conditional steady state probabilities of each NCD partition as

$$\pi_1^{sup} = [0.210000, 0.500400, 0.403761], \ \pi_1^{inf} = [0.099600, 0.386239, 0.400000],$$

$$\pi_2^{sup} = [0.429185, 0.571837], \ \pi_2^{inf} = [0.428163, 0.570815],$$

$$\pi_3^{sup} = [0.240882, 0.203819, 0.555724], \ \pi_3^{inf} = [0.240679, 0.203393, 0.555502].$$

Since Step 1 of Algorithm 1 is over, Step 2 starts executing. In Step 2.a, the matrices U and L of order 3 are computed using π_i^{sup} and π_i^{inf}, $i \in \{1, 2, 3\}$, as

$$U = \begin{pmatrix} 0.999600 & 0.000877 & 0.000100 \\ 0.000615 & 0.999500 & 0.000100 \\ 0.000056 & 0.000044 & 0.999900 \end{pmatrix}, L = \begin{pmatrix} 0.999000 & 0.000737 & 0.000100 \\ 0.000614 & 0.999000 & 0.000100 \\ 0.000056 & 0.000044 & 0.999900 \end{pmatrix}.$$

In Step 2.b, the stochastic matrices $\overline{S}$ and $\underline{S}$ corresponding to L and U are computed as

$$\overline{S} = \begin{pmatrix} 0.999023 & 0.000877 & 0.000100 \\ 0.000614 & 0.999286 & 0.000100 \\ 0.000056 & 0.000044 & 0.999900 \end{pmatrix}, \ \underline{S} = \begin{pmatrix} 0.999163 & 0.000737 & 0.000100 \\ 0.000615 & 0.999285 & 0.000100 \\ 0.000056 & 0.000044 & 0.999900 \end{pmatrix}.$$

In Step 2.c, st-monotone upper- and lower-bounding matrices for C are computed as

$$\overline{Q} = \begin{pmatrix} 0.999023 & 0.000877 & 0.000100 \\ 0.000614 & 0.999286 & 0.000100 \\ 0.000056 & 0.000044 & 0.999900 \end{pmatrix}, \ \underline{Q} = \begin{pmatrix} 0.999163 & 0.000737 & 0.000100 \\ 0.000615 & 0.999285 & 0.000100 \\ 0.000056 & 0.000044 & 0.999900 \end{pmatrix}.$$

These two bounding matrices are also irreducible.

In Step 2.d, the st-monotone bounding matrices are solved for their steady state vectors, which are given by

$$\overline{\xi}^{st} = [0.210388, 0.289612, 0.500000], \ \underline{\xi}^{st} = [0.230836, 0.269164, 0.500000].$$

Using these vectors, in Step 2.e componentwise bounds on the steady state probabilities of C are computed as

$$\xi^{sup} = [0.230836, 0.289612, 0.500000], \ \xi^{inf} = [0.210388, 0.269164, 0.500000].$$

In Step 3, componentwise bounds on the steady state probabilities of each NCD partition are computed as

$$\xi_1^{sup}\pi_1^{sup} = [0.048476, 0.115510, 0.093203], \ \xi_1^{inf}\pi_1^{inf} = [0.020955, 0.081260, 0.084155],$$

$$\xi_2^{sup}\pi_2^{sup} = [0.124297, 0.165611], \ \xi_2^{inf}\pi_2^{inf} = [0.115246, 0.153643],$$

$$\xi_3^{sup}\pi_3^{sup} = [0.120441, 0.101910, 0.277862], \ \xi_3^{inf}\pi_3^{inf} = [0.120339, 0.101697, 0.277751].$$

Finally, the componentwise bounding vectors are permuted back to their original orderings, and we obtain componentwise upper- and lower-bounding vectors on π:

$$\pi^{sup} = [0.093203, 0.115510, 0.048476, 0.165611, 0.124297, 0.120441, 0.277862, 0.101910],$$

$$\pi^{inf} = [0.084155, 0.081260, 0.020955, 0.153643, 0.115246, 0.120339, 0.277751, 0.101697].$$

Compare the result of the improved algorithm with those of the following three cases:

(i) First improvement turned off (i.e., no reorderings used):

$$\pi^{sup} = [0.093817, 0.116272, 0.048795, 0.166606, 0.125044, 0.166694, 0.309165, 0.125083]$$

$$\pi^{inf} = [0.083459, 0.080588, 0.020781, 0.152774, 0.114594, 0.100000, 0.208222, 0.090835]$$

(ii) Second improvement turned off (i.e., Algorithm 10 used instead of Algorithm 7):

$$\pi^{sup} = [0.093277, 0.115602, 0.049383, 0.165698, 0.124363, 0.120552, 0.277862, 0.101910]$$

$$\pi^{inf} = [0.084094, 0.081201, 0.020149, 0.153538, 0.115168, 0.120228, 0.277751, 0.101697]$$

(iii) Both improvements turned off (i.e., basic algorithm):

$$\pi^{sup} = [0.128242, 0.124810, 0.052378, 0.168326, 0.126335, 0.200943, 0.309165, 0.125083]$$

$$\pi^{inf} = [0.059553, 0.079426, 0.020482, 0.143034, 0.107289, 0.065751, 0.208222, 0.090835]$$

After assessing the quality of the bounds, we conclude that the performance of Algorithm 1 on the Courtois example is extremely good, and it is superior to each of the three cases. However, the Courtois problem is small, and to have a better understanding of Algorithm 1, we must apply it to larger examples. Section 5 is dedicated to such a problem.

4 Analysis

In this section, we state various definitions, lemmas, and theorems showing why Algorithm 1 works. The discussion assumes matrices of order 2 or more. First, we introduce two types of stochastic matrices.

Definition 4 *A stochastic matrix A of order m that satisfies:*

(i) $\exists j \in \{2, 3, \ldots, m\}$ *such that* $a_{1,j} > 0$,

(ii) $\exists i \in \{1, 2, \ldots, m-1\}$ *such that* $a_{i,m} > 0$,

(iii) $\forall i \in \{1, 2, \ldots, m-1\}$ $\exists k \leq i$ *and* $\exists j > i$ *such that* $a_{k,j} > 0$

is called a type-1 stochastic matrix.

Definition 5 *A stochastic matrix A of order m that satisfies:*

(i) $\exists j \in \{1, 2, \ldots, m-1\}$ *such that* $a_{m,j} > 0$,

(ii) $\exists i \in \{2, 3, \ldots, m\}$ *such that* $a_{i,1} > 0$,

(iii) $\forall i \in \{2, 3, \ldots, m\}$ $\exists k \geq i$ *and* $\exists j < i$ *such that* $a_{k,j} > 0$

is called a type-2 stochastic matrix.

Lemma 1 *Let $\overline{S}_i$ be the stochastic matrix computed by Algorithm 2 for the submatrix $P_{i,i}$ of order n_i in Algorithm 1. Then $\overline{S}_i$ is a type-1 stochastic matrix.*

Lemma 2 *Let $\underline{S}_i$ be the stochastic matrix computed by Algorithm 3 for the submatrix $P_{i,i}$ of order n_i in Algorithm 1. Then $\underline{S}_i$ is a type-2 stochastic matrix.*

Proof. Let us prove Lemma 1. The proof of Lemma 2 is similar. The proof consists of showing that parts (i), (ii), and (iii) of Definition 4 hold for $\overline{S}_i$. Note that $\overline{S}_i$ (alternatively, $\underline{S}_i$) is $P_{i,i}$ with its last (alternatively, first) column perturbed. See Remark 1 and consider its implications on Algorithms 2 and 3. For ease of understanding, let us denote $P_{i,i}$ by Y, $\overline{S}_i$ by A, and n_i by m.

(i) There are two cases. If $\sum_{j=1}^{m} y_{1,j} = 1$, then $Y_{1,*} = A_{1,*}$ implying $\exists j \in \{2, 3, \ldots, m\}$ such that $a_{1,j} > 0$, otherwise state 1 would be absorbing contradicting the fact that P is irreducible. If $\sum_{j=1}^{m} y_{1,j} < 1$, then by Algorithm 2 we have $\epsilon_1^{(0)} > 0$ and $\delta_{1,m} > 0$ implying $a_{1,m} > 0$. Hence, $\exists j \in \{2, 3, \ldots, m\}$ such that $a_{1,j} > 0$.

(ii) In Y, it is not possible to have $y_{i,m} = 0$ and $\sum_{j=1}^{m} y_{i,j} = 1$ $\forall i \in \{1, 2, \ldots, m-1\}$, otherwise P would be reducible. There are two cases. Suppose for a row $i \in \{1, 2, \ldots, m-1\}$, we have $y_{i,m} > 0$. Then $a_{i,m} > 0$. On the other hand, suppose for a row $i \in \{1, 2, \ldots, m-1\}$, we have $\sum_{j=1}^{m} y_{i,j} < 1$. Then by Algorithm 2, $\epsilon_i^{(0)} > 0$ and $\delta_{i,m} > 0$ implying $a_{i,m} > 0$. Hence, $\exists i \in \{1, 2, \ldots, m-1\}$ such that $a_{i,m} > 0$.

(iii) Let l be the smallest row index among $i \in \{1, 2, \ldots, m-1\}$ for which $a_{i,m} > 0$. From part (ii), there exists such an l. By considering the particular values $k = l$ and $j = m$, for each $i \in \{l, l+1, ..., m-1\}$ $\exists k \leq i$ and $\exists j > i$ such that $a_{k,j} > 0$. Since for each $i \in \{1, 2, ..., l-1\}$, $y_{i,m} = 0$ and $\sum_{j=1}^{m} y_{i,j} = 1$, the irreducibility of P implies that for each $i \in \{1, 2, ..., l-1\}$ $\exists k \leq i$ and $\exists j > i$ such that $a_{k,j} > 0$. □

Lemma 3 *Let $\overline{S}$ be the stochastic matrix computed by Algorithm 2 for the componentwise upper- and lower-bounding coupling matrices U and L of order N in Algorithm 1. Then $\overline{S}$ is a type-1 stochastic matrix.*

Lemma 4 *Let $\underline{S}$ be the stochastic matrix computed by Algorithm 3 for the componentwise upper- and lower-bounding coupling matrices U and L of order N in Algorithm 1. Then $\underline{S}$ is a type-2 stochastic matrix.*

Proof. Let us prove Lemma 3. The proof of Lemma 4 is similar. The proof consists of showing that parts (i), (ii), and (iii) of Definition 4 hold for $\overline{S}$. Note that, if C is the coupling matrix given by equation (2), then $L \leq C \leq U$ by construction (see Algorithms 8 and 9).

(i) Since C is irreducible, there is at least one column $j \in \{2, 3, \ldots, N\}$ such that $c_{1,j} > 0$. Now, there are two cases. When $l_{1,j} = 0$, we have $\epsilon_1^{(0)} > 0$ and $\delta_{1,j} > 0$ (since $L \leq C$) implying $\exists k \geq j$ $\overline{s}_{1,k} > 0$. When $l_{1,j} > 0$, we have $\overline{s}_{1,j} > 0$. Hence, $\exists j \in \{2, 3, \ldots, N\}$ such that $\overline{s}_{1,j} > 0$.

(ii) Since C is irreducible, there is at least one row $i \in \{1, 2, \ldots, N-1\}$ such that $c_{i,N} > 0$. Now, there are two cases. When $l_{i,N} = 0$, we have $\epsilon_i^{(0)} > 0$ and $\delta_{i,N} > 0$ (since $L \leq C$) implying $\overline{s}_{i,N} > 0$. When $l_{i,N} > 0$, we have $\overline{s}_{i,N} > 0$. Hence, $\exists i \in \{1, 2, \ldots, N-1\}$ such that $\overline{s}_{i,N} > 0$.

(iii) Since C is irreducible, for each row $i \in \{1, 2, \ldots, N-1\}$, $\exists k \leq i$ and $\exists j > i$ such that $c_{k,j} > 0$. Again there are two cases. For row i, $l_{k,j} = 0$ implies $\epsilon_k^{(0)} > 0$ and $\delta_{k,j} > 0$ (since $L \leq C$). Then $\exists l \geq j$ $\overline{s}_{k,l} > 0$. For row i, $l_{k,j} > 0$ implies $\overline{s}_{k,j} > 0$. Hence, for each row $i \in \{1, 2, \ldots, N-1\}$, $\exists k \leq i$ and $\exists j > i$ such that $\overline{s}_{k,j} > 0$. □

Lemma 5 *If the input matrix $\overline{S}$ to Algorithm 5 is a type-1 stochastic matrix of order m, then there is a path from each state $i \in \{1, 2, \ldots, m-1\}$ to state m in the output st-monotone upper-bounding matrix $\overline{Q}$.*

Lemma 6 *If the input matrix $\underline{S}$ to Algorithm 6 is a type-2 stochastic matrix of order m, then there is a path from each state $i \in \{2, 3, \ldots, m\}$ to state 1 in the output st-monotone lower-bounding matrix $\underline{Q}$.*

Proof. Let us prove Lemma 5. The proof of Lemma 6 is similar. Let l be the state with the smallest index in $\overline{S}$ such that $\overline{s}_{l,m} > 0$. Since $\overline{S}$ is a type-1 stochastic matrix, the existence of such an l is guaranteed by part (ii) of Definition 4. From Algorithm 5, $\overline{q}_{l,m} > 0$ as well. From the st-monotonicity of $\overline{Q}$, for each $i \in \{l, l+1, \ldots, m-1\}$ we have $\overline{q}_{i,m} > 0$ implying a path of length one from each state $e \in \{l, l+1, \ldots, m-1\}$ to state m in $\overline{Q}$. What remains

to be done is to show that there is a path from each state $e \in \{1, 2, \ldots, l-1\}$ to state m in $\overline{Q}$.

Now, let l_1 be the state with the largest index such that $\overline{s}_{1,l_1} > 0$. The existence of such an l_1 is guaranteed by part (i) of Definition 4. From Algorithm 5, $\overline{q}_{1,l_1} > 0$ since the first rows of $\overline{S}$ and $\overline{Q}$ are identical. From the st-monotonicity of $\overline{Q}$, for each $i \in \{2, 3, \ldots, m\}$ $\exists j \geq l_1$ it must be that $\overline{q}_{i,j} > 0$. Hence, there are two cases depending on the value of l_1.

When $l_1 \geq l$, for each state $e \in \{2, 3, \ldots, l-1\}$ there is a direct transition to a state $j(\geq l_1)$ and a transition from state j to state m in $\overline{Q}$. When $l_1 < l$, consider part (iii) of Definition 4 and notice that $\exists i \leq l_1$ and $\exists k > l_1$ such that $\overline{s}_{i,k} > 0$. Since $\overline{Q}$ is an st-monotone upper-bounding matrix, this implies $\exists l_2 \geq k > l_1$ such that $\overline{q}_{i,l_2} > 0$. Now, let i_1 be the state with the smallest index among $i(\leq l_1)$ corresponding to the state with the largest index l_2; then $\overline{q}_{i_1,l_2} > 0$. From the st-monotonicity of $\overline{Q}$, for each $i \in \{i_1, i_1+1, \ldots, m\}$ $\exists j \geq l_2$ it must be that $\overline{q}_{i,j} > 0$. Again, there are two cases.

When $e \geq i_1$, there is a direct transition from state e to a state $e'(\geq l_2 > l_1)$ in $\overline{Q}$. When $e < i_1$, there is a direct transition from state e to a state $j(\geq l_1)$. Since $i_1 \leq i \leq l_1$, we have $j \geq i_1$ implying the existence of direct transitions from state e to a state j and from state j to a state e'. Then we must observe the value of l_2. If $l_2 \geq l$, we are at the very first case. If $l_2 < l$, we must continue the recursive analysis as above until l_2 becomes larger than l. Note that $l_2 > l_1$ and l_2 will eventually exceed l. □

Theorem 2 *If the input matrix $\overline{S}$ to Algorithm 5 is a type-1 stochastic matrix of order m, then there is a single irreducible (sub)set of states that includes state m in the given ordering of states in the output st-monotone upper-bounding matrix $\overline{Q}$.*

Theorem 3 *If the input matrix $\underline{S}$ to Algorithm 6 is a type-2 stochastic matrix of order m, then there is a single irreducible (sub)set of states that includes state 1 in the given ordering of states in the output st-monotone lower-bounding matrix $\underline{Q}$.*

Proof. Let us prove Theorem 2. The proof of Theorem 3 is similar. Since $\overline{Q}$ is not necessarily irreducible, it may have several classes of states (see [18, p. 26]). Let us denote these classes by C_j and the class which contains state m as C_l. Now, if C_l is an irreducible class, then from each class C_j, $j \neq l$, there must be a path to C_l. This follows from Lemma 5 and the fact that the classes C_j form an exact partition of the state space. Furthermore, if C_l is an irreducible class, it is not possible to leave C_l. Suppose C_l is transient and there is path from C_l to C_j for some $j \neq l$. Then C_l and C_j are equivalent, which contradicts the fact that C_l and C_j are distinct classes. Hence, C_l must be irreducible. Note that C_j cannot be an irreducible class since there is a path to C_l. □

The optimality of the st-monotone upper-bounding matrix computed by Algorithm 5 is proved in [1, pp. 12–14]. Here we give the proof for the st-monotone lower-bounding matrix computed by Algorithm 6.

Theorem 4 *Let $\underline{Q}$ be the st-monotone lower-bounding matrix computed by Algorithm 6 for the stochastic matrix $\underline{S}$ of order m. Let $\underline{R}$ be another st-monotone lower-bounding matrix for $\underline{S}$. Then $\underline{Q}$ is optimal in the sense that $\underline{R} \leq_{st} \underline{Q}$.*

Proof. The proof is by induction. By construction, row m in $\underline{Q}$ and $\underline{S}$ are identical (see Algorithm 6). Since $\underline{R}$ is another st-monotone lower-bounding matrix for $\underline{S}$, it must be that

$\underline{R}_{m,*} \leq_{st} \underline{Q}_{m,*} = \underline{S}_{m,*}$. This forms the basis step. Now, let us assume that $\underline{R}_{i,*} \leq_{st} \underline{Q}_{i,*}$ for $i \in \{l, l+1, \ldots, m-1\}$. This forms the induction hypothesis. We must show that $\underline{R}_{l-1,*} \leq_{st} \underline{Q}_{l-1,*}$.

Since $\underline{R}$ is st-monotone, $\underline{R}_{l-1,*} \leq_{st} \underline{R}_{l,*}$. Since it is an st-monotone lower-bound for $\underline{S}$, we have $\underline{R}_{l,*} \leq_{st} \underline{S}_{l,*}$. Then from the induction hypothesis, $\underline{R}_{l-1,*} \leq_{st} \underline{R}_{l,*} \leq_{st} \underline{Q}_{l,*}$. Observe that the inequality $\underline{R}_{l-1,*} \leq_{st} \underline{Q}_{l-1,*}$ to be proven is equivalent to

$$\sum_{j=1}^{k} \underline{r}_{l-1,j} \geq \sum_{j=1}^{k} \underline{q}_{l-1,j} \qquad \forall k \in \{1, 2, \ldots, m\}.$$

Now, we analyze Algorithm 6 to see how the elements of $\underline{Q}$ are computed. In the algorithm,

$$\sum_{j=1}^{k} \underline{q}_{l-1,j} = \max(1 - b_{l-1,k+1}, \sum_{j=1}^{k} \underline{q}_{l,j}) \qquad \forall k \in \{1, 2, \ldots, m-1\}$$

(when $k = m$, all row sums are 1). We remark that

$$1 - b_{l-1,k+1} = \sum_{j=1}^{k} \underline{s}_{l-1,j}$$

from Algorithm 4; therefore, the $(1 - b_{l-1,k+1})$ argument of max is due to comparison with matrix $\underline{S}$ for a lower-bound. The second argument of max is due to comparison with row l of $\underline{Q}$ for st-monotonicity. Hence, there are two cases for each $k \in \{1, 2, \ldots, m-1\}$. We either have

$$\sum_{j=1}^{k} \underline{q}_{l-1,j} = \sum_{j=1}^{k} \underline{s}_{l-1,j} \quad \text{implying} \quad \sum_{j=1}^{k} \underline{r}_{l-1,j} \geq \sum_{j=1}^{k} \underline{q}_{l-1,j}$$

since $R \leq_{st} S$, or we have

$$\sum_{j=1}^{k} \underline{q}_{l-1,j} = \sum_{j=1}^{k} \underline{q}_{l,j} \quad \text{again implying} \quad \sum_{j=1}^{k} \underline{r}_{l-1,j} \geq \sum_{j=1}^{k} \underline{q}_{l-1,j}$$

since $R_{l-1,*} \leq_{st} Q_{l,*}$. Combining the two cases, we obtain

$$\sum_{j=1}^{k} \underline{r}_{l-1,j} \geq \sum_{j=1}^{k} \underline{q}_{l-1,j} \qquad \forall k \in \{1, 2, \ldots, m\}$$

implying $\underline{R}_{l-1,*} \leq_{st} \underline{Q}_{l-1,*}$. $\square$

Lemma 7 *Algorithm 7 computes better componentwise probability bounds than the Algorithm 10 used in [21].*

Proof. The proof is by induction. We must show that $v^{sup} \leq w^{sup}$ and $v^{inf} \geq w^{inf}$. By construction (see Algorithms 7 and 10), $v_m^{sup} = w_m^{sup} = \overline{v}_m^{st}$ and $v_m^{inf} = w_m^{inf} = \underline{v}_m^{st}$. This is the basis step. Now, let us assume that $v_j^{sup} \leq w_j^{sup}$ and $v_j^{inf} \geq w_j^{inf}$ for $j \in \{l, l+1, \ldots, m-1\}$. This forms the induction hypothesis. We must show that $v_{l-1}^{sup} \leq w_{l-1}^{sup}$ and $v_{l-1}^{inf} \geq w_{l-1}^{inf}$.

From Algorithms 7 and 10, we respectively have

$$v_{l-1}^{sup} = v_l^{sup} + \overline{v}_{l-1}^{st} - \underline{v}_l^{st} \qquad \text{and} \qquad w_{l-1}^{sup} = w_l^{sup} + \overline{v}_{l-1}^{st} - w_l^{inf}.$$

From the induction hypothesis, we have $v_l^{sup} \leq w_l^{sup}$ and $v_l^{inf} \geq w_l^{inf}$. Observing that $\underline{v}_l^{st} \geq v_l^{inf}$ from Algorithm 7, we obtain $v_{l-1}^{sup} \leq w_{l-1}^{sup}$.

Similarly, from Algorithms 7 and 10, we respectively have

$$v_{l-1}^{inf} = v_l^{inf} + \underline{v}_{l-1}^{st} - \overline{v}_l^{st} \qquad \text{and} \qquad w_{l-1}^{inf} = w_l^{inf} + \underline{v}_{l-1}^{st} - w_l^{sup}.$$

Observing that $\overline{v}_l^{st} \leq v_l^{sup}$ and using the induction hypothesis, we obtain $v_{l-1}^{inf} \geq w_{l-1}^{inf}$. □

5 Numerical results

The implementation of the algorithms in Section 3 is done in compact sparse row (CSR) Harwell-Boeing format which requires for each coefficient matrix of order m one real and one integer array of size nz_m (i.e., number of nonzero elements in the coefficient matrix), one integer array of size $(m+1)$, and temporary workspace to accommodate fill-in during factorization. All code is written in Fortran/C and compiled in double precision with $g77/gcc$ on a SUN UltraSparcstation 10 with 128 MBytes of RAM running Solaris 2.6. The numerical experiments are timed using a C function that reports CPU time. Since the resulting NCD MCs are of moderate order (i.e., thousands of states) and sparsity (i.e., tens of nonzeros per row), we consider the direct solution method of Grassmann–Taksar–Heyman (GTH) [8] at each level of Algorithm 1. This method is a more robust version of Gaussian elimination (GE) in which arithmetic with only positive numbers is performed [7].

We compare the run-time of Algorithm 1 with that of GTH and IAD [18] which are both geared towards NCD MCs. In order to make a fair comparison, with IAD we use the same partitionings as in Algorithm 1. For all combinations of the integer parameters we considered, there is sufficient space to factorize in sparse format (that is, to apply sparse GE to) the diagonal blocks in IAD. Furthermore, we use BGS in the disaggregation step and employ a stopping tolerance of 10^{-15} on the infinity norm of the residual vector at each iteration. We remark that for each problem solved, the relative backward error in IAD turns out to be less than 10^{-16}. See [6] for recent results on the computation of the steady state vector of Markov chains.

The application that we consider arises in wireless asynchronous transfer mode (ATM) networks. In [22], a multiservices resource allocation policy (MRAP) is developed to integrate two types of service over time division multiple access (TDMA) frames in a mobile communication environment. These are the constant bit rate (CBR) service for two types of voice calls (i.e., handover calls from neighboring cells and new calls) and the available bit rate (ABR) service for data transfer. A single cell and single carrier frequency is modeled. However, the arrival process of data and the service process of calls we consider is quite general and subsumes the model in [22].

The TDMA frame is assumed to have C slots. Handover requests have priority over new call arrivals and they respectively arrive with probabilities p_h and p_n. Each voice call

takes up a single slot of a TDMA frame but may span multiple TDMA frames whereas each data packet is served in a single slot of a single TDMA frame. When all the slots are full, incoming voice calls are rejected. The number of voice calls that may terminate in a given TDMA frame depends on the number of active calls and is modeled as a binomial process with parameter p_s. The parameters of the model are $p_h = C \times 10^{-5}$, $p_n = C \times 5 \times 10^{-6}$, and $p_s = C \times 5 \times 10^{-6}$.

Data is queued in a FIFO buffer of size B and has the least priority. The arrival of data packets is modeled as an on-off process. The process moves from the on state to the off state with probability α and from the off state to the on state with probability β. The load offered to the system is defined as $L = \beta/(\alpha + \beta)$. Assuming that the time interval between two consecutive on periods is t, the burstiness of such an on-off process is described by the square coefficient of variation, $S_C = Var(t)/[E(t)]^2$. In terms of L and S_C, we have $\beta = 2L(1-L)/(S_C + 1 - L)$ and $\alpha = \beta(1-L)/L$. When the on-off process is in the on state, we assume that $i \in \{0, 1, 2, 3\}$ data packets may arrive with probability p_{di}. The mean arrival rate of data packets in the on state is defined as $R = \sum_{i=1}^{3} i \times p_{di}$. Hence, the global mean arrival rate of data packets is given by $G = L \times R$. We set $(p_{d0}, p_{d1}, p_{d2}, p_{d3}) = (0.4, 0.3, 0.2, 0.1)$ implying $R = 1.0$. When the buffer is full, any excess packet is dropped. We do not consider the arrival of multiple handovers or multiple new calls during a TDMA frame duration. Observe that there is orders of magnitude between the average interarrival time of voice calls and the average interarrival time of data packets, which makes this problem NCD. In fact, the smallest degree of coupling values we computed for this problem are on the order of 10^{-4}.

The performance measures of interest are the blocking probability of voice calls and the dropping probability of data packets. If the underlying MC is represented by a three-component state descriptor (a, b, c), where a denotes the state of the data arrival process, b denotes the number of data packets in the buffer and c denotes the number of active voice calls, then the blocking probability of voice calls is given by

$$\begin{aligned} p_{block} &= [(p_n(1-p_h) + (1-p_n)p_h + 2p_np_h)(1-p_s)^C \sum_{i=0}^{1}\sum_{j=0}^{B} \pi_{i,j,C} \\ &\quad + p_np_hC(1-p_s)^{C-1}p_s \sum_{i=0}^{1}\sum_{j=0}^{B} \pi_{i,j,C} \\ &\quad + p_np_h(1-p_s)^{C-1} \sum_{i=0}^{1}\sum_{j=0}^{B} \pi_{i,j,C-1}]/[p_n(1-p_h) + (1-p_n)p_h + 2p_np_h] \end{aligned}$$

and the dropping probability of data packets is given by

$$p_{drop} = [(p_{d1}+2p_{d2}+3p_{d3}) \sum_{i=0}^{C} \pi_{1,B,i} + (p_{d2}+2p_{d3}) \sum_{i=0}^{C} \pi_{1,B-1,i} + p_{d3} \sum_{i=0}^{C} \pi_{1,B-2,i}]/[p_{d1}+2p_{d2}+3p_{d3}].$$

We remark that the above formulae is defined on the product state space having $2(B+1)(C+1)$ states of which some are unreachable.

In Figure 1, using Algorithm 1 we present bounds on the blocking probability of voice calls and the dropping probability of data packets in the system with $B = 30$, $C = 10$. We take take $L \in \{0.1, 0.2, \ldots, 0.9\}$ and $S_C \in \{1, 10, 100\}$. The NCD partitionings considered for $S_C = 1$ in Figures 1.(a)–(b) all have 11 blocks with orders between 42 and 62, and a

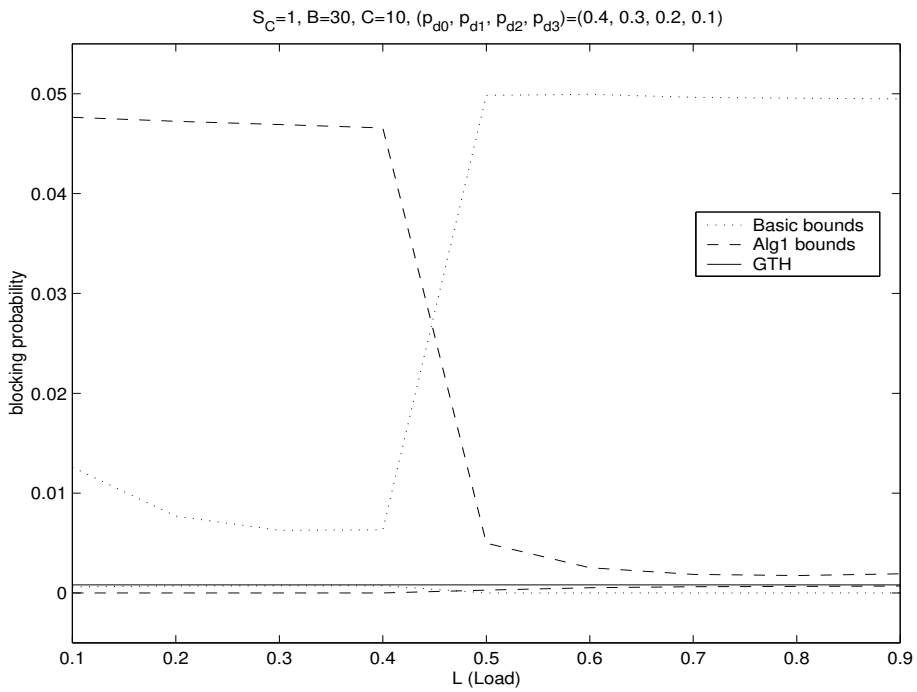

Fig. 1.(a) Blocking probabilities for $S_C = 1$ when $B = 30$ and $C = 10$.

degree of coupling 6×10^{-4}. The NCD partitionings considered for $S_C = 10$ in Figures 1.(c)–(d) all have 22 blocks with orders between 21 and 31, and degree of coupling values between 1×10^{-1} (for $L = 0.1$) and 2×10^{-2} (for $L = 0.9$). The NCD partitionings considered for $S_C = 100$ in Figures 1.(e)–(f) all have 22 blocks with orders between 21 and 31, and degree of coupling values between 2×10^{-2} (for $L = 0.1$) and 2×10^{-3} (for $L = 0.9$). The underlying MC that has 572 states and 20,198 nonzero elements takes 0.3 seconds to solve when $S_C = 1$ and 0.2 seconds to solve when $S_C \in \{10, 100\}$ using Algorithm 1. Steps 0 and 1.a take a total of about 0 seconds. It takes 2.6 seconds to solve the same MC by GTH for each L. It takes at least 1.5 seconds (5 iterations) to solve when $S_C = 1$ and at least 1.8 seconds (9 iterations) to solve when $S_C \in \{10, 100\}$ using IAD.

In Figure 2, using Algorithm 1 we present bounds on the blocking probability of voice calls and the dropping probability of data packets in the system with $B = 60$, $C = 30$. We take take $L \in \{0.1, 0.2, \ldots, 0.9\}$ and $S_C \in \{1, 10, 100\}$. The NCD partitionings considered for $S_C = 1$ in Figures 2.(a)–(b) all have 31 blocks with orders between 62 and 122, and a degree of coupling 5×10^{-3}. The NCD partitionings considered for $S_C = 10$ in Figures 2.(c)–(d) all have 62 blocks with orders between 31 and 61, and degree of coupling values between 2×10^{-1} (for $L = 0.1$) and 2×10^{-2} (for $L = 0.9$). The NCD partitioning considered for $S_C = 100$ in Figures 2.(e)–(f) all have 62 blocks with orders between 31 and 61, and degree of coupling values between 2×10^{-2} (for $L = 0.1$) and 7×10^{-3} (for $L = 0.9$). The underlying MC that has 2,852 states and 217,778 nonzero elements takes 3.3 seconds (Step 0: 0.3 seconds; Step 1.a: 0.3 seconds) to solve when $S_C = 1$ and 2.5 seconds (Step 0: 0.3 seconds; Step 1.a: 0.2 seconds) to solve when $S_C \in \{10, 100\}$ using Algorithm 1. It takes 260.0 seconds to solve the same MC by GTH for each L. It takes at least 64.2 seconds (3 iterations) to solve when $S_C = 1$ and at least 75.4 seconds (4 iterations) to solve when

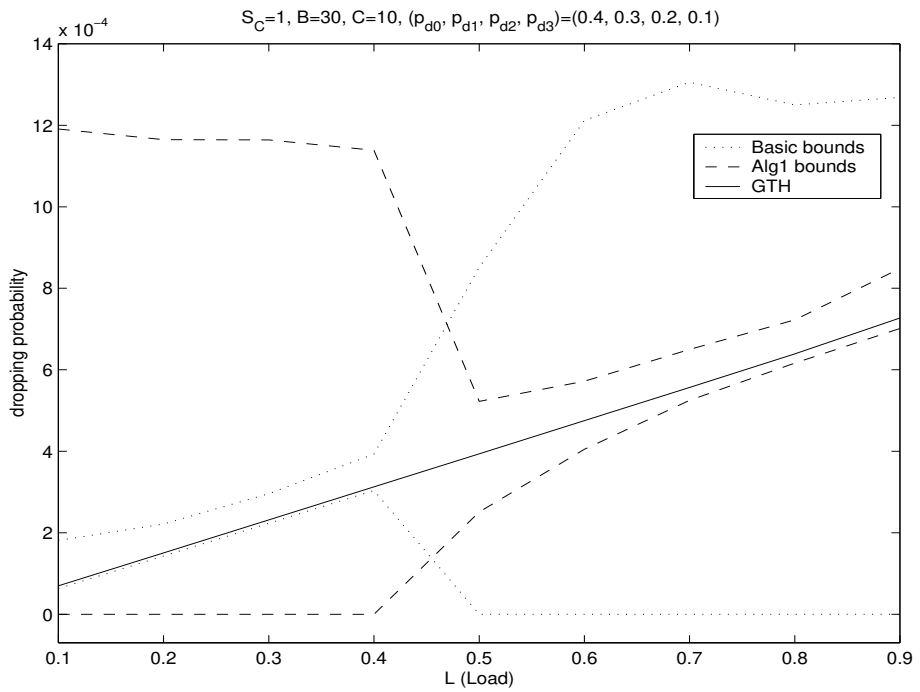

Fig. 1.(b) Dropping probabilities for $S_C = 1$ when $B = 30$ and $C = 10$.

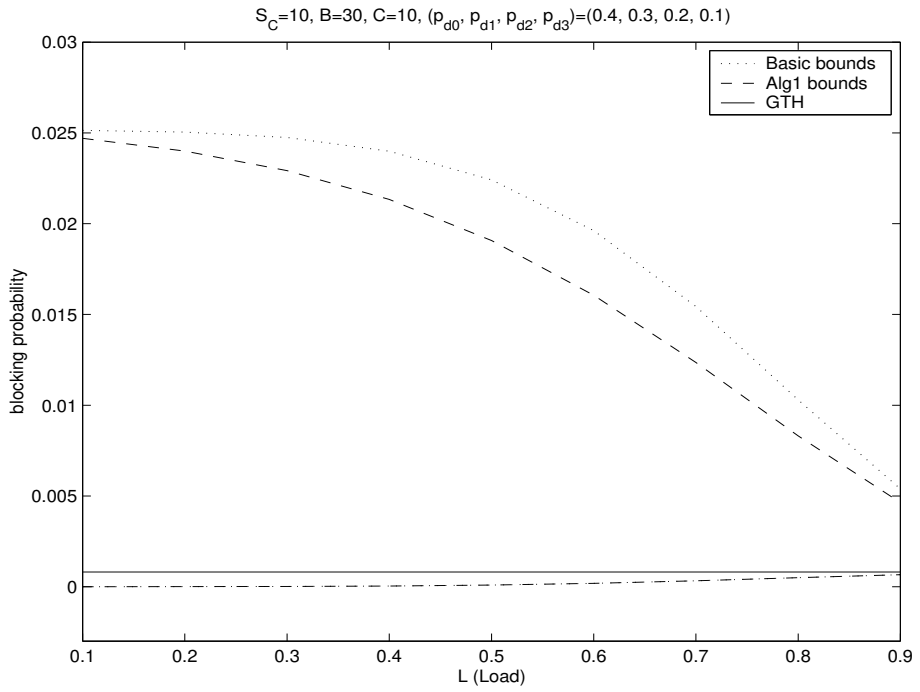

Fig. 1.(c) Blocking probabilities for $S_C = 10$ when $B = 30$ and $C = 10$.

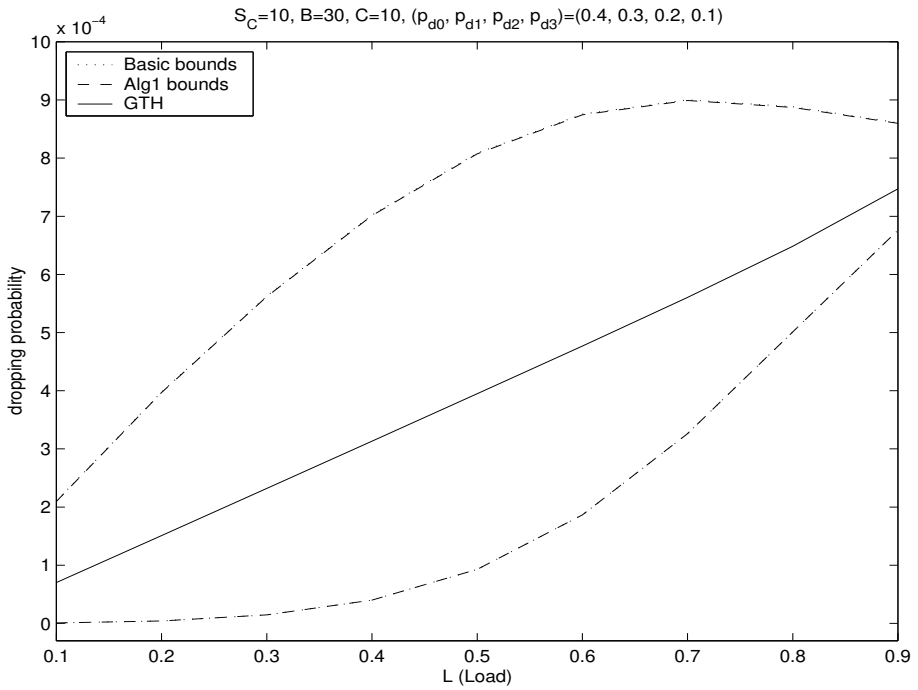

Fig. 1.(d) Dropping probabilities for $S_C = 10$ when $B = 30$ and $C = 10$.

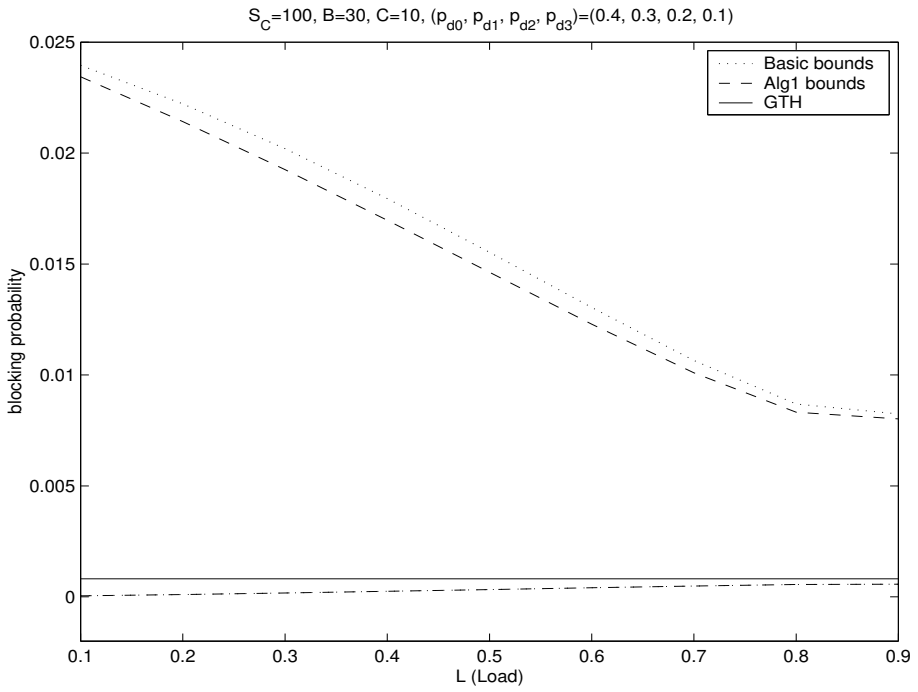

Fig. 1.(e) Blocking probabilities for $S_C = 100$ when $B = 30$ and $C = 10$.

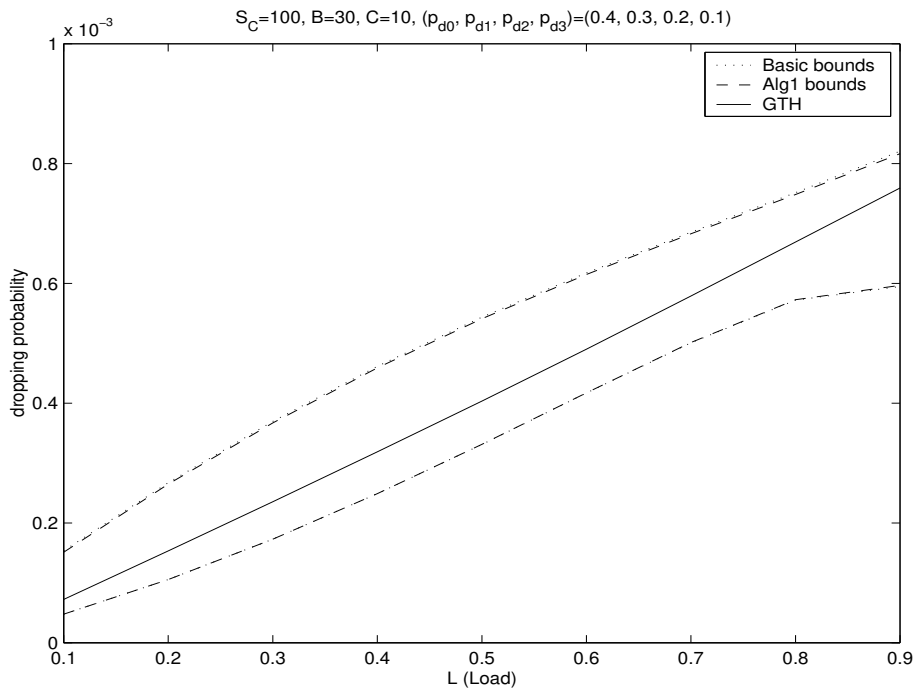

Fig. 1.(f) Dropping probabilities for $S_C = 100$ when $B = 30$ and $C = 10$.

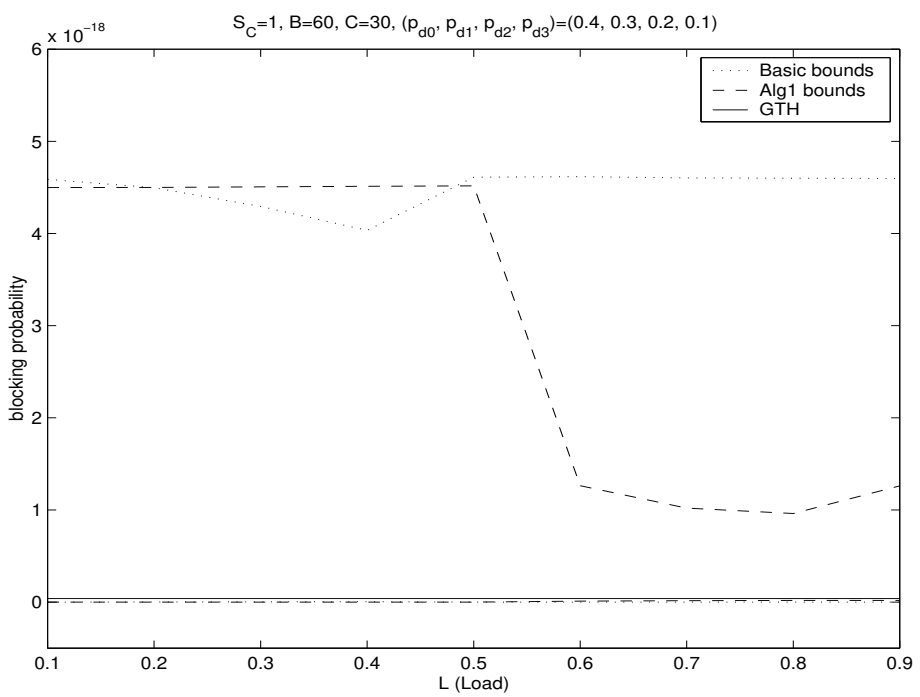

Fig. 2.(a) Blocking probabilities for $S_C = 1$ when $B = 60$ and $C = 30$.

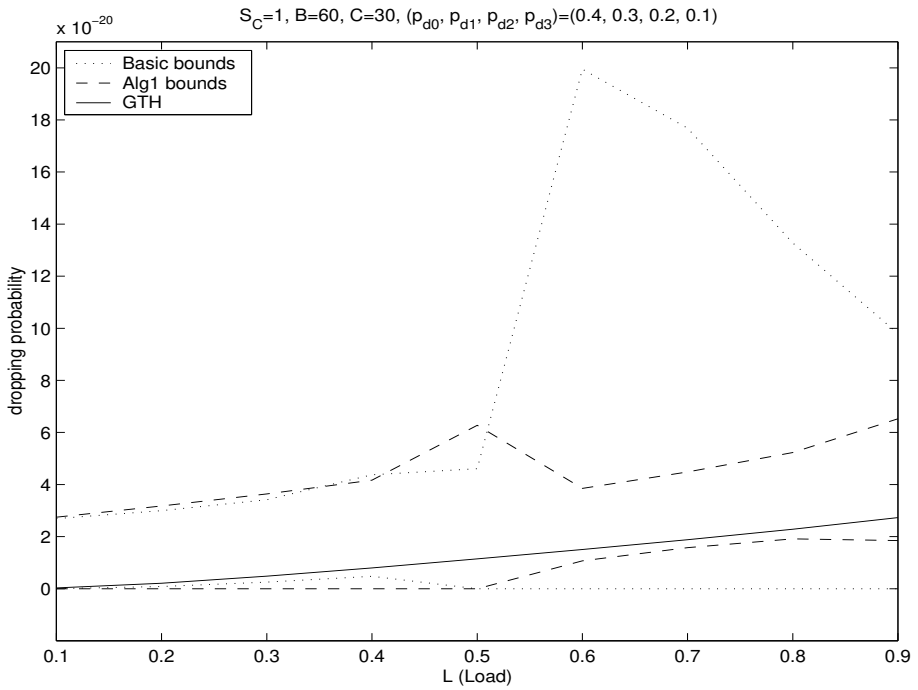

Fig. 2.(b) Dropping probabilities for $S_C = 1$ when $B = 60$ and $C = 30$.

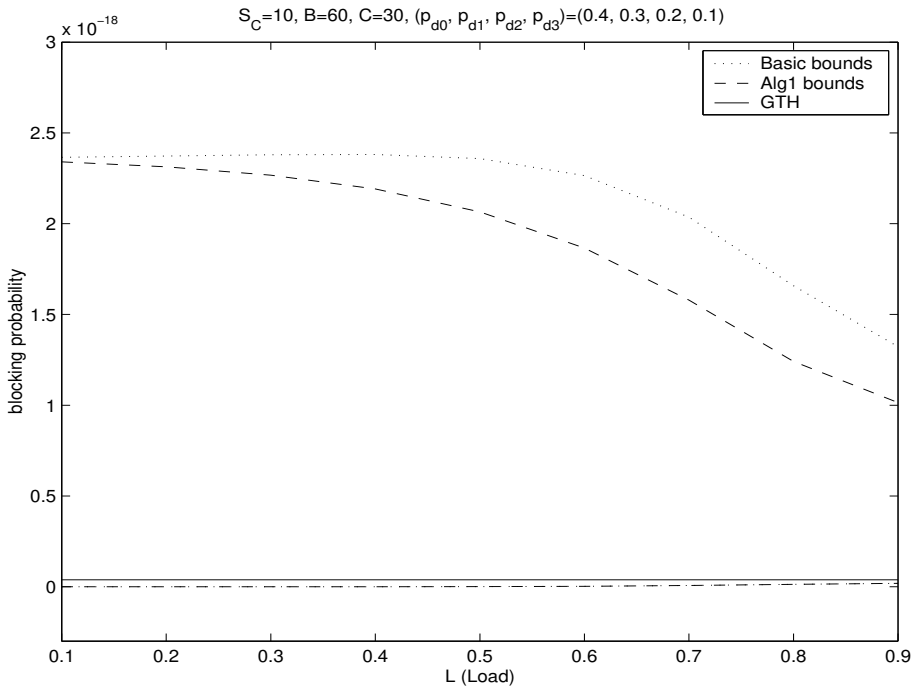

Fig. 2.(c) Blocking probabilities for $S_C = 10$ when $B = 60$ and $C = 30$.

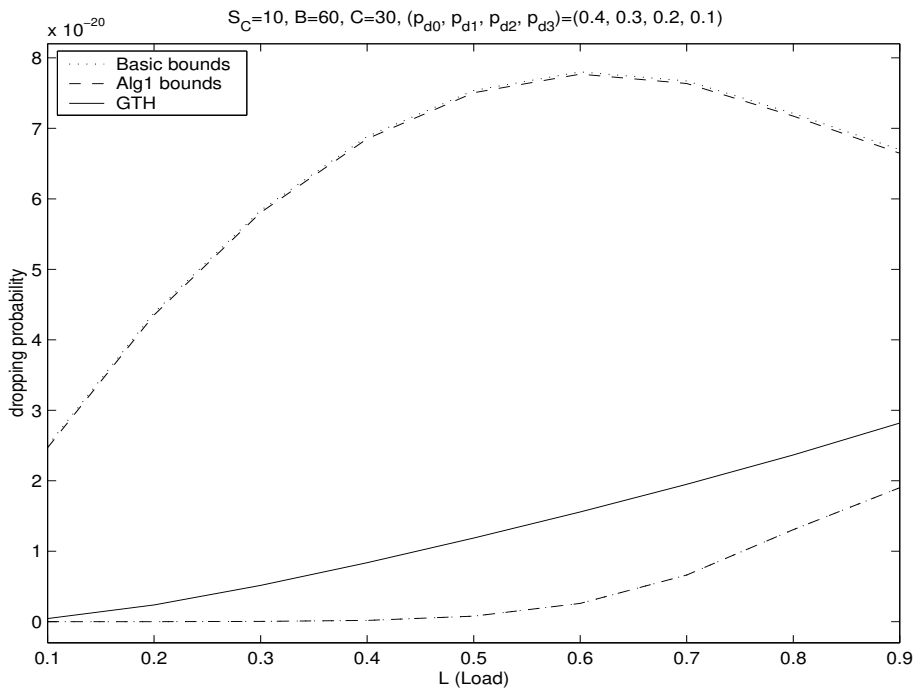

Fig. 2.(d) Dropping probabilities for $S_C = 10$ when $B = 60$ and $C = 30$.

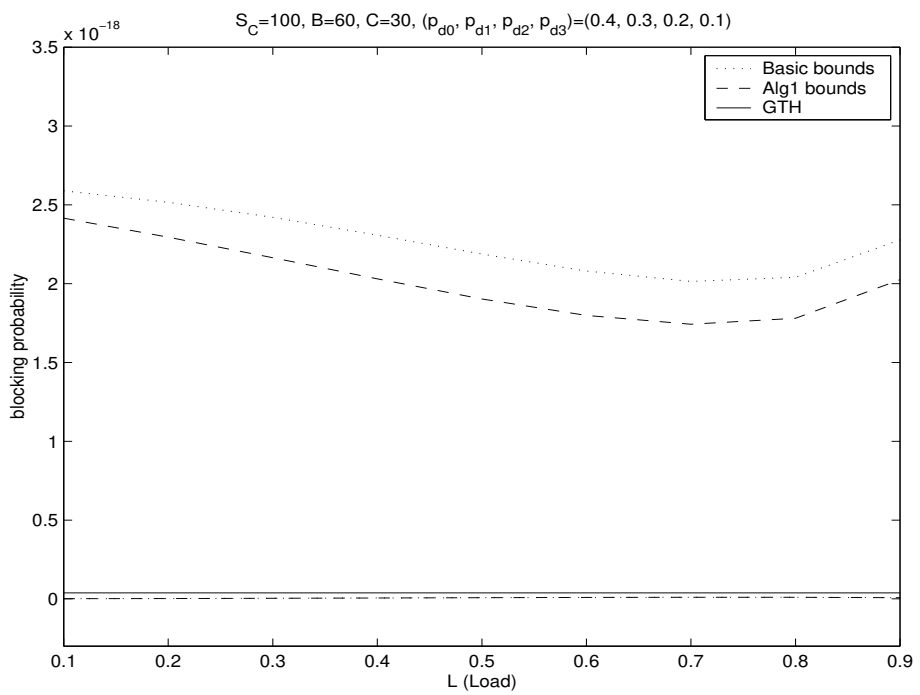

Fig. 2.(e) Blocking probabilities for $S_C = 100$ when $B = 60$ and $C = 30$.

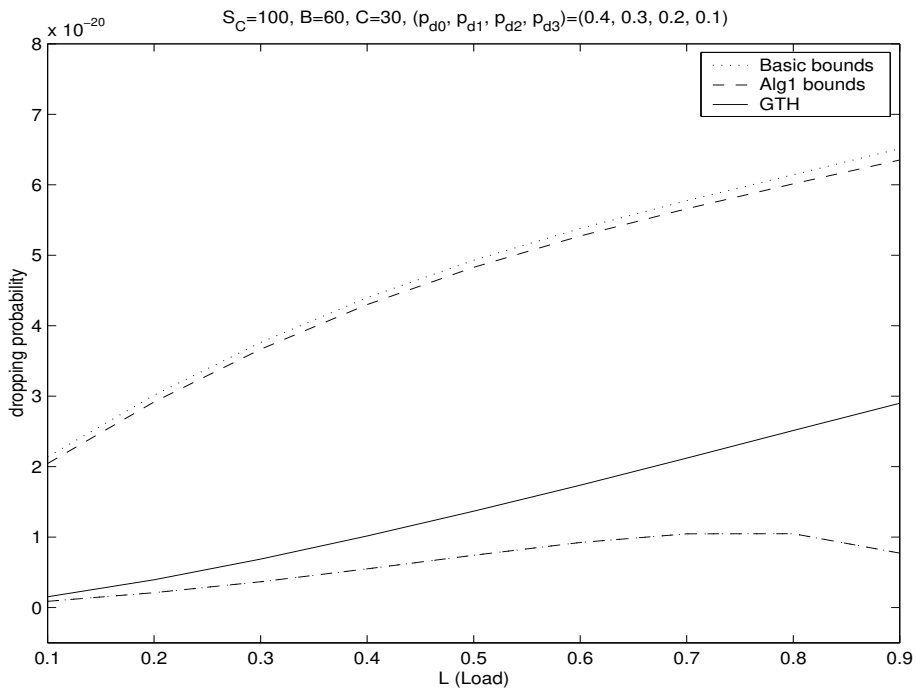

Fig. 2.(f) Dropping probabilities for $S_C = 100$ when $B = 60$ and $C = 30$.

$S_C \in \{10, 100\}$ using IAD.

Since voice calls have priority in service, their blocking probability is not affected by L and S_C (see Figures 1 and 2 parts (a),(c),(e)) whereas the dropping probability of data packets increases with L and S_C though the increase with S_C happens very slowly (see Figures 1 and 2 parts (b),(d),(f)). Both probabilities decrease when we move from Figure 1 to Figure 2. A bigger C implies a smaller blocking probability for voice calls, bigger B and C imply a smaller dropping probability for data packets.

The time spent to compute bounds using Algorithm 1 is very promising compared to solving the NCD MCs using GTH or IAD. This is understandable since Algorithm 1 solves multiple smaller systems (i.e., two systems corresponding to each NCD block i with order at most n_i) and two aggregated systems of order at most N whereas GTH solves the global system of order n and IAD performs a number of aggregation-disaggregation iterations. The bounds computed on p_{block} and p_{drop} using Algorithm 1 are highly acceptable; the bounds on p_{drop} are especially tight. Furthermore, the upper-bounds on p_{block} computed by Algorithm 1 are mostly better than those computed by the basic algorithm. Note that the $4(B+1)$ steady state probabilities used in computing p_{block} comprise those $3(C+1)$ used in computing p_{drop}. If we remove the unreachable states from the two formulae, there happens to be exactly $4(B+1)-2$ steady state probabilities that contribute to p_{block} and 6 that contribute to p_{drop}. We remark that st-comparison is expected to provide better componentwise bounds for states placed towards the end of the underlying MC. When we compare the values of $4(B+1)-2$ with N, it is clear that not all states contributing to p_{block} can be placed as such. This is an intuitive explanation for having tighter bounds on p_{drop} compared to those on p_{block}. We believe this to be also the reason behind obtaining better bounds on p_{block} with Algorithm 1 compared to the basic algorithm. Due to the reordering

of states in Step 1.a of Algorithm 1, we expect more improvement on performance measures computed using a large number of states than those that depend on a few states. There are other factors that influence the quality of the computed bounds such as the NCD partitioning employed, the ordering chosen by our heuristic within each NCD block, and the irreducibility structure of the computed st-monotone matrices.

Regarding the time it takes to compute bounds with Algorithm 1, the most important factor is the nonzero structure of the underlying MC. Consider, for instance, the application in [17] which also happens to be in the area of communications. It introduces an MMPP|E$_k$ |1| K continuous-time queueing model for a video source that is fed to an ATM multiplexer. When the number of stages in the Erlang (E) process approximating the deterministic service distribution is $k = 5$, the buffer size of the multiplexer is $K = 74$, the Markov Modulated Poisson Process (MMPP) modeling the arrival distribution has 8 states with a different arrival rate in each state, and the real parameters in the paper are used, we have a CTMC with $n = 8k(K+1) = 3,000$ states. The corresponding DTMC obtained through uniformization is highly NCD with a partitioning that has 8 blocks induced by the states of the MMPP. Each block is of order 375 and the degree of coupling is on the order of 10^{-5}. However, in each of its rows, this NCD MC has a maximum of 10 nonzero elements 7 of which are in the off-diagonal blocks. The diagonal blocks which are in the form of a quasi-birth-and-death (QBD) process have a maximum of 3 nonzero elements and lend themselves to relatively sparse factorizations. Hence, IAD is able to compute the steady state vector rapidly, and Algorithm 1, which favors relatively dense NCD MCs, cannot be recommended in this case.

6 Conclusion

In this paper, we have given the first numerical results on an application with (an improved version of) a componentwise bounding algorithm for the state probability vector of nearly completely decomposable Markov chains. The given two-level algorithm uses aggregation and stochastic comparison with the strong stochastic (st) order. In order to improve accuracy, it employs reordering of the states and a better componentwise probability bounding algorithm given st upper- and lower-bounding probability vectors. A thorough analysis of the algorithm from the point of view of irreducibility is provided. The run-time of the algorithm is much better than that of GTH and iterative aggregation-disaggregation in sparse storage, and the quality of the computed bounds on steady state probabilities are highly acceptable for the chosen application. It is difficult to make strong generalizations, but it is our experience that this algorithm will be useful in relatively dense nearly completely decomposable Markov chains with highly unbalanced steady state probabilities, a small number of states accumulating a large probability mass, and a small degree of coupling. Future work should focus on utilizing the algorithm in computing transient performance measures.

References

[1] O. Abu-Amsha, J.-M. Vincent, An algorithm to bound functionals of Markov chains with large state space, Rapport de recherche MAI n 25, IMAG, Grenoble, France, 1996.

[2] P.-J. Courtois, Decomposability: Queueing and Computer System Applications, Academic Press, New York, 1977.

[3] P.-J. Courtois, P. Semal, Bounds for the positive eigenvectors of nonnegative matrices and for their approximations by decomposition, Journal of the Assocation for Computer Machinery 31 (1984) 804–825.

[4] T. Dayar, Permuting Markov chains to nearly completely decomposable form, Technical Report BU-CEIS-9808, Department of Computer Engineering and Information Science, Bilkent University, Ankara, Turkey, 1998 (Downloadable from website ftp://ftp.cs.bilkent.edu.tr/pub/tech-reports/1998/).

[5] T. Dayar, N. Pekergin, Stochastic comparison, reorderings, and nearly completely decomposable Markov chains, in: B. Plateau, W. J. Stewart, M. Silva (Eds.), Numerical Solution of Markov Chains (NSMC'99), Prensas Universitarias de Zaragoza, Spain, 1999, pp. 228–246.

[6] T. Dayar, W.J. Stewart, Comparison of partitioning techniques for two-level iterative solvers on large, sparse Markov chains, SIAM Journal on Scientific Computing 21 (2000) 1691–1705.

[7] T. Dayar, W.J. Stewart, On the effects of using the Grassmann-Taksar-Heyman method in iterative aggregation-disaggregation, SIAM Journal on Scientific Computing 17 (1996) 287–303.

[8] W.K. Grassmann, M.I. Taksar, D.P. Heyman, Regenerative analysis and steady state distributions for Markov chains, Operations Research 33 (1985) 1107–1116.

[9] J. Kamburowski, Stochastically minimizing the makespan in two-machine flow shops without blocking, European Journal of Operational Research 112 (1999) 304–309.

[10] J. Keilson, A. Kester, Monotone matrices and monotone Markov processes, Stochastic Processes and Their Applications 5 (1977) 231–241.

[11] W.A. Massey, Stochastic orderings for Markov processes on partially ordered spaces, Mathematics of Operations Research 12 (1987) 350–367.

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

[13] N. Pekergin, Stochastic delay bounds on fair queueing algorithms, in: Proceedings of INFOCOM'99, New York, 1999, pp. 1212–1220.

[14] N. Pekergin, Stochastic performance bounds by state reduction, Performance Evaluation 36–37 (1999) 1–17.

[15] P. Semal, Analysis of large Markov models, bounding techniques and applications, Doctoral Thesis, Université Catholique de Louvain, Belgium, 1992.

[16] M. Shaked, J.G. Shantikumar, Stochastic Orders and Their Applications, Academic Press, California, 1994.

[17] P. Skelly, M. Schwartz, S. Dixit, A histogram-based model for video traffic behavior in an ATM multiplexer, IEEE/ACM Transactions on Networking 1 (1993), 446–459.

[18] W.J. Stewart, Introduction to the Numerical Solution of Markov Chains, Princeton University Press, New Jersey, 1994.

[19] D. Stoyan, Comparison Methods for Queues and Other Stochastic Models, John Wiley & Sons, Berlin, Germany, 1983.

[20] M. Tremolieres, J.-M. Vincent, B. Plateau, Determination of the optimal upper bound of a Markovian generator, Technical Report 106, LGI-IMAG, Grenoble, France, 1992.

[21] L. Truffet, Near complete decomposability: bounding the error by stochastic comparison method, Advances in Applied Probability 29 (1997) 830–855.

[22] V. Vèque, J. Ben-Othman, MRAP: A multiservices resource allocation policy for wireless ATM network, Computer Networks and ISDN Systems 29 (1998) 2187–2200.