This report is a revision of report 343, November 1984. To appear in ACM Trans. Mathematical Software

Solving Narrow Banded Systems on Ensemble Architectures

Lennart Johnsson

Research Report YALEU/DCS/RR-418
August 1985

This work was supported by the Office of Naval Reseach under Contract No. N00014-84-K-0043

Table of Contents

Solving Narrow Banded Systems on Ensemble Architectures

S. Lennart Johnsson
Department of Computer Science and
Electrical Engineering
Yale University

August 1985

Abstract

We present concurrent algorithms for the solution of narrow banded systems on ensemble architectures, and analyze the communication and arithmetic complexities of the algorithms. The algorithms consist of three phases. In phase 1 a block tridiagonal system of reduced size is produced through largely local operations. Diagonal dominance is preserved. If the original system is positive, definite and symmetric, so is the reduced system. It is solved in a second phase, and the remaining variables are obtained through local backsubstitution in a third phase. With sufficiently many processing elements, there is no first and third phase. We investigate the arithmetic and communication complexity of Gaussian elimination and block cyclic reduction for the solution of the reduced system on boolean cubes, perfect shuffle and shuffle-exchange networks, binary trees and linear arrays.

With an optimum number of processors the minimum solution time on a linear array is of an order that ranges from $O(m^2\sqrt{Nm})$ to $O(m^3 + m^3 log_2(N/m))$ depending on the bandwidth, the dimension of the problem, and the times for communication and arithmetic. For boolean cubes, cube-connected cycles, perfect shuffle and shuffle-exchange networks, and binary trees the minimum time is $O(m^3 + m^3 log_2(N/m))$ including the communication complexity.

1. Introduction.

The architectural model used for the complexity analysis is one in which there are no shared resources. Each processor has its own storage and executes its own instruction stream. Hence, the architecture is of the MIMD type in Flynn's classification, [2]. We refer to such architectures as *ensemble architectures*, a term introduced many years ago (e.g., in the PEPE, Parallel Element Processing Ensemble). The processor interconnections considered here are linear arrays, boolean cubes, perfect shuffle and shuffle-exchange networks, and binary trees. These configurations offer different trade-offs between communication capabilities and interconnection cost. For computations distributed throughout the ensemble, and requiring global communication, such as the solution of irreducible systems of linear equations, the complexity of computation contains a term proportional to the diameter of the ensemble, at best [3]. Contention for communication resources or sequential dependencies in the algorithm can considerably increase this lower bound on the communication complexity. The lowest communication complexity of the algorithms presented in this paper is proportional to the diameter of the ensemble.

The banded irreducible system $AX = Y$ is assumed to be of order N and bandwidth $2m + 1$, where $a_{ij} = 0$ for $\mid i - j \mid < m$. The number of processing elements relative to the bandwidth of the matrix and the interconnection of the processing elements are important factors in choosing an algorithm for the solution of a banded system of equations. Here we present and analyze the

arithmetic and communication complexities of some algorithms for systems of a narrow bandwidth, $m/N \ll 1$. No advantage is taken of any particular structure within the band. Algorithms for the solution of tridiagonal systems of equations on a variety of ensemble architectures are described and analyzed in [8]. In [10] we present and analyze algorithms for banded systems with a large bandwidth.

Parallel algorithms for the solution of a system of linear eqautions are often devised through partitioning of the set of equations into subsets, substructures, for which computations to a certain extent can be performed independently and concurrently. The substructuring technique employed for the algorithms presented here is inspired by that of [14], [11] for banded systems, and [15] for tridiagonal systems. It is closely related to nested dissection [5], [4], [12], which would yield P substructures not being bisectors for $P-1$ bisectors. An algorithm based on substructuring techniques for banded systems have also been proposed by Reiter and Rodrigue [13] who analyze some of the numerical properties of their algorithm.

Our algorithm proceeds in 3 phases (as do the algorithms of Sameh et. al., and Wang). In phase 1 computations are performed within the substructures and with limited communication between adjacent substructures so that the banded system is transformed into a form from which a block tridiagonal system of order P and block size m by m can be separated out and solved in phase 2. If nested dissection is applied to the graph corresponding to a banded matrix with dense band of width $2m+1$, then the reduced system would correspond to the $P-1$ bisectors, each of which is of size m (and the reduced system be of order $(P-1)m$). In the graph corresponding to the N by N matrix A, there are N vertices labeled with a distinct index $i = \{1, 2, ..., N\}$, and a directed edge from node i to node j if $a_{ij} \neq 0$. In phase 3 the variables internal to the substructures are computed through backsubstitution. Lawrie and Sameh [11] transform the system of equations in their first phase such that a block pentadiagonal system of order $2(P-1)$ with blocks of order m by m can be separated out and solved in a second phase. The number of arithmetic operations required to deduce the block tridiagonal system is the same as the number of operations required to deduce the pentadiagonal system. If the original matrix is symmetric and positive definite, then so is the block tridiagonal system. This is not necessarily true for the pentadiagonal system of Lawrie and Sameh's algorithm. The pentadiagonal system can be permuted into a block tridiagonal system by pairwise column permutations, but this system is no longer diagonally dominant.

Lawrie and Sameh investigate the solution of the pentadiagonal system by Gaussian elimination on a linear array. The solution of the block tridiagonal system by 2-way Gaussian elimination yields an arithmetic complexity with the highest order terms equal to $(7/3m^3 + 3m^2)P$ for phase 2 of our algorithm compared to $(9m^3 + 12m^2)P$ for Lawrie and Sameh's algorithm, a reduction by approximately a factor of 4. If the banded system is symmetric and positive definite then an additional reduction in the arithmetic complexity of our method is possible. In [1] the solution of the reduced system by block-Jacobi iterations, and by the preconditioned conjugate gradient method on a linear array is studied. We show that block cyclic reduction (BCR) is of a lower total complexity than 2-way Gaussian elimination (2GE) on a linear array, if $\alpha = t_c/t_a \leq 28m/9$ for one right hand side, and $\alpha \leq 4NR$ if $NR \gg m$, where NR is the number of right hand sides. The time for communicating one floating-point number is denoted t_c and the time for an arithmetic operation t_a. Some detailed optimizations of the algorithm, that are not carried out here, change

the constants in these relations only to a minor extent. Depending on the relative values of m, N, t_a and t_c, the minimum complexity of BCR is $O(m^2\sqrt{N} + m^3 log_2 N)$ or $O(m^3 + m^3 log_2(N/m))$.

Block cyclic reduction is particularly well suited for ensemble architectures configured as boolean cubes, cube-connected cycles, perfect shuffle and shuffle-exchange networks, and binary trees. With one processor per block row and $O(N/m)$ processing elements the banded system can be solved with a computational complexity of $O(m^3 + m^3 log_2(N/m))$ on those ensemble configurations compared to a minimum complexity of $O(m^2\sqrt{Nm})$ for Gaussian elimination on a linear array of $O(\sqrt{N/m})$ processing elements.

For the algorithms analysed here N/P consecutive equations are allocated to a processor that computes N/P unknowns. P is constrained by the relation $P \leq \lceil N/m \rceil$, in order that the partitioned system be block tridiagonal. If substructuring through nested dissection is made, then each substructure will have $(N+m)/P - m$ equations for local variables, and with an entire bisector assigned to a substructure not being a bisector all but 1 of those will have an additional m equations and variables. Since the bisectors for a matrix with dense band of width $2m+1$ is of size m it follows that the number of bisectors **and** other substructures must be less than N/m.

The number of processors can be larger than P. Multiple processors can be used per partition. With clusters of m^2 suitably configured processors assigned to each partition, a solution can be obtained in a time of order $O(m + m log_2(N/m))$ [10], including the time for communication. Those algorithms combine the algorithms presented here with an algorithm obtained from a transformation of the concurrent Gaussian elimination algorithm in [7]. In a boolean cube lower dimensional subcubes can be considered as clusters.

In section 2 we define the submatrices that follow from the partitioning of the banded system and introduce the notation used in describing the algorithm and performing the analysis. In section 3 the computations and communication of phases 1 and 3 are defined and analysed. An algorithm implementing a version of Gauss-Jordan elimination for the substructures is described first. Then a parallel Cholesky factorization algorithm is given for symmetric, positive definite matrices. Section 4 deals with the solution of the reduced system, phase 2, by Gaussian elimination and block cyclic reduction on a variety of ensemble architectures. The complexity results for the solution of the banded system are summarized in section 5. Section 6 contains the conclusions of the analysis.

2. Preliminaries.

The banded system $AX = Y$ is partitioned into P partitions, with each partition consisting of consecutively indexed equations. The partitioned matrix is block tridiagonal for $N/P \geq m$. For $m \ll N/P$ the off diagonal blocks contain a large fraction of zeroes. Advantage can be taken of this property, and a reduced system derived with dense matrices of order m by m. For $q = N/P$ not an integer, $N - P\lceil N/P \rceil$ partitions are assigned $\lfloor N/P \rfloor$ equations (and the remaining partitions $\lceil N/P \rceil$ equations). We ignore this possible difference in partition size in the description and analysis of the algorithms. The reduced system is always of size mP by mP for the substructuring employed here ($(P-1)m$ by $(P-1)m$ for nested dissection). The partitioned banded system has the following block form:

$$\begin{pmatrix} A_1 & B_1 & & & & & & X_1 & & Y_1 \\ C_2 & A_2 & B_2 & & & & & X_1 & & Y_1 \\ & C_3 & A_3 & B_3 & & & & X_3 & & Y_3 \\ & & \ddots & \ddots & \ddots & & & \vdots & & \vdots \\ & & & C_i & A_i & B_i & & X_i & & Y_i \\ & & & & \ddots & \ddots & \ddots & \vdots & = & \vdots \\ & & & & & C_N & A_N & X_N & & Y_N \end{pmatrix}$$

The blocks have an additional structure as illustrated in Figure 1 for $P \ll N/m$ and $P \approx N/m$.

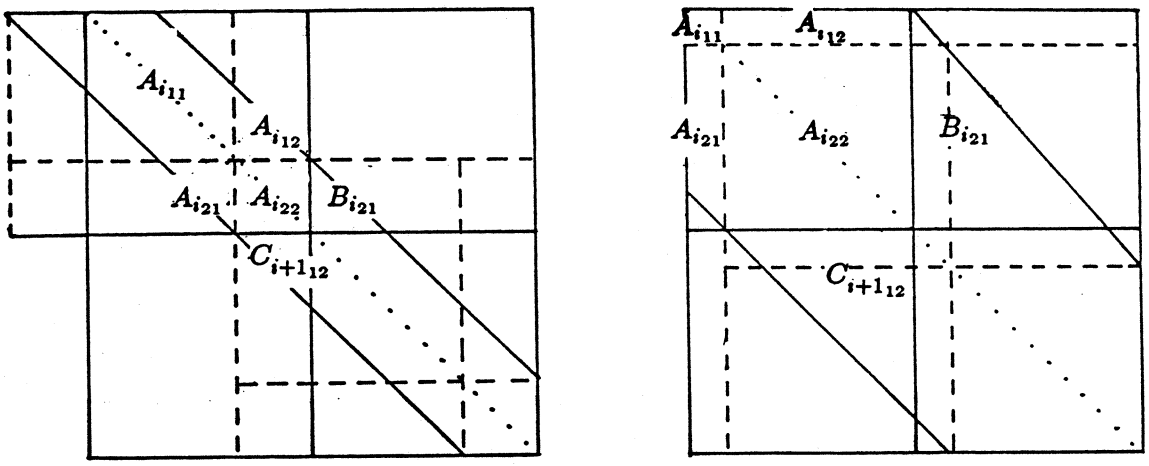

Figure 1: Partitioning of a banded matrix.

In the following we use the notation:

$$A_i = \begin{pmatrix} E_{i_1} & E_{i_2} \\ E_{i_3} & E_{i_4} \end{pmatrix} \quad B_i = \begin{pmatrix} 0 & 0 \\ F_{i_1} & F_{i_2} \end{pmatrix} \quad C_i = \begin{pmatrix} 0 & G_{i_1} \\ 0 & G_{i_2} \end{pmatrix} \quad X_i = \begin{pmatrix} U_{i_1} \\ U_{i_2} \end{pmatrix} \quad Y_i = \begin{pmatrix} H_{i_1} \\ H_{i_2} \end{pmatrix}$$

The matrices E_{i_4}, F_{i_2}, and G_{i_2} are m by m matrices, and H_{i_2} and U_{i_2} are m by NR matrices. The matrices G_{i_2}, E_{i_4}, F_{i_2}, and H_{i_2} form a block row of the reduced block tridiagonal system, the

solution of which is denoted U_{i_2}, $i = \{1,2,...,P\}$, or X_2. The matrix E_{i_1} is a $q-m$ by $q-m$ matrix, E_{i_2} and G_{i_1} are $q-m$ by m matrices, H_{i_1}, and U_{i_1} are $q-m$ by NR matrices. E_{i_3} and F_{i_1} are m by $q-m$ matrices.

With the notation in Figure 1 the initial assignments are:

$$E_{i_1} = A_{i_{11}},\ E_{i_2} = A_{i_{12}},\ E_{i_3} = A_{i_{21}},\ E_{i_4} = A_{i_{22}},\ (F_{i_1}F_{i_2}) = (B_{i_{21}}0)\text{, and } \begin{pmatrix} G_{i_1} \\ G_{i_2} \end{pmatrix} = \begin{pmatrix} C_{i_{12}} \\ 0 \end{pmatrix}$$

For a symmetric matrix A $\quad B_{i_{21}} = C^T_{i+1_{12}}$, $\quad A_{i_{21}} = A^T_{i_{12}}$, $\quad A_{i_{11}} = A^T_{i_{11}}$, $\quad$ and $\quad A_{i_{22}} = A^T_{i_{22}}$.

3. Local eliminations.

The complete algorithm for the solution of narrow banded systems proceeds in 5 steps as follows:

1. Locally solve in each partition a banded system with system matrix E_{i_1} and the right hand side $G_{i_1}E_{i_2}H_{i_1}$.
2. Locally eliminate the matrix E_{i_3}.
3. Through communication with adjacent partitions eliminate F_{i_1}.
4. Solve the reduced system of equations to obtain $U_{i_2}, i = \{1,2,...,P\}$. (Requires global communication).
5. Solve for the remaining variables $U_{i_1}, i = \{1,2,..,P\}$, X_1, through communication with adjacent partitions.

The first 3 steps constitute phase 1, step 4 phase 2, and step 5 phase 3.

In this section we define the algorithm for phase 1, in which a banded system is transformed into a system of equations from which a block tridiagonal system of order P and block size m by m can be separated out and solved independently. We also give an algorithm for the computation of the remaining variables, phase 3. The arithmetic and communication complexity of the computations are derived. Preservation of diagonal dominance, symmetry, and positive definiteness in phase 1 is proved.

3.1. An algorithm for phases 1 and 3.

Below we specify the computations of each of the steps 1 - 3 and 5 in some detail.

Step 1.

Factor the blocks E_{i_1} and solve the first $q-m$ equations in each partition.

$$E_{i_1} \leftarrow L_{i_1}U_{i_1}, \quad \text{or for } A \text{ symmetric} \quad E_{i_1} \leftarrow L_{i_1}D_{i_1}L^T_{i_1}, \quad i = \{1,2,...,P\}$$
$$G_{i_1} \leftarrow E^{-1}_{i_1}G_{i_1}, \quad E_{i_2} \leftarrow E^{-1}_{i_1}E_{i_2}, \quad H_{i_1} \leftarrow E^{-1}_{i_1}H_{i_1}, \quad i = \{1,2,...,P\}$$

The system of equations now have the form shown in Figure 2.

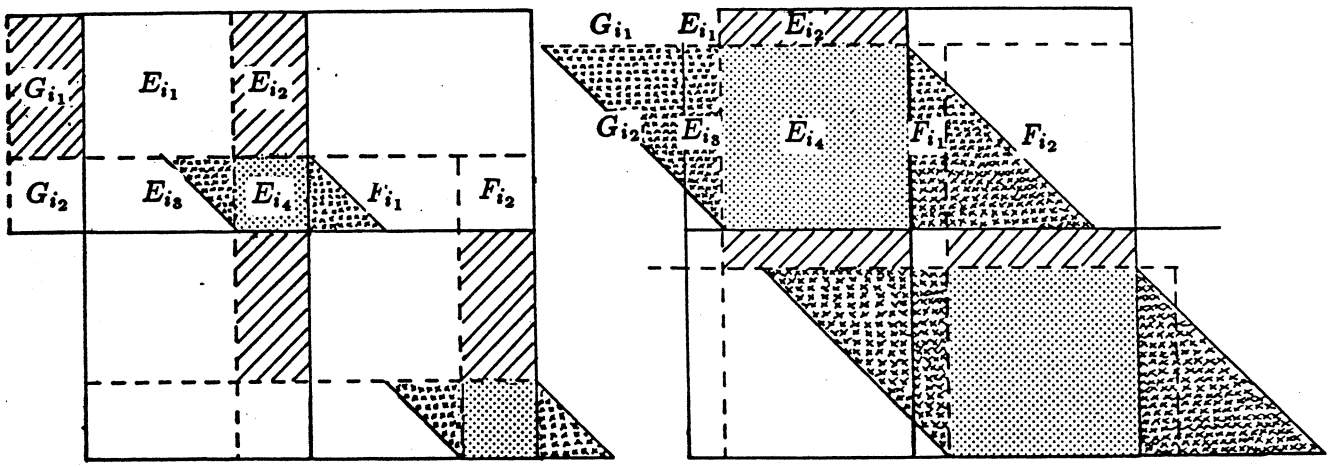

Figure 2: The system of equations after a local solve on the first $N/P - m$ equations.

Step 2.

Eliminate the matrix E_{i_3} by multiplying the first $q - m$ rows in partition i by E_{i_3} and subtracting the result from the last m rows of the partition. The computations are:

$$G_{i_2} \leftarrow G_{i_2} - E_{i_3}G_{i_1}, \quad E_{i_4} \leftarrow E_{i_4} - E_{i_3}E_{i_2}, \quad \text{and} \quad H_{i_2} \leftarrow H_{i_2} - E_{i_3}H_{i_1}, \quad i = \{1, 2, .., P\}$$

Note that if $q \geq 2m$, then G_{i_2} is initially 0, and the first $q - 2m$ columns of E_{i_3} are 0, while the last m columns contain an upper triangular matrix. Hence, effectively the last $min(q - m, m)$ rows of G_{i_1}, E_{i_2}, and H_{i_1} are used for the elimination.

Step 3.

Eliminate the matrix F_{i_1} by multiplying the first $q - m$ rows of partition $i + 1$ by F_{i_1} and subtracting the product from the last m rows of partition i, $i = \{1, 2, .., P - 1\}$. This step requires communication between adjacent partitions. The computations are:

$$E_{i_4} \leftarrow E_{i_4} - F_{i_1}G_{i+1_1}, \quad F_{i_2} \leftarrow F_{i_2} - F_{i_1}E_{i+1_2}, \quad \text{and} \quad H_{i_2} \leftarrow H_{i_2} - F_{i_1}H_{i+1_1}, \quad i = \{1, 2, ..., P-1\}$$

Note that if $q \geq 2m$, then F_{i_2} is initially 0, and the last $q - 2m$ columns of F_{i_1} are 0, while the first m columns contain a lower triangular matrix. Effectively the first $min(q - m, m)$ rows of G_{i+1_1}, E_{i+1_2}, and H_{i+1_1} are used for the elimination.

At this point the form of the system of equations is as shown in Figure 3.

The last m equations of each partition together form a block tridiagonal system of equations of P block rows. Before discussing step 4, the solution of the reduced block tridiagonal system, we consider step 5, the step in which $q - m$ variables for each right hand side are solved for in each partition.

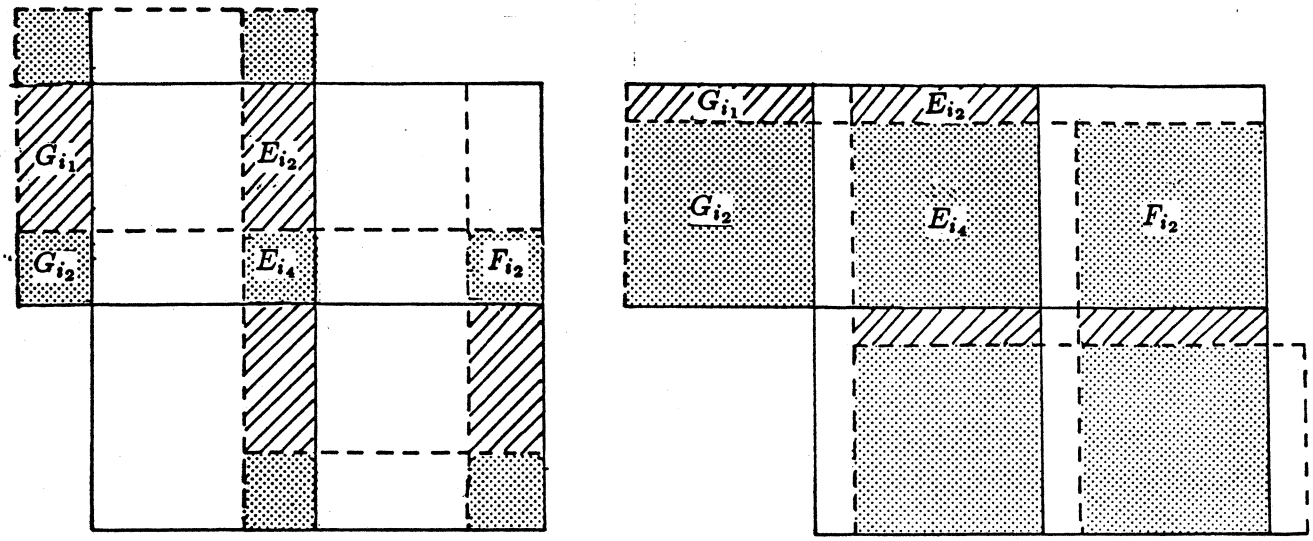

Figure 3: The system of equations after steps 1 - 3.

Step 5.

Having solved the block tridiagonal system, step 4, X_2 is known and X_1 can be computed.

$$U_{1_1} \leftarrow H_{1_1} - E_{1_2}U_{1_2}, \quad \text{and} \quad U_{i_1} \leftarrow H_{i_1} - E_{i_2}U_{i_2} - G_{i_1}U_{i-1_2}, \quad i = \{2,3,...,P\}$$

Complexities of steps 1 - 3 and 5.

Each of the four arithmetic operations ($+, -, \star$, and $/$) requires time t_a. Partitions i and $i+1$, $i = \{1,2,...,P-1\}$, are assumed to be in adjacent processing elements. This assumption is important for the communication in step 3. The time to communicate a floating-point number is t_c. The following complexity estimates can be derived assuming that A is symmetric, and that adjacent partitions are mapped to neighboring processors.

Arithmetic: $((7m^2 + 8mNR + 3m + 2NR + 1)(N/P - 2m) + m((m+1)(4m + (2m+1)/3) + NR(8m+2) + 3) - 2)t_a$.

Communication: $(min(q-m, m)(2m + NR) + mNR)t_c$.

The arithmetic complexity is increased by a term $m^2(N/P - 2m)t_a$ if A is not symmetric. The communication complexity can be reduced by $min(q-m, m)(min(q-m, m) - 1)/2t_c$ if the elimination of F_{i_1} is made in partition $i+1$.

For ensembles configured as boolean cubes and cube-connected cycles networks and linear arrays mapping adjacent partitions into neighboring processing elements is feasible also with respect to the solution of the reduced block tridiagonal system. For a binary tree configured ensemble an inorder mapping of partitions to processing elements is feasible with respect to phase 2. The inorder mapping increases the communication complexity by a factor of $log_2 P - 1$.

3.2. Symmetric positive definite matrices.

If A is symmetric and positive definite it is desirable to exploit this property in solving the system of equations. Next we will show that our substructuring technique allows for the use of Cholesky factorization instead of Gauss-Jordan elimination if the matrix A is symmetric and positive definite. Moreover, the Cholesky factorization is parallelizable in the same way as the

Gauss-Jordan elimination. It follows from the use of Cholesky's method that, at the end of phase 1, the partially factored matrix A is symmetric positive definite and the reduced system can also be solved taking advantage of symmetry and positive definiteness. The reduced system obtained by Sameh et. al. [11] is not necessarily symmetric and positive definite.

We will aslo show that diagonal dominance is preserved.

3.2.1. Preservation of symmetry and positive definiteness.

To show that parallel Cholesky factorization can be performed on A (if it is symmetric and positive definite) both in phases 1 and 2 we perform a permutation on rows and columns such that the first $q-m$ rows and columns of all partitions are ordered before the last m rows and columns of any partition. The order of rows and columns within the subpartitions is preserved, and so is the order of the subpartitions. The permutation matrix R is:

$$R = \left(\begin{array}{ccccccc} I_{q-m} & 0_{q-mm} & 0_{q-m} & 0_{q-mm} & 0_{q-m} & 0_{q-mm} & \cdots \\ 0_{q-m} & 0_{q-mm} & I_{q-m} & 0_{q-mm} & 0_{q-m} & 0_{q-mm} & \cdots \\ 0_{q-m} & 0_{q-mm} & 0_{q-m} & 0_{q-mm} & I_{q-m} & 0_{q-mm} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \hline 0_{mq-m} & I_m & 0_{mq-m} & 0_m & 0_{mq-m} & 0_m & \cdots \\ 0_{mq-m} & 0_m & 0_{mq-m} & I_m & 0_{mq-m} & 0_m & \cdots \\ 0_{mq-m} & 0_m & 0_{mq-m} & 0_m & 0_{mq-m} & I_m & \cdots \end{array}\right)$$

The subscript denotes the dimension of the matrix. For a square matrix one subscript is given, for a rectangular matrix the number of rows are given before the number of columns. Clearly $RR^T = I$, and $RAR^T = \tilde{A}$ is also symmetric positive definite. The matrix $\tilde{A}$ is of the form:

$$\tilde{A} = \left(\begin{array}{cccccc|cccccc} E_{1_1} & & & & & & E_{1_2} & & & & & \\ & E_{2_1} & & & & & G_{2_1} & E_{2_2} & & & & \\ & & E_{3_1} & & & & & G_{3_1} & E_{3_2} & & & \\ & & & \ddots & & & & & \ddots & \ddots & & \\ & & & & E_{P_1} & & & & & G_{P_1} & E_{P_2} & \\ \hline E_{1_3} & F_{1_1} & & & & & E_{1_4} & F_{1_2} & & & & \\ & E_{2_3} & F_{2_1} & & & & G_{2_2} & E_{2_4} & F_{2_2} & & & \\ & & \ddots & \ddots & & & & \ddots & \ddots & \ddots & & \\ & & & E_{P_3} & F_{P_1} & & & & & G_{P_2} & E_{P_4} & \end{array}\right)$$

The matrix $\tilde{A}$ has a symmetric factorization $\tilde{L}\tilde{L}^T$. Any positive definite symmetric N by N matrix M can be factored as (Cholesky):

$$M = \begin{pmatrix} d_1 & v_1^T \\ v_1 & \bar{M}_1 \end{pmatrix} = \begin{pmatrix} \sqrt{d_1} & 0 \\ \frac{v_1}{\sqrt{d_1}} & I_{N-1} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & \hat{M}_1 \end{pmatrix} \begin{pmatrix} \sqrt{d_1} & \frac{v_1^T}{\sqrt{d_1}} \\ 0 & I_{N-1} \end{pmatrix}$$

$$M = L_1 M_1 L_1 = L_1 \begin{pmatrix} 1 & 0 \\ 0 & \hat{M}_1 \end{pmatrix} L_1^T$$

where $\hat{M}_1 = \bar{M}_1 - \frac{v_1 v_1^T}{d}$. $\hat{M}_1$ is clearly symmetric, and it is also positive definite, and the process can be repeated for M_1.

$$M_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & d_2 & v_2^T \\ 0 & v_2 & \hat{M}_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{d_2} & 0 \\ 0 & \frac{v_2}{\sqrt{d_2}} & I_{N-2} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \hat{M}_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{d_2} & \frac{v_2^T}{\sqrt{d_2}} \\ 0 & 0 & I_{N-2} \end{pmatrix}$$

$M_1 = L_2 M_2 L_2^T$ and $M = L_1 L_2 M_2 L_2^T L_1^T$. $L_1 L_2$ is also lower triangular.

For the matrix $\tilde{A}$ the factorization with respect to any column $(q-m)j+k, k = \{1,2,...,q-m\}$ is independent of the factorization with respect to any column $(q-m)i+k, k = \{1,2,...,q-m\}$, $i \neq j$, $i,j = \{1,2,...,P\}$. Hence, the factorization with respect to the different blocks of $q-m$ columns can be performed concurrently. Communication between adjacent partitions is needed due to the blocks F_{i_1} and each E_{i_4}, $i = \{1,2,...,P-1\}$ is updated by partition i and $i+1$. After factorization with respect to the first $(q-m)P$ columns the matrix $\tilde{A}$ is factored as:

$$\tilde{A} = L \begin{pmatrix} I_{(q-m)P} & 0 \\ 0 & \tilde{A}_2 \end{pmatrix} L^T$$

where $\tilde{A}_2$ is symmetric, positive definite and of order mP by mP. It defines the reduced system to be solved in phase 2. It is readily verified to be block tridiagonal with block size m by m. The matrix L is of the form:

$$L = \left(\begin{array}{ccccc|ccc} L_{1_1} & & & & & & & \\ & L_{2_1} & & & & & & \\ & & L_{3_1} & & & & & \\ & & & \ddots & & & & \\ & & & & L_{P_1} & & & \\ \hline L_{1_3} & L_{1_2} & & & & I_m & & \\ & L_{2_3} & L_{2_2} & & & & I_m & \\ & & \ddots & \ddots & & & & \ddots \\ & & & & L_{P_3} & & & I_m \end{array}\right)$$

The solution to the banded system can be obtained as follows:

$$L \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \qquad \begin{pmatrix} I & 0 \\ 0 & \tilde{A}_2 \end{pmatrix} \begin{pmatrix} S_1 \\ S_2 \end{pmatrix} = \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} \qquad L^T \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} S_1 \\ S_2 \end{pmatrix}$$

It follows that $S_1 = Z_1$, $\tilde{A}_2 S_2 = Z_2$ and $X_2 = S_2$. Hence, after the forward substitution the set of variables $U_{i_2}, i = \{1,2,...,P\}$ are obtained from the solution of the reduced system.

The reduced block tridiagonal system can be solved through continued symmetric factorization. But, the factorization with respect to any column in parition i, $(q-m)P+m(i-1)+k, k = \{1,2,...,m\}$ affect all the columns in partition $i+1$, i.e., columns $(q-m)P+mi+k, k = \{1,2,...,m\}$. However, a permutation of the last P block row and columns can be performed to yield the elimination order of block cyclic reduction. Then, approximately $P/2$ blocks can be factored concurrently, then $P/4$, etc.

The permutation of the last row and columns corresponding to block cyclic reduction is the same as if nested dissection is applied to the graph corresponding to the banded matrix, and the first bisector labeled last and the last set of bisectors labeled first.

3.2.2. Diagonal dominance.

We will now show that diagonal dominance in a certain sense is preserved during phase 1 of the algorithm.

Theorem 3.1. *Let the matrix A satisfy the following diagonal dominance conditions:*

$$\|E_{1_1}^{-1}E_{1_2}\| < 1, \quad \|E_{1_4}^{-1}E_{1_3}\| + \|E_{1_4}^{-1}F_{1_1}\| + \|E_{1_4}^{-1}F_{1_2}\| < 1$$
$$\|E_{i_1}^{-1}G_{i_1}\| + \|E_{i_1}^{-1}E_{i_2}\| = a_i + b_i < 1, \qquad i = \{2,3,...,P\}$$
$$\|E_{i_4}^{-1}G_{i_2}\| + \|E_{i_4}^{-1}E_{i_3}\| + \|E_{i_4}^{-1}F_{i_1}\| + \|E_{i_4}^{-1}F_{i_2}\| = c_i + d_i + e_i + f_i < 1, \quad i = \{2,3,...,P-1\}$$
$$\|E_{P_4}^{-1}G_{P_2}\| + \|E_{P_4}^{-1}E_{P_3}\| < 1$$

Then the nonzero submatrices of $\tilde{A}_2$, $\tilde{E}_{i_4}$, $\tilde{F}_{i_2}$, *and* G_{i_2} *satisfy the relations:*

$$\|\tilde{E}_{1_4}^{-1}\tilde{F}_{1_2}\| < 1, \qquad \|\tilde{E}_{i_4}^{-1}\tilde{G}_{i_2}\| + \|\tilde{E}_{i_4}^{-1}\tilde{F}_{i_2}\| < 1, \quad i = \{2,3,...,P-1\} \qquad \|\tilde{E}_{P_4}^{-1}\tilde{G}_{P_2}\| < 1$$

Proof.

We will prove that the dominance conditions hold for $i = \{2,3,...,P-1\}$. The cases $i = 1$ and $i = P$ are proved similarly.

$$\tilde{E}_{i_4} = E_{i_4} - E_{i_3}E_{i_1}^{-1}E_{i_2} - F_{i_1}E_{i+1_1}^{-1}G_{i+1_1}, \quad i = \{1,2,...,P-1\}$$
$$\tilde{G}_{i_2} = G_{i_2} - E_{i_3}E_{i_1}^{-1}G_{i_1}, \qquad i = \{2,3,...,P\}$$
$$\tilde{F}_{i_2} = F_{i_2} - F_{i_1}E_{i+1_1}^{-1}E_{i+1_2}, \quad i = \{1,2,...,P-1\}$$

We need to show that:

$$\|(E_{i_4} - E_{i_3}E_{i_1}^{-1}E_{i_2} - F_{i_1}E_{i+1_1}^{-1}G_{i+1_1})^{-1}(G_{i_2} - E_{i_3}E_{i_1}^{-1}G_{i_1})\| +$$
$$+\|(E_{i_4} - E_{i_3}E_{i_1}^{-1}E_{i_2} - F_{i_1}E_{i+1_1}^{-1}G_{i+1_1})^{-1}(F_{i_2} - F_{i_1}E_{i+1_1}^{-1}E_{i+1_2})\| < 1$$
or
$$\|(I - E_{i_4}^{-1}E_{i_3}E_{i_1}^{-1}E_{i_2} - E_{i_4}^{-1}F_{i_1}E_{i+1_1}^{-1}G_{i+1_1})^{-1}E_{i_4}^{-1}(G_{i_2} - E_{i_3}E_{i_1}^{-1}G_{i_1})\| +$$
$$+\|(I - E_{i_4}^{-1}E_{i_3}E_{i_1}^{-1}E_{i_2} - E_{i_4}^{-1}F_{i_1}E_{i+1_1}^{-1}G_{i+1_1})^{-1}E_{i_4}^{-1}(F_{i_2} - F_{i_1}E_{i+1_1}^{-1}E_{i+1_2})\| < 1$$

or with the shorthand notation introduced above:

$$\|(I - d_ib_i - e_ia_{i+1})^{-1}(c_i - d_ia_i)\| + \|(I - d_ib_i - e_ia_{i+1})^{-1}(f_i - e_ib_{i+1})\| = g_i$$

But, $$\|(I - d_ib_i - e_ia_{i+1})^{-1}\| \le \frac{1}{1 - \|d_ib_i + e_ia_{i+1}\|} \le \frac{1}{1 - d_ib_i - e_ia_{i+1}}$$

and $$g_i \le \frac{1}{1 - d_ib_i - e_ia_{i+1}}(c_i + d_ia_i + f_i + e_ib_{i+1})$$

From the diagonal dominance assumptions it follows that:

$$d_ib_i + e_ia_{i+1} + c_i + d_ia_i + f_i + e_ib_{i+1} = c_i + d_i(a_i + b_i) + e_i(a_{i+1} + b_{i+1}) + f_i < c_i + d_i + e_i + f_i < 1$$

$$\text{or} \qquad c_i + d_ia_i + f_i + e_ib_{i+1} < 1 - d_ib_i + e_ia_{i+1}$$

∎

4. Solution of the block tridiagonal system, step 4.

The feasibility of different methods for solving the block tridiagonal system depends on numerical aspects, and the configuration of the ensemble of processing elements. We will compare the complexities of 2GE and BCR on linear arrays, boolean cubes, perfect shuffle and shuffle-exchange networks and binary trees.

4.1. Linear Array.

4.1.1. 2-way Gaussian Elimination.

The blocks are treated as dense blocks. We assume that $P = 2^p - 1$ in order to simplify the comparison between 2GE and BCR. In 2GE the elimination process proceeds towards the middle from both ends, followed by backsubstitution from the middle towards the ends. Since P is odd, the elimination phase terminates with one block system to be solved. For $P = 3$ 2GE and BCR are identical. Assuming that:

1. There is one block row per processing element;
2. The solve operation on a block row with the off-diagonal blocks appended to the right hand sides is carried out in the processing element storing the row;
3. The matrix multiplication and addition required for the elimination of an off diagonal block is performed in the processing element storing the block to be eliminated;

We derive the following complexity estimates:

Arithmetic: $(m(7m^2/3+3mNR-(9m+6NR+1)/12)(P-1)+m(2mNR+(4m^2+3m-1)/6))t_a$.

Communication: (number of elements) $\times$ (distance) $(m(m+2NR)(P-1)/2+m^2)t_c$.

The minimum number of interprocessor communications is $P-1$.

4.1.2. BCR.

For BCR we base the complexity estimates on the following assumptions:

1. There is one block row per processing element.
2. The solve operation on a block row with the off-diagonal blocks appended to the right hand sides is carried out in the processing element storing the row.
3. The matrix multiplication and addition required for the elimination of an off diagonal block is performed in the processing element storing the block row used for the elimination.
4. The elimination is performed in the block row storing the blocks to be eliminated.

The first two assumptions are the same as for the Gaussian elimination algorithm. The third assumption allows concurrent operations on the two rows used for the elimination of the two off-diagonal blocks to be eliminated in a row. This assumption implies that m^2 elements need to be communicated between the processors storing the blocks to be eliminated and the processors storing the rows used for the elimination. This communication is not necessary in the Gaussian elimination algorithm above, and is the price paid for operating concurrently on the two rows used for the elimination of the two off-diagonal blocks in a row. Elimination of the off diagonal blocks is then accomplished in the processing element storing the block row in which elimination is to take place by adding block rows received in communication with the appropriate processing elements. In the backsubstitution phase two vectors of dimension m of previously computed unknowns are

needed. The vectors can be communicated over different paths. Cyclic reduction on a linear array can be carried out by explicitly implementing shuffle operations, or *in-place*. In such an algorithm partitions reside in a given processor until needed in the reduction process by another processor. In [8] we proved that the in-place algorithm is more efficient on a linear array than a shuffle based algorithm. Our complexity estimates are as follows:

Arithmetic: $(m(26m^2/3+8mNR-3m-2/3)(log_2P-2)+m((16m^2-1)/3-m+NR(8m-1)))t_a$.

Communication: (number of elements) $\times$ (distance) $((m(5m+4NR)(P+1)/4-m(3m+2NR))t_c$.

The minimum number of interprocessor communications is $P-1$.

4.1.3. Comparing the complexities of 2GE and BCR on a linear array.

The parallel arithmetic complexities of both Gaussian elimination and BCR as given above can be reduced somewhat by distributing the computations over additional processing elements at the expense of additional communication.

Theorem 4.1. *The parallel arithmetic complexity of BCR is at most the same as that of 2GE for* $P \geq 3$.

Proof. The proof is by induction. The arithmetic complexities are the same for $p = 2$, i.e., $P = 3$, and the proposition is true. Assume it is true for $p = n$, $n > 2$. Then the increment in arithmetic complexity is easily shown to be less for BCR than for 2GE, and the proposition is true.

∎

If the number of interprocessor communications is considered, then 2GE and BCR are equivalent, and if the communication bandwidth requirement is measured by the number of elements communicated times the distance, then 2GE is the most efficient. The communication bandwidth requirement is quadratic in the matrix bandwidth, whereas the arithmetic complexity is cubic. If the architecture allows concurrent computation and communication the time for arithmetic will dominate if the communication bandwidth is high. If the bandwidth is small the sequential dependences are such that it is necessary to add at least the time for communicating one floating-point number. If the architecure does not allow concurrent communication and computation – a pessimistic assumption – then, considering only highest order terms, BCR may still be preferable.

Theorem 4.2. *Considering only highest order terms, there exists some* P_0 *such that for* $P \geq P_0$ *BCR is of a lower total complexity (arithmetic + communication) than 2GE on a linear array, if* $\alpha = t_c/t_a < 28m/9$ *and* $NR \ll m$, *and* $\alpha < 4NR$ *for* $NR \gg m$.

Proof. Follows directly from the complexity estimates.

∎

Corollary 4.1. *BCR is of a lower total complexity than 2GE on any linear array for which* $\alpha < 1, P \geq 3$.

Proof. It is true for $m = 1$, [8]. The complexity of BCR increases at a lower rate with the matrix bandwidth than 2GE on a linear array.

∎

Hence, BCR is of a lower complexity than Gaussian elimination even under more pessimistic assumptions than those in [11]. Figure 4 shows the boundary between the regions in which each method is preferable with respect to total complexity. The curves are computed from the complexity estimates given above and with the assumption that communication and computation do not take place concurrently. The region in which BCR is of lower complexity increases with NR, and decreases with α.

Note, that with communication and computations actually occuring concurrently the region in which BCR is of a lower complexity than 2GE is increased.

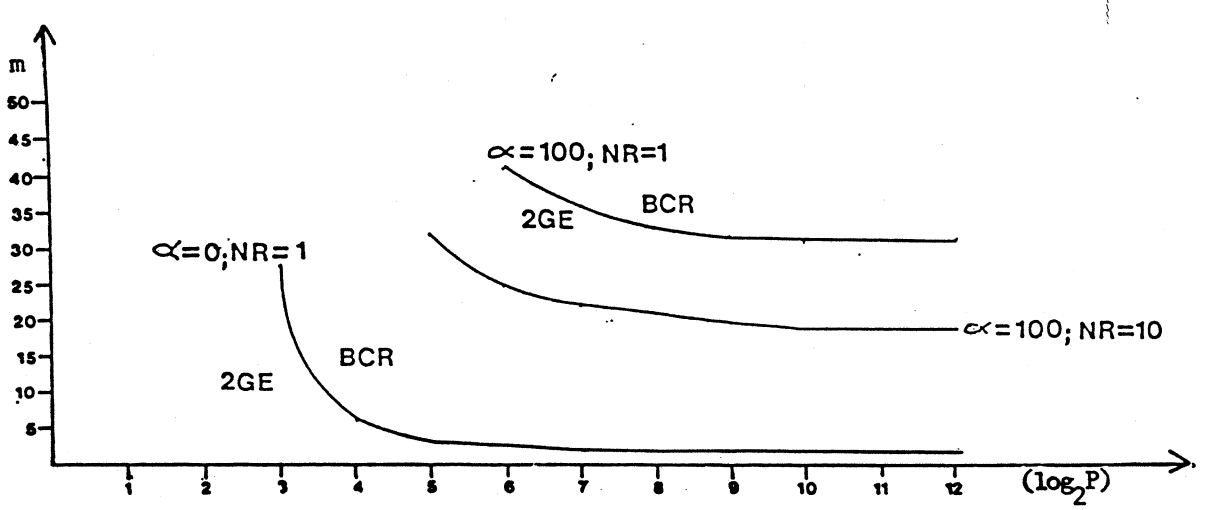

Figure 4: Regions of lowest total complexity for 2GE and BCR on a linear array.

4.2. Boolean cube and Perfect Shuffle Networks.

We choose to perform the elimination of off diagonal blocks in the same manner as on a linear array. Hence, the arithmetic complexity is the same for a linear array, a boolean cube and a perfect shuffle network with the same number of processing elements. The communication complexities of BCR on boolean cube and perfect shuffle networks are the same - namely:

Communication: (number of elements)$\times$(distance) $(m(3m+2NR)(log_2P-2)+2m(m+NR))t_c$

The minimum number of interprocessor communications is $4(log_2P - 2) + 2$.

BCR is always of a lower total complexity than 2-way Gaussian elimination on these networks.

4.3. Binary Trees and Shuffle-Exchange Networks.

For a binary tree of processors we assume an inorder mapping of partitions to processors, following [8]. A binary tree is a natural computation structure for partitioning by nested dissection, in which case the first bisector is mapped into the root, and successive levels of bisectors are mapped on to successive levels of the tree. The last set of bisectors are mapped to the leaves of the tree. Substructures not being bisectors are also mapped to the leaf nodes (except 1 such substructure that is mapped to a parent node of a leaf node).

For the binary tree we make the same assumptions regarding the distribution of computations for the elimination of a row as for Gaussian elimination on a linear array. The distribution used for BCR on linear arrays, boolean cubes and perfect shuffle networks requires communication in both directions, which prevents effective pipelining of operations on a binary tree with an inorder mapping of equations. The following complexity estimates can be derived:

Arithmetic: $2(m(38m^2/3+8mNR-2m-2/3)(log_2P-2)+m((16m^2-1)/3-m+NR(8m-1)))t_a$

The minimum number of interprocessor communications is $3log_2P-4$.

BCR on a shuffle-exchange network can be performed by simulating a binary tree. An effective embedding is obtained by labeling the root of the tree 1, label the left descendant of any node labeled p by $2p$ and the right descendant $2p+1$. The distance from a parent node to its right child is 2, and the communication complexity is somewhat higher than for the tree.

5. Summary of computational complexities for narrow banded systems.

Adding the complexities for all 5 steps, and assuming that the communication and arithmetic complexities are additive, the highest order terms for 2GE are $7m^2N/P+(7/3m^3+m^2\alpha/2)P$ for 1 right hand side. The optimum value of P_{opt} is $\sqrt{3N/(m+3\alpha/14)}$, which is approximately $\sqrt{3N/m}$ for large matrix bandwidths, or communciation of a speed comparable to the time for arithmetic. The minimum time T_{min} is approximately $14m^2\sqrt{Nm/3}$. For block cyclic reduction on a linear array the highest order terms are $7m^2N/P+26/3m^3log_2P+5/4m^2\alpha P$, and the optimum value of P, $P_{opt} \approx 52m/(15\alpha)(\sqrt{28N/(5\alpha(15\alpha/52m)^2)+1}-1)$, is obtained as the solution to a second order equation. For $m^2 \ll N\alpha/2$ $P_{opt}=\sqrt{28N/(5\alpha)}$ and $T_{min} \approx 6m^2\sqrt{N\alpha}+14/3m^3log_2N/\alpha$. For $m^2 \approx N\alpha/2$ $P_{opt}=\beta\sqrt{28N/(5\alpha)}$, and $T_{min} \approx (\beta+1/\beta)6m^2\sqrt{N\alpha}+14/3m^3log_2N/\alpha$. Finally, for $m^2 \gg N\alpha/2$ $P_{opt}=21N/26m$, and $T_{min} \approx 26/3m^3(1+log_2(N/m))$.

For a boolean cube and a perfect shuffle network the highest order terms are $7m^2N/P+(26/3m^3+3m^2\alpha)log_2P$, and $P_{opt}=21N/(26m+9\alpha)$, and $T_{min} \approx (26/3m^3+3m^2\alpha)(1+log_2(N/m))$. For the binary tree the highest order terms are $7m^2N/P+(76/3m^3+2m^2\alpha)log_2P$. The value of P_{opt} for a binary tree is approximately $21N/(76m+6\alpha)$, which for m/α large is approximately 1/3 of the value for a boolean cube or perfect shuffle network, and for m/α small 1.5 times larger than for the two other networks. Note that the partitioning method as used here is only valid if $P \leq N/m$. Table 1 summarizes the results for 1 right hand side.

For a large number of right hand sides, $NR \gg m$, the highest order terms for 2GE are $8mNRN/P+3m^2NRP+mNR\alpha P$, and for BCR on a linear array $8mNRN/P+8m^2NRlog_2P+mNR\alpha P$, on a perfect shuffle or boolean cube $8mNRN/P+(8m^2NR+2mNR\alpha)log_2P$, and on a tree $8mNRN/P+(16m^2NR+2mNR\alpha)log_2P$, respectively. Optimization with respect to highest order terms yields $P_{opt}=\sqrt{8N/(3m+2\alpha)}$ with $T_{min} \approx 4mNR\sqrt{2N(3m+2\alpha)}$ for 2GE. For BCR on a linear array and $m^2 \ll N\alpha/2$ $P_{opt}=\sqrt{8N/\alpha}$, and $T_{min} \approx 4mNR\sqrt{2N\alpha}+4m^2NRlog_2(N/\alpha)$. For $m^2 \approx N\alpha/2$ $P_{opt}=\beta\sqrt{8N/\alpha}$, and $T_{min} \approx (\beta+1/\beta)4mNR\sqrt{2N\alpha}+4m^2NRlog_2(N/\alpha)$. For $m^2 \gg N\alpha/2$ $P_{opt}=N/m$, and $T_{min} \approx 8m^2NR(1+log_2(N/m))$. For boolean cubes and perfect shuffle networks $P_{opt}=8N/(8m+2\alpha)$, and $T_{min} \approx (8m^2NR+2mNR\alpha)(1+log_2(N/(m+\alpha/4)))$. For a binary tree finally $P_{opt}=8N/(16m+2\alpha)$ with a corresponding difference in T_{min}. Table 2 summarizes the results for multiple right hand sides.

Configuration	Optimum Number of Processors	Minimum time
Linear array		
2GE	$\sqrt{3N/(m+3\alpha/14)}$	$14m^2\sqrt{N(m/3+\alpha/14)}$
BCR $m^2 \ll N\alpha/2$	$\sqrt{28N/(5\alpha)}$	$6m^2\sqrt{N\alpha}+14/3m^3log_2N/\alpha$
$m^2 \approx N\alpha/2$	$\beta\sqrt{28N/(5\alpha)}$	$(\beta+1/\beta)6m^2\sqrt{N\alpha}+14/3m^2log_2N/\alpha$
$m^2 \gg N\alpha/2$	$21N/26m$	$26/3m^3(1+log_2(N/m))$
Boolean cube Perfect Shuffle	$21N/(26m+9\alpha)$	$(26/3m^3+3m^2\alpha)(1+log_2(N/(m+\alpha/3)))$
Binary tree	$21N(76m+6\alpha)$	$(76/3m^3+3m^2\alpha)(1+log_2(N/(m+\alpha/13)))$

Table 1: Estimates of optimum number of processors and solution time, one problem.

Configuration	Optimum Number Processors	Minimum time
Linear array		
2GE	$\sqrt{8N/(3m+2\alpha)}$	$4mNR\sqrt{2N(3m+2\alpha)}$
BCR $m^2 \ll N\alpha/2$	$\sqrt{8N/\alpha}$	$4mNR\sqrt{2N\alpha}+4m^2NRlog_2(N/\alpha)$
$m^2 \approx N\alpha/2$	$\beta\sqrt{8N/\alpha}$	$(\beta+1/\beta)4mNR\sqrt{2N\alpha}+4m^2NRlog_2(N/\alpha)$
$m^2 \gg N\alpha/2$	N/m	$8m^2NR(1+log_2(N/m))$
Boolean cube Perfect shuffle	$8N/(8m+2\alpha)$	$(8m^2NR+2mNR\alpha)(1+log_2(N/(m+\alpha/4)))$
Binary tree	$8N(16m+2\alpha)$	$(16m^2NR+2mNR\alpha)(1+log_2(N/(m+\alpha/8)))$

Table 2: Estimates of optimum number of processors and solution time, multiple problems.

Comparing the estimates in the two tables we conclude that the value of P_{opt} only changes to a small extent with the number of right hand sides.

The estimates in Table 2 are based on the assumption that the entire ensemble is used for the solution of each problem. Alternatively the ensemble can be partitioned such that a partition solves a subset of the problems. The optimum partitioning depends on the values of m, NR, t_a, and t_c. For $\alpha \rightarrow \infty$ $NEP_{opt} \rightarrow P$, where NEP is the number of ensemble partitions, i.e., for sufficently slow communication the optimum ensemble partitioning is obtained by dividing the set of problems among the processors and solving each set locally. However, if t_a and t_c are of the same order, then $NEP_{opt} \neq P$. The optimum value is found as the solution of a higher order equation, even if only the arithmetic terms are considered.

6. Summary and Conclusions.

We described concurrent algorithms suitable for the solution of narrow banded systems. In such systems the indepedence of the operations in the elimination of different variables from the system of equations is the main source of concurrency, whereas relatively few operations are required for the elimination of a single variable. The potential concurrency in such an operation is low.

The algorithms are based on partitioning the sets of equations into subsets, substructures. The substructuring used for the algorithm is closely related to nested dissection. Similarities and differences between the different substructuring techniques are discussed. A block tridiagonal subsystem is the result of concurrent Gauss-Jordan elimination or Cholesky factorization in a first phase in which most operations are local to a substructure. Some communication with adjacent substructures is required.

For symmetric positive definite matrices it is shown that Cholesky factorization can be performed concurrently in all substructures in phase 1 of the algorithm. The reduced system can also be factored by Cholesky factorization in a sequence of steps with successively decreasing concurrency by using a cyclic reduction procedure. In the context of nested dissection this procedure corresponds to factoring the last set of bisectors first, and the first bisector last.

We proved that the algorithm preserves diagonal dominance.

We showed that for ensembles configured as linear arrays the computational complexity, including communication, is lower for block cyclic reduction than 2-way Gaussian elimination for $P \geq P_0$, and $t_c/t_a < 28m/9$ if $NR \ll m$ or $t_c/t_a = 4NR$ if $NR \gg m$. If $t_c/t_a \leq 1 then P_0 = 1$, i.e., block cyclic reduction is always preferable.

We showed that block cyclic reduction on boolean cubes, perfect shuffle and shuffle-exchange networks, and binary trees is of a lower complexity (because of lower communication complexity), than on a linear array. The optimum number of partitions is increased from $O(\sqrt{N/m})$ for Gaussian elimination on a linear array to $O(N/m)$ (assuming one processor per partition) for block cyclic reduction on the networks of diameter $log_2 P$. The minimum time is $O(m^3 + m^3 log_2(N/m))$.

The complexity advantage of block cyclic reduction increases if the reduced block tridiagonal system is sufficiently diagonally dominant to allow for a truncation of the reduction process, [6]. In a fully parallel implementation of (block) cyclic reduction the solution time is proportional to the number of reduction steps. The benefit of truncating the reduction process is substantially greater than in a sequential implementation, since half the number of arithmetic operations are performed in the first reduction step.

With increasing values of $t_c/t_a = \alpha$ the value of P_{opt} decreases and T_{min} increases. The dependence is by a factor of $\sqrt{\alpha}$ for a linear array, and by a factor of α for boolean cubes, perfect shuffle and shuffle-exchange networks, and binary trees.

For multiple right hand sides partitioning of the set of problems into subsets and the processing ensemble into subensembles is always benefitial with respect to computational complexity if $m = 1$ [8]. For $m > 1$ this is not true, in general, since the computations required for the factorization of the system matrix may be dominating the operations involving the right hand sides to the extent that the reduced communication needs do not make up for the increased amount of arithmetic per processor.

The boolean cube, perfect shuffle and shuffle-exchange networks, and the binary tree configurations offer a high speed-up for small values of m/N for the substructuring technique used for the

algorithms. For large values of m/N the speed-up is poor. The substructuring technique applied above exploit the independence of the operations required for the elimination of different variables from the system of equations. For banded systems with $m/N \approx 1$ there are only a few variables that can be eliminated concurrently, but the amount of work required for the elimination of each is of order $O(N^2)$. The independence of the operations in the elimination of a single variable is the main source of concurrency [9].

Acknowledgement

The author wishes to acknowledge the interest of Jack Dongarra in this work, which prompted the proof of preservation of diagonal dominance, symmetry and positive definiteness of the algorithms. Implementation of the substructuring technique and a few different block tridiagonal solvers for different parallel architectures are being made as a joint effort. The implementation and experimental evaluation were initiated during a visit to the Mathematics and Computer Science division of the Argonne National Laboratories.

Thanks also go to Andrea Pappas for her assistance with the manuscript.

The generous support of the Office of Naval Research under contract N00014-84-K-0043 is gratefully acknowledged.

References

[1] Dongarra J.J., Sameh A.H., *On Some Parallel Banded System Solvers,* Technical Report ANL/MCS-TM-27, Argone National Laboratories, Mathematics and Computer Science Division, 1984.

[2] Flynn M.J., *Very High-Speed Computing Systems,* Proc. of the IEEE, 12 (1966), pp. 1901–1909.

[3] Gentleman W.M., *Some Complexity Results for Matrix Computations on Parallel Processors,* J. ACM, 25/1 January (1978), pp. 112–115.

[4] George A., Liu J.W., *Computer Solution of Large Sparse Positive Definite Solutions,* Prentice-Hall, 1981.

[5] George A., *Nested dissection of a regular finite element grid,* SIAM J. on Numerical Analysis, 10 (1973), pp. 345–363.

[6] Heller D., *Some Aspects of the Cyclic Reduction Algorithm For Block Tridiagonal Linear Systems,* SIAM J. Num. Anal, 13/4 (1976), pp. 484–496.

[7] Johnsson S.L., *Computational Arrays for Band Matrix Equations,* Technical Report 4287:TR:81, Computer Science, California Institute of Technology, May 1981.

[8] ————, *Odd-Even Cyclic Reduction on Ensemble Architectures andthe Solution Tridiagonal Systems of Equations,* Technical Report YALE/CSD/RR-339, Department of Computer Science, Yale University, October 1984.

[9] ————, Band Matrix Systems Solvers on Ensemble Architectures, *Algorithms,Architecture, and the Future of Scientific Computation,* University of Texas Press, 1985. Technical report YALEU/CSD/RR-367, Yale University, Dept. of Computer Science.

[10] ————, *Fast Banded Systems Solvers for Ensemble Architectures,* Technical Report YALEU/CSD/RR-379, Department of Computer Science, Yale University, March 1985.

[11] Lawrie D.H.,Sameh A.H., *The Computational and Communication Complexity of a Parallel Banded System Solver,* ACM Trans. Math. Software, 10/2 June (1984), pp. 185–195.

[12] Lipton R.J., Rose D..J., Tarjan R.E., *Generalized nested dissection,* SIAM J. Numerical Analysis, 16 (1979), pp. 346–358.

[13] Reiter E., Rodrigue G., An Incomplete Cholesky Factorization By a Matrix Partitioning Algorithm, *Elliptic Problem Solvers II,* Academic Press, 1983, pp. 161–174.

[14] Sameh A.H., Kuck D.J., *On Stable Parallel Linear System Solvers,* JACM, 25/1 January (1978), pp. 81–91.

[15] Wang H.H., *A Parallel Method for Tridiagonal Equations,* ACM Trans. Math. Software, 7/2 June (1981), pp. 170–183.