Some Notes on Parallel Implementations of Solvers for Stochastic Matrices on Cluster of Workstations

Drkošová, Jitka
2001

Institute of Computer Science
Academy of Sciences of the Czech Republic

Some notes on parallel implementation of solvers for stochastic matrices on cluster of workstations

Jitka Drkošová, Petr Mayer, Arnošt Štědrý

Technical report No. 834

July 6. 2001

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: (+4202) 6605 1111, fax: (+4202) 858 57 89,
e-mail:ics@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

Some notes on parallel implementation of solvers for stochastic matrices on cluster of workstations

Jitka Drkošová, Petr Mayer, Arnošt Štědrý

Technical report No. 834

July 6. 2001

Abstract:

Parallel implementation of iterative agregation/disagregation method with various block smoothers is proposed. These methods are suitable for parallelization and can efficiently use the properties of stochastic matrices.

Keywords:
parallel, iterative methods, agregation/disagregation, stochastic matrices

1 Introduction

In past several decades much attention has been devoted to iterative methods for large sparse systems of linear equations. Krylov subspace methods become very popular, especially for solving the systems that arise from discretization of partial differential equations. The efficiency of these methods is dependent on system matrix properties. Although Krylov subspace methods have several limits when nonsymmetric systems are solved, fast convergence can be achieved if the system matrix is symmetric and positive definite. There are many libraries where Krylov subspace methods are implemented taking into account both the theoretical results and the efficiency of the implementation.

Considering this we decided to investigate another approach that is not based on Krylov subspace methods and a class of matrices that is more general than symmetric and positive definite. Szyld showed that Schwarz type methods work well for M-matrices; the behaviour is similar to that one that arises from elliptic equations. The choice of subspace plays a crucial role for domain decomposition methods. The subspace is determined by geometrical properties in the case of elliptic equations. We hope that the relevant information can be obtained from the Markov chains that describe the dependence among the elements.

Many real problems can be described by models based on Markov chains with a huge spaces of states that lead to the systems with large stochastic matrices. We would like to show that these systems can be solved quite efficiently when "domain decomposition techniques with core space" are implemented on a cluster of workstations.

In this report, based on known theoretical results, the parallel implementation of the iterative aggregation/dissagregation method with various smoothers is suggested. Formulating the algorithm, we emphasize on its computational efficiency in the parallel environment.

2 Markov Models

In this section some basic definitions and well-known relations concerning Markov chains and systems with stochastic matrices are briefly reviewed.

Definition 2.1 *A Markov process is a stochastic process which satisfies the memoryless condition known as Markov property*

$$Prob\left\{X\left(t\right) \leq x \mid X\left(t_0\right) = x_0, \ldots, X\left(t_n\right) = x_n\right\} = Prob\left\{X\left(t\right) \leq x \mid X\left(t_n\right) = x_n\right\}$$

for all integers n and for any sequence $t_0, t_1, \ldots, t_n$ such that $t_0 \leq t_1 \leq \ldots \leq t_n \leq t$.

If the departure of the state $X\left(t\right)$ is dependent on parameter t, the Markov process is referred to as *non homogeneous*. If not, it is referred to as *homogeneous*. If the state space of the Markov process is discrete it is called *Markov chain*. In this case a subset of natural numbers is chosen.

Two types of Markov chains may be identified: discrete time Markov chains (DTMC) and continuous time Markov chains (CTMC). In the case of CTMC the Markov property causes exponential distribution of the time which is spent in a fixed state. For DTMC the Markov property causes a geometrical distribution. In the following we shall focus on DTMC.

DTMC are established by a transient matrix $P\left(k\right) = \left(p\left(k\right)_{ij}\right) \in \Re^{N \times N}$ whose elements describe one step transition probabilities in step k. The element $p(k)_{ij}$ of the matrix is a probability of transition from one state to another in k-th time step

$$p\left(k\right)_{ij} = Prob\left\{X_{k+1} = j \mid X_k = i\right\}. \tag{1}$$

The probability that the process in n-th step goes from state i to state j is defined by matrix

$$P\left(m\right)^{(n)} = \left(p\left(m\right)_{ij}^{(n)}\right)$$

$$p\left(m\right)_{ij}^{(n)} = Prob\left\{X_{m+n} = j \mid X_m = i\right\}.$$

For $P(m)^{(n)}$ we can write

$$P(m)^{(n)} = P(m)\,P(m+1)\ldots P(m+n-1)$$

For any l holds the Chapman-Kolmogorov equation

$$P(m)^{(n)} = P(m)^{(l)}\,P(m+l)^{(n-l)}.$$

In case of homogenous DTMC, it means that one step transition matrices do not depend on the step number, i.e. $P(k) = P(l)$ for any k, l. It allows us to simplify formulas above in the following way

$$P^{(n)} = P^n$$

For a description of the process we can use nonnegative vector x whose elements x_i satisfy the condition $\sum_{i=1}^{N} x_i = 1$. Each element $x_i^{(k)}$ of vector describes probability for the Markov chain to be in state i in step k.The formula below shows how to calculate the probability vector $x^{(n+m)}$ in time $n+m$ from known vector $x^{(n)}$ in time n

$$x^{(n+m)^T} = x^{(n)^T} P^m.$$

This formula can be simplified and the relevant information is than obtained from the equation

$$x^T = x^T P, \qquad x^T e = 1, \tag{2}$$

where $e = (1, \ldots, 1)^T$.The solution of (2) can be interpreted as long term behaviour of the Markov chain usually called *stationary probability vector*. Note that the matrix P is *stochastic matrix* since $p_{ij} \geq 0$ for all i, j and $\sum_j p_{ij} = 1$ for all i.

Stochastic matrices are matrices with special properties. These can be expressed as a consequence of general Perron-Frobenius theorem, for proof see [4].

Theorem 2.1 *(Perron-Frobenius) Let P is a square nonnegative irreducible matrix of order $n \geq 1$. Then its spectral radius $\rho(P)$ is a positive simple eigenvalue of matrix P with positive eigenvector. No other eigenvalue of P has a nonnegative eigenvector.*

The following lemma is a straightforward consequence of Perron-Frobenius theorem for stochastic matrices.

Lemma 2.1 *Spectral radius of stochastic matrix is always 1.*

Let us express the equations (2) as follows

$$Ax = 0, \qquad x^T e = 1, \tag{3}$$

where $A = (I - P)^T$ and I is the identity matrix.

From Perron-Frobenius theorem we conclude that for an irreducible stochastic matrix the solution of (2) always exists and it is unique. The equivalence of equations (3) and (2) ensures that the solution of (3) also exists and (if P is irreducible) is unique. Since 1 is a simple eigenvalue of matrix P, matrix A is singular with rank $n-1$.

3 Numerical methods for stochastic matrices

In this section we overview methods that can be used for solving system (2) with stochastic irreducible matrix P as well as the methods that can be applied to the system (3) with a singular matrix A. Widely used methods as direct methods or Krylov subspace methods are only briefly mentioned with respect to their application for the system (2); point iterative methods for both (2) and (3) are considered as a special case of block iterative methods. The main attention is devoted to the aggregation/disaggregetion method and its part – block iterative methods. These algorithms are discussed in more details.

3.1 Direct methods

Direct methods, such as Gauss elimination and LU (LDU) decomposition can be applied to solve (3). Note that they are implemented in a special way since the coefficient matrix is singular and the system is homogeneous. If we consider irreducible Markov chain, then the coefficient matrix A is irreducible and it has one-dimensional null space. The implementation of direct methods that handles singularity makes use of this fact.

3.2 Krylov subspace methods

The class of Krylov subspace methods involves both the algorithms that generate the approximation to the eigenvalue problem as well as these ones that solve system of linear equations with nonsingular coefficient matrix. The use of Krylov subspace methods for the system (3) with singular matrix A is a slight modification of standard implementation, for details see[13].

3.3 Block iterative methods

In this section, block formulation of iterative methods such as power method, Jacobi method and Gauss-Seidel method are presented. They are simple modifications of point iterative methods. Block formulation allows us to make use of parallel implementation. First, we shall state the theorem that justifies the use of block iterative methods for the system (2), proof of the theorem can be found for example in [4].

Theorem 3.1 *Let P^T is a stochastic irreducible acyclic matrix of order n with $rank(P^T) = n - 1$. Then $\lim_{k\to\infty}(P^T)^k = ev^T$, where $e = (1, \ldots, 1)^T$, $v^T e = 1$ and $Pv = v$.*

Note that the theorem can be applied even when the matrix P is not acyclic. Denote $\tilde{P} = 1/2(I+P)$ and suppose that P is irreducible and $Pv = v$ for some nonnegative vector v. Then $\tilde{P}$ is irreducible and acyclic matrix and $\tilde{P}v = v$ is satisfied.

3.3.1 Block power method

Block power method is based on (point) power method that is well known as a method for determining the eigenvector corresponding to a dominant eigenvalue of a matrix.

In order to describe the block power method let us denote a partitioning of the matrix P as follows

$$\begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1n} \\ P_{21} & P_{22} & \cdots & P_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ P_{n1} & P_{n2} & \cdots & P_{nn} \end{pmatrix},$$

where n is a number of block rows/columns. The vector $x = (X_1, \ldots, X_n)$ is partitioned in the same way. We consider only such splitting which ensures that the diagonal blocks are square matrices. The algorithm that describes the block power method is very simple and can be expressed as follows.

Algorithm 1 Block power method

X^0 arbitrary such that $X^0 \geq 0$ and $\sum_i X_i^0 = 1$
for $k = 1, 2, \ldots$ until X^k is accurate enough **do**
  for $i = 1, \ldots, n$ **do**
    $X_i^k = \sum_{j=1}^{N} P_{ij} X_j^{k-1}$
  end for
end for

The convergence of block power method is satisfied due to theorem 3.1.

3.3.2 Block Jacobi method

Jacobi method generates the approximation to the solution of the nonhomegeneous system with non-singular coefficient matrix. Jacobi method converges if the spectral radius of the iteration matrix is less than one. Let us express the matrix A from (3) as $A = D - (L+U)$, where the matrices D, L and U represent diagonal, strictly lower-triangular and strictly upper-triangular parts of A, respectively. The iterative matrix for Jacobi method is $D^{-1}(L+U)$. This matrix is nonnegative and note that the solution of the homogeneous system of linear equations (3) is equivalent to the solution of the eigenvalue equation $D^{-1}(L+U)x = x$. The assumptions of Perron-Frobenius theorem are fulfilled and the existence and uniqueness of the solution are assured. If the matrix A is acyclic then the spectral radius $\rho(D^{-1}(L+U) < 1$ and the Jacobi method with $X^0 \geq 0$ converges. Compare with theorem 3.1.

Block Jacobi method is a generalization of point Jacobi method. Let us consider block partitioning of matrix A such that diagonal blocks are square matrices whose dimension can vary. Vector X represents the corresponding partitioning of the vector x. Using this notation and supposing that the matrix A consists of $n \times n$ blocks, the system (3) can be rewritten using the block formulation as follows.

$$\begin{pmatrix} D_{11} & U_{12} & \cdots & U_{1n} \\ L_{21} & D_{22} & \cdots & U_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ L_{n1} & L_{n2} & \cdots & D_{nn} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix} = 0$$

Based on this notation the pseudocode of block Jacobi algorithm is described by algorithm 2.

Algorithm 2 Block Jacobi algorithm

for $i = 1, \ldots, n$ **do**
  $D_{ii} = l_i u_i$
  l_i is lower triangular matrix with ones on its diagonal
  u_i is upper triangular matrix.
end for
X^0 arbitrary such that $X^0 \geq 0$ and $\sum_i X_i^0 = 1$
for $k = 1, 2, \ldots$ until X^k is accurate enough **do**
  for $i = 1, \ldots, n$ **do**
    $Y_i^k = -(\sum_{j=1}^{i-1} L_{ij} X_j^{k-1} + \sum_{j=i}^{n} U_{ij} X_j^{k-1})$
    $X_i^k = (l_i u_i)^{-1} Y_i^k$ is solved using back substitution
  end for
end for

3.3.3 Block Gauss-Seidel method

Gauss-Seidel method could be seen as a slight modification of Jacobi method. If we consider the same splitting of the matrix A as for Jacobi method, e.g. $A = D - (L + U)$ then the iterative matrix is $(D - L)^{-1}U$ and this matrix is nonnegative. The equivalence of Gauss-Seidel method with power method can be shown in the same way as for Jacobi method. Its convergence is also assured according theorem 3.1 supposing that A is an acyclic matrix.

Using the notation introduced in the previous subsection we have algorithm 3 for block Gauss-Seidel algorithm.

Remark

1. Convergence for block iterative methods is a consequence of the convergence of the point iterative methods. The matrix properties that ensure the convergence are preserved when block formulation is used.

Algorithm 3 Block Gauss-Seidel algorithm

for $i = 1, \ldots, n$ **do**
  $D_{ii} = l_i u_i$
  l_i is lower triangular matrix with ones on its diagonal
  u_i is upper triangular matrix.
end for
X^0 arbitrary such that $X^0 \geq 0$ and $\sum_i X_i^0 = 1$
for $k = 1, 2, \ldots$ until X^k is accurate enough **do**
  for $i = 1, \ldots, n$ **do**
    $Y_i^k = -(\sum_{j=1}^{i-1} L_{ij} X_j^k + \sum_{j=i}^{n} U_{ij} X_j^{k-1})$
    $X_i^k = (l_i u_i)^{-1} Y_i^k$ is solved using back substitution
  end for
end for

2. Note that if the Jacobi/Gauss-Seidel matrix is a cyclic matrix than convergence of methods is not assured since the spectral radius is one.

3.4 Aggregation/disaggregation algorithm

Now we address some effective method for computing stationary probability vectors. We analyze aggregation/disaggregation methods, which show very good behaviour, especially in the case of large Markov chains. It is well known (see [9],[13]) that such type of method is extremely good in the case of irreducible, but nearly completely decomposable matrices. But even if matrix is only irreducible, the method is very useful.

The aggregation/disaggregation algorithm is expressed by following few steps.
We shall define mapping $g : 1, \ldots, N \longmapsto 1, \ldots, n$, which maps the indices to aggregation groups. Using g we define communication operators: the restriction R and prolongation operator $S(x)$.

$$(Rx)_i = \sum_{g(j)=i} x_j, \qquad x \in \Re^N$$

$$(S(x)z)_i = z_{g(i)} \frac{x_i}{(Rx)_{g(i)}} \qquad for \qquad x > 0.$$

For nonzero elements of R and $S\,(x)$ we can write

$$R_{g(i)i} = 1 \tag{4}$$

$$S\,(x)_{ig(i)} = \frac{x_i}{(Rx)_{g(i)}} \tag{5}$$

The fact that steps 2 and 3 of the algorithm 4 are sensible is formulated in the following lemma that is proved in [9].

Lemma 3.1 *If P is an irreducible stochastic matrix and x is a positive vector than $B(x)$ defined in step 2 of the algorithm 4 is also irreducible stochastic matrix.*

One can think about SPV algorithm 4 as a kind of two grids method. The steps 2,3,4 represent coarse grid correction, step 5 is smoothing, with operator T as a smoother.

Theorem below, which is proved in [9], states the local convergence of algorithm 4.

Theorem 3.2 *Let P is an irreducible stochastic matrix, let g be a mapping which defines aggregation, let M, K be a splitting which defines both smoother T and matrix Y. Then algorithm 4 is locally convergent for any parameters s, t .*

Algorithm 4 SPV(P,T,Y)

Require: Let P be irreducible stochastic matrix, g define aggregation, $\widetilde{x_1} > 0$ is initial vector.

Require: Let $\widehat{P} = \left(I + P^T\right)/2$, $A = I - \widehat{P} = M - K$. Define $T = \left(M^{-1}K\right)^t$ and $Y = \left(\widehat{P}\right)^s$ for $t \geq 1$ and $s \geq 1$, $x_1 = T\widetilde{x_1}$. Let $\varepsilon > 0$ is final tolerance.

Step 1 Set $k = 1$

Step 2 Construct

$$B(x_k) = RYS(x_k) \tag{6}$$

Step 3 Solve $B\left(x_k\right) z_k = z_k, \qquad e^T z_k = 1$

Step 4 Set $v_{k+1} = S\left(x_k\right) z_k$

Step 5 Compute and normalize $x_{k+1} = Tv_{k+1}$

if $\|x_k - x_{k+1}\| \geq \varepsilon$ **then**

$k = k + 1$, go to Step 2

else

Stop

end if

We will be interested mainly in three types of smoother.

1. simple power method $M = I \qquad K = \widehat{P}$
2. Block Jacobi method – known as Vantilborg method $M_{ij} = A_{ij}$ when $g\left(i\right) = g\left(j\right)$
3. Block Gauss-Seidel mehod – known as Koury-McAllister-Stewart method $M_{ij} = A_{ij}$ when $g\left(i\right) \geq g\left(j\right)$.

First of them is, of course, the slowest. But in many situations one may not be able to get matrix P elementwise. For example, when we analyze Stochastic Automata Networks, P is formed as a sum of tensor products. In such situations, the first smoother can be the only one, which is computable.

4 Parallel implementation

The formulation of aggregation/disaggregation algorithm with any type of the block smoothers allows its parallel implementation. It turns out that the efficiency of the computation is dependent on the implementation of smoothers since the main amount of work is done in them. Load balancing can be treated by many ways. One possible way is to create enough groups and then to distribute average amount of work to each processor. Using this approach, one should face the problem, that aggregation groups can be dictated by the problem and the convergence rate can be decreased by artificial changing of them. Nevertheless some strategies should be found.

4.1 Parallel performance of the aggregation/disaggregation algorithm

Our purpose is to implement the algorithm 4 with smoothers described by algorithms 1, 2, 3 in a parallel environment. Roughly speaking, the agreggation/disaggregation algorithm consists of three main parts. First of them can be described as a construction of the matrix $B(x_k)$ and it corresponds with step 2 of the algorithm 4. In the second part, a normalized solution of the homogeneous system with the coefficient matrix $B(x_k)$ is searched and the transformed vector is computed using prolongation operator (step 3,4) . In the third part (step 5), a particular smoother T – generated by any of block iterative algorithms – is used to generate new approximation x_{k+1}.

Our implementation is based on the block partitioning of the system coefficient matrix A according to the aggregation groups. Namely, if $g(i) = g(i+1) = ... = g(i+k) = j$ for $i \in \{1, \ldots, N\}$, $j \in \{1, \ldots, n\}$ then the j-th block row/column consists of $i, \ldots, i+k$ (point) rows/columns of matrix A. Block rows of the matrix are distributed among processors. From the point of view of parallelization, the first part of the algorithm 4 seems to be a bottleneck since a communication among processors

is necessary for computation of matrix $B(x_k)$. The second part of the algorithm is accomplished using only one processor. Note that the matrix $B(x_k)$ is typically of a "small" dimension under the condition that $n << N$. The third part of the algorithm – that can be itself an iteration process – includes the biggest amount of arithmetic operations. On the other hand, block approach allows parallel implementation of step 5 of the algorithm 4.

4.2 Tools

The first part and the third one of the algorithm 4 are treated using parallel tools. Let us suppose that the matrix is distributed among m processors. In order to compute the matrix $B(x_k)$ data from $m-1$ processors are gathered on one processor. Standard MPI functions that provide communication among processors and exchanging data are used. The coarse operation of the third part of the algorithm 4 is a sparse matrix vector multiplication in he parallel environment. Functions from Aztec library are called in this step. Aztec library is also used when block iterative methods such as power, Jacobi and Gauss-Seidel are implemented in parallel way.

4.3 Data structure – DVBR format

The matrices are handled using standard efficient sparse schemes. We need such a sparse scheme that is suitable for parallel computations and it is both flexible and standard. Considering our purposes we finally decided for *distributed varying block row* (DVBR) format, that is the generalization of the VBR format, for details see [14, 1] . This format is based on a block partitioning of the sparse matrix. The blocks can be easily distributed over the nodes of cluster as it is shown on figure 1.

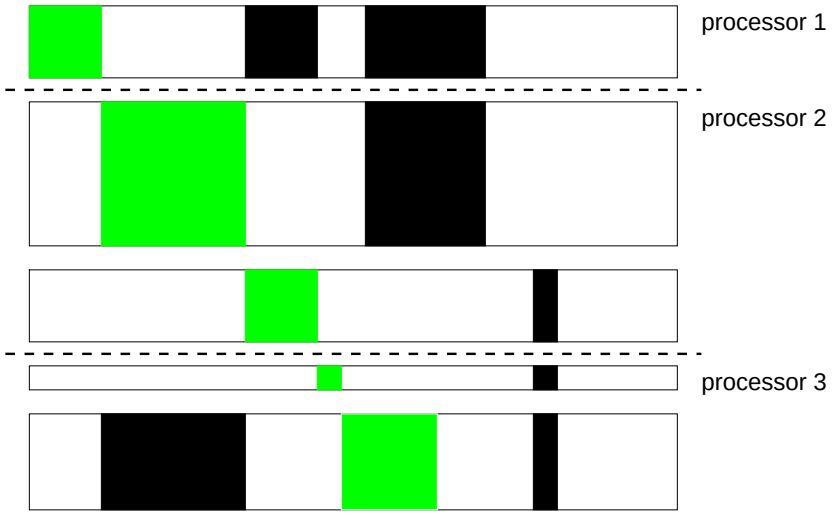

Figure 1: Schema of block structure of matrix. Diagonal blocks are grey, the others are black.

References

[1] S. Carney, M. Heroux, and G. Li. A proposal for a sparse blas toolkit. Technical report, Cray Research Inc Eagen, MN, 1993.

[2] G. Ciardo, Blakmore. A., P.F. jr. Chimento, J. K. Muppala, and K. S. Trivedi. Automated generation and analysis of markov reward models using stochastic reward nets. In *Linear Algebra, Markov Chain, and Queueing Models, Ed. By C. D. Meyer, R. J. Plemmons*, pages 145–191. Springer-Verlag, 1993.

[3] Ren David and Hassane Alla. *Petri nets and Grafcet: tools for modelling discrete event systems.* Prentice Hall International, 1992.

[4] M. Fiedler. *Speciální.*

[5] Barry W. Johnson. *Design and Analysis of Fault-Tolerant Digital Systems.* Addison-Wesley Publishing Company, Massachusetts, 1989.

[6] Š. Klapka and P. Mayer. Some aspects of modelling railway safety. In *Proceedings of the XIIIth SANM, Nečtiny*, pages 135–140, 1999.

[7] K. Kule. *Spolehlivost a bezpečnost zabezpečovacích zařízení.* NADAS, Praha (in Czech), 1980.

[8] I. Marek and P. Mayer. Iterative aggregation/disaggregation methods for computing stationary probability vectors of stochastic matrices can by finitely terminating.

[9] I. Marek and P. Mayer. Convergence analysis of an iterative aggregation/disaggregation method for computing stationary probability vectors of stochastic matrices. *Numer. Linear Algebra with Applications*, 1998.

[10] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets.* John Wiley and Sons, 1995.

[11] B. Plateau and K. Atif. Stochastic automata network for modelling parallel systems. *IEEE transaction on software engineering*, 1991.

[12] K. Rástočný. *Modely pre analýzu bezpečnosti počítačových zabezpečovacích systémov.* PhD thesis, Žilina, 1998.

[13] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains.* Princenton University Press, Princenton, New Jersey, 1994.

[14] R. S. Tuminaro, M. Heroux, S. A Hutchinson, and J. N. Shadid. Official aztec user's guide, version 2.1. Technical report, Massively Parallel Computing Research Laboratory, Sandia National Laboratories, Albuquerque, NM 87185, 1999.

[15] J. Walter. *Stochastické modely v ekonomii - in Czech.* SNTL, Praha, 1970.